Available online at www.sciencedirect.com
ScienceDirect
Computer Networks 50 (2006) 3024–3042
Computer Networks
www.elsevier.com/locate/comnet

A practical scheme for MPLS fault monitoring and alarm correlation in backbone networks ☆

Ming Yu [a,*], Wenjui Li [b], Li-jin W. Chung [b]

[a] Department of Electrical and Computer Engineering, State University of New York, Binghamton, NY 13902, United States
[b] AT&T Labs, Middletown, NJ 07748, United States

Received 18 September 2004; received in revised form 26 May 2005; accepted 25 November 2005
Available online 28 December 2005

Responsible Editor: I. Matta

Abstract

As current backbone network evolution involves replacing today's multiple networks with a single global multi-protocol label switching (MPLS)-enabled backbone over an intelligent optical IP-based core network, fault management system (FMS) becomes critical for network service providers to monitor network health, performance, and to quickly identify and resolve operational problems.

In this paper, we present a practical scheme for the fault management of MPLS-enabled backbone networks. First, we describe a hierarchical fault management architecture that scales well to large backbone networks. Then, we present an OAM tool, called MPLS Connectivity Monitor (CMON), to monitor MPLS operation and generate MPLS alarms. After that, we propose a hybrid technique to efficiently correlate MPLS alarms to other equipment and service alarms, including event aggregation, rule-based method, and codebook approach. Finally, we report our testing result obtained from a large-scale backbone network to demonstrate the effectiveness of the proposed scheme.

Keywords: MPLS; Network; Fault management; Event correlation

Abbreviations: AE, aggregated event; AI, artificial intelligence; AMS, alarm management system; BDI, Backward Defect Indicator; BFD, Bidirectional Forwarding Detection; BGP, Border Gate Protocol; CCS, customer care systems; CE, customer's edge (router); CLI, Command Line Interface; CMIP, Common Management Information Protocol; CV, Connectivity Verification; EMS, element management system; FDI, Forward Defect Indicator; FEC, Forwarding Equivalence Class; FMS, fault management system; FTN, FEC-To-NHLFE; ICMP, Internet Control Message Protocol; KPI, Key Performance Index; LDP, Label Distribution Protocol; LSP, Label Switched Path; LSR, Label Switched Router; MIB, management information bases; MPLS, multi-protocol label switching; NCS, network care systems; NE, network elements; NMS, network management system; NOC, network operation center; OAM, operation, administration and maintenance; OSPF, Open Shortest Path First; OSS, operation supporting system; P, provider (router); PE, provider's edge (router); PWE3, Pseudo Wire Emulation Edge-to-Edge; SLA, service level agreement; SMI, Structure of Management Information; SNMP, Simple Network Management Protocol; TE, Traffic Engineering; VCCV, VC Connectivity Verification; VPN, virtual private network; VRF, VPN Routing and Forwarding.

☆ This work was done while the first author was with AT&T Labs, Middletown, NJ 07748.

* Corresponding author. Tel.: +1 607 777 6133; fax: +1 607 777 4464.
E-mail addresses: mingyu@binghamton.edu (M. Yu), wenjuili@att.com (W. Li), lic@att.com (L.-j.W. Chung).

1389-1286/$ - see front matter © 2005 Elsevier B.V. All rights reserved.
doi:10.1016/j.comnet.2005.11.006

1. Introduction

A large backbone network can be defined as a collection of high-bandwidth links that connect a number of routers throughout a large geographical area, maybe as large as between continents. The bandwidth of the backbone must be high enough to support all the traffic that goes through the backbone. Large backbone network operators often own multiple infrastructures, such as IP, ATM, and frame relay networks, over the years of network evolutions.

The driving forces to the large-scale deployment of MPLS in backbone networks come from two aspects. The first is to achieve fast deployment of new revenue-generating services with cost-effective operation, administration and maintenance (OAM) for network service providers. Thus, there is a need to integrate the existing multiple networks, including IP, ATM, and frame relay networks, into a single network. The second is to provide manageable IP QoS architecture and mechanism to network customers. The network integration and deployment of MPLS technologies in backbone networks have introduced great challenges to network OAM. Therefore, fault management becomes critical in monitoring network health, performance, and quickly identifying and resolving operational problems.

The goal of fault management is to collect, detect, and respond to fault conditions in the network, which are reported as trap events or alarm messages. These messages may be generated by a managed object or its agent built in a network device, such as Simple Network Management Protocol (SNMP) traps or Common Management Information Protocol (CMIP) event notifications [1]; or by network management system (NMS), such as synthetic traps or probing events generated by HP OpenView stations. Fault management systems handle network failures by generating, collecting, processing, identifying, and reporting the trap and alarm messages.

1.1. Requirements on fault management system

In order to not only reduce operation cost, but also meet SLA requirement, a key performance index (KPI) is measured by the extent of end-to-end process automation, from trap generation, alarm processing, fault identification, to trouble ticket creation and close. To improve the KPI, network service providers require new solutions and tools for FMS to monitor the MPLS-enabled networks more effectively.

Specifically, the requirements on FMS can be summarized as

Scalability: FMS must support multiple existing networks, including IP, ATM, and frame relay networks. Each network may have hundreds of provider's edge (PE) and thousands of customer's edge (CE) routers with complex network topologies and multiple protocol layers. Also, the FMS must support a large number of MPLS VPNs in the network. Therefore, the FMS has to scale with the network sizes.

Correlation capability: The FMS should be able to correlate fault events from all types of fault sources across different network platforms, different switches and routers, and different protocol layers, including MPLS protocols. One example is the correlation between the physical layer and the application service layer, which also needs to correlate fault events from both network care systems (NCS) and customer care systems (CCS).

Intelligence: An effective FMS must be able to pinpoint the root cause among many alarms and identify the problems that need to be fixed in order to sustain the network service.

Integration capability: The FMS should be able to work seamlessly with other MPLS diagnostic tools or systems. One of the tools is MPLS Connectivity Monitor (CMON) (see Section 3.2), which monitors both the physical and logical connectivity among the PE routers.

1.2. Current architecture of fault management

Since MPLS is an IP-based technology and all the MPLS control protocols are based on the IP protocol suite, the Internet-standard framework for network management can be still used for the fault management of MPLS networks.

In the existing multiple network environment, each domain network, including IP, ATM, and frame relay network, has its own NMS and thus FMS. The fault management is in a flat structure, in which each element management system (EMS) collects all the traps generated by the element, filters unnecessary traps, and sends the rest to the network's NMS, which processes the traps from all the EMS within the network and then forward the resulted alarms to the network's operation

supporting system (OSS), which is typically an alarm processing and ticketing system. Currently, for a typical backbone network, the OSS within the network operation center (NOC) normally receives over a million of alarms a day, which is far beyond the capability of any existing event correlation engine within the OSS. As more and more monitoring and diagnosis systems are added to the network, the number of alarms could be continually increasing. Moreover, there is a need to integrate the existing multiple domain NMS's, for IP, ATM, and frame relay networks, into a single enterprise NMS, called alarm management system (AMS). Thus, a practical scheme to solve the fault management problems is in critical need.

1.3. Related work in alarm correlation

As the major component of FMS, alarm correlation is the procedure of correlating a set of fault events into a single event that represents the root cause of the faults by filtering the redundant events. The event correlation methods can be divided into rule-based reasoning, artificial intelligence (AI) method, codebook approach, model-based reasoning, case-based reasoning, and state transition graphs.

In **rule-based reasoning**, a set of rules are matched to events when they arrive at the correlation engine [2]. The rules can be described by text processing languages in the form of *if condition then conclusion*, where *condition* indicates the defined conditions the received events have to meet, while *conclusion* indicates the specific actions the FMS has to take, including choosing the next rule. The rule-based methods are fast in processing well-defined fault events. But frequent changes in network topologies may lead to frequent updates of many rules [3]. A commercial product based on this method is IMPACT [4,5], which also uses model-based reasoning. Currently, rules have to be created manually by experts with system knowledge [6]. Note that the correlation rules can be extracted from properly formatted system logs [7]. Also, AI method combined with human knowledge is more useful than either technique alone [8].

In **AI methods**, event correlation are based on Bayesian belief networks or dependency graphs by using probabilistic reasoning [9–12]. In [13], the authors apply Bayesian reasoning and propose approximate algorithms for fault localization to avoid the computational complexity.

In [14], a conceptual framework that can both describe causal and temporal correlations is proposed. In [15], the author proposes to construct a belief network for each particular fault, such as link up/down. In [16], the authors propose a proactive method to detect faults based on Bayesian network, by monitor traffic changes. Currently, there is no correlation system of practical size developed based on this type of methods.

In **codebook approach**, all the events are grouped into symptoms and problems to form a correlation matrix, from which a reduced size of the matrix, called codebook, is used to identify a problem among observed symptoms by using coding theory [17,18]. The codebook is created in terms of causality graph, which can be created automatically. In [19], the authors describe an algorithm to optimize the codebook size. The codebook method has the advantages of high speed, and resilient to high symptom loss rate. But dynamic network environment may result in frequent updating of the codebook. The SMARTS InCharge is a commercial product based on this method [18].

In **model-based reasoning**, physical and logical entities are described by relational models, which are described by attributes, relations to other models, and behaviors. Event correlations are performed by searching collective behaviors of all the related models in an entity database [4]. For large-scale networks, it may be difficult to establish the models correctly and maintain it constantly. A commercial product based on this method is SPECTRUM [20]. The NetExpert is another example system of this method, which also uses rule-based methods [21].

In **case-based reasoning**, a knowledge base builds a new case by searching a similar former case in a case library and modifying the case to learn actions for current problem [22]. The learning ability is an advantage for dynamic network environments. But efficient searching may be difficult for large-scale systems. An example system of this method is SpectroRx, which is an add-on application for SPECTRUM [20].

In **state transition graphs**, the relations among different events are described by using state, token, and arc. Change in state is represented by a token's movement along an arc. When a token enters a state, an action will be triggered. For large-scale networks, it is difficult to create such transition graphs. An example system of this method is NerveCenter [20].

It is worth pointing out that there are other products for alarm correlations such as HP OpenView's event-correlation service (ECS) component and Cisco's InfoCenter. A possible solution to alarm correlations in MPLS-enabled backbone networks may need to combine several methods and develop a hybrid one, which is the goal of this paper in correlating the MPLS alarms.

1.4. Contribution of this paper

In this paper, we propose a practical scheme for MPLS fault monitoring and alarm correlation in backbone networks. In order to meet the four requirements on FMS, first, we propose an architecture of three-level hierarchy for FMS: EMS, NMS, and AMS. Fault management is conducted at each level, including alarm generation, collection, and correlation. Second, we present an alarm generation and collection tool for MPLS-enabled networks, i.e., MPLS Connectivity Monitor (CMON). Third, we design an expanded trap format, which carries the component's topology information and enables efficient alarm correlations. Fourth, we propose a hybrid correlation scheme to efficiently correlate the MPLS alarms to other equipment and service alarms. The hybrid scheme consists of an event aggregation scheme, the rule-based reasoning, and the codebook approach. Compared to existing schemes, the hybrid scheme scales well to large-scale problem, quickly adapts to topology changes, and supports both causal and temporal events. Fifth, testing results have been reported from real-world network experiments, which may be used to provide a deployment guideline for various FMS's.

The rest of the paper is organized as follows. The fault management architecture is described in Section 2. The MPLS monitoring and alarms generation are presented in Section 3. The hybrid correlation scheme is presented in Section 4. Experimental studies are provided in Section 5. We make our conclusions and remarks in Section 6.

2. Fault management architecture

There are three levels of management systems used for fault management: EMS, that is developed by network equipment vendors and specializes in managing vendors' equipment; NMS, that aims at managing networks with heterogeneous equipment; and AMS, an alarm managing system that is developed for enterprise network operator's specific OAM needs. The architecture of the proposed fault management system is shown in Fig. 1.

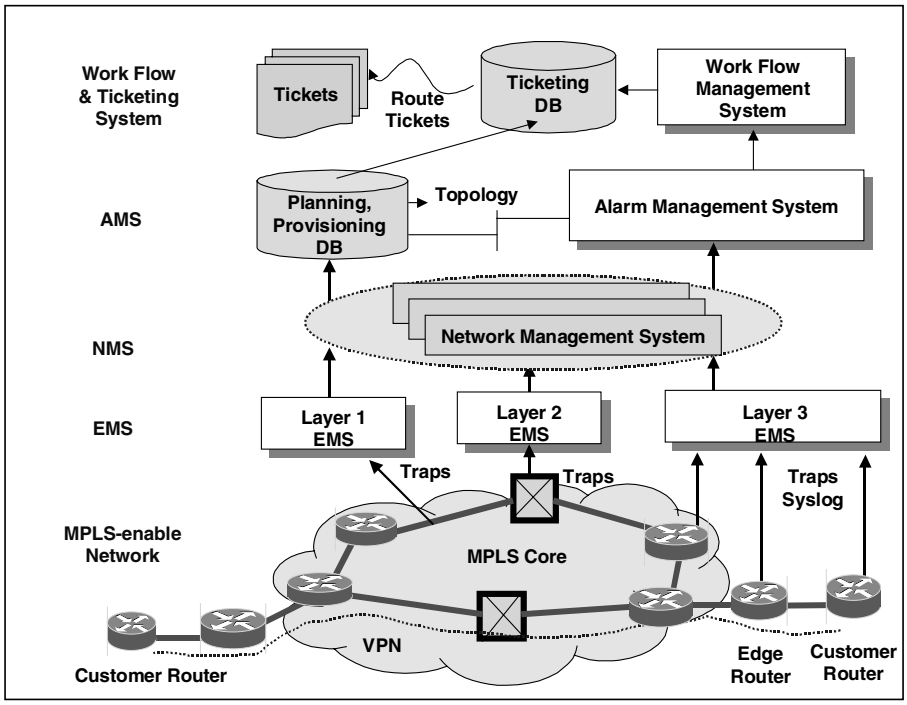

Fig. 1. Architecture of fault management system.

Whenever there is a fault condition in the MPLS core network, each impacted network element, including SONET (Layer 1), ATM switch (Layer 2), or IP router (Layer 3), starts sending traps to its EMS station. Each EMS collects and processes traps independently. Typically, one fault event causes multiple alarms throughout a network and may propagate to multiple interconnected networks. Thus, more than one EMS and NMS may get involved with a single fault event. The symptoms of the fault may behave like a hardware or software failure, or performance degradation. Each involved NMS correlates all the related fault events collected from the network. The partially correlated alarms from each NMS are then fed into the AMS. By using the provisioning database, the AMS correlates all the related fault events collected from different NMS's to identify the root cause. On top of the AMS is a post alarm processing system, i.e., the work-flow and ticketing management system, as shown in the figure.

A key feature of this architecture is that each level performs all the functions of alarm generation, collection and processing. Each level makes correlation decision in a more efficient way by using different scheme and generates high-quality alarms to its upper level.

2.1. Element management system level

Each network element (NE) has a built-in agent that reports management information about the NE's status and operation and takes action under the control of a management system. The protocol commonly used for communication between an NE and its management system is SNMP. The management information is organized as management information bases (MIBs), with MIB objects specified in structure of management information (SMI). For fault management purpose, a NOTIFICATION-TYPE construct provided by SMI is used to specify "Information Request" message or "SNMPv2-Trap". The information request generated by an SNMP manager is sent to the SNMP agent in an NE and used by the agent to query (get) or modify (set) MIB object values associated with the NE. Traps are messages generated by an SNMP agent to notify an SNMP manager of an exceptional situation that has resulted in changes to MIB object values [23]. Some generic trap types, such as cold or warm start by a device, a link going up or down, the loss of a neighbor, or an authentication failure event, have been defined by SNMP standards. Other traps are defined by enterprises.

At this level, an EMS performs simple rule-based correlation on the Layers 1 and 2 alarms it has received. One example is the container-based rule. Another is the connectivity-based rule. These rules will be explained in Section 4.2.

2.2. Network management system level

An NMS processes the alarm events from multiple EMS stations, and also generates alarms, such as synthetic alarms from polling events and syslog. The NMS also supports the SNMP protocol. Many NMS tools also provide some automated assistance for alarm correlation, such as HP OpenView, which has been used to manage heterogeneous networks enabled by MPLS technology.

At this level, the NMS performs alarm correlation by using complicated protocol-based rules on Layer 3 alarms it has received from its domain network, which will be explained in Section 4.2. It also uses the event aggregation scheme, which will be explained in Section 4.1. To monitor the operation of the MPLS-enabled network, the NMS station of each involved domain network has to integrate CMON into its FMS. Therefore, the alarms generated and collected by CMON are fed into the NMS station for alarm correlation, in a way similar to that for the Layer 3 alarms shown in Fig. 1.

2.3. Alarm management system level

An AMS has three functions: receiving alarms, correlating the received alarms, and reporting the correlation results to upper level. Among them, the most important one is to correlate the received alarms and thus identify the fault source.

2.3.1. Alarm receiving

The AMS receives and collects from all the NMS's and monitoring tools. The most common fault data received are the SNMP traps. There are other types of fault data, such as the syslog messages, which can be converted into SNMP traps first and then forwarded to the AMS. Except for the generic traps defined in SNMP standards, the common types of traps received by the AMS can be summarized as follows:

- Environmental traps: e.g., power, fan, temperature traps.

- Resource management traps: e.g., CPU, memory utilization over threshold limits.
- Equipment component traps: e.g., card, line, port traps.
- Traps regarding redundancy: e.g., card and line switchover traps.
- Interface related traps: e.g., SONET interface, POS interface failure traps.
- Traps regarding end-to-end circuit: e.g., PVC traps.
- Protocol traps: e.g., OSPF, BGP, MPLS traps.
- Traps regarding network management connectivity: traps for IP unreachable between an EMS and its managed router/switches.

Note that not all these traps are needed by an AMS for a specific application. For example, for NCS, the goal is to quickly identify the fault sources and then restore the service. Thus, only a subset of NCS related traps are needed. For CCS, the two goals are to quickly identify which customers are impacted and how to restore the service. Thus, the AMS needs both NCS and CCS traps in alarm correlations.

2.3.2. Alarm correlating

Here, the codebook approach is adopted due to its high speed and resilience to symptom loss, which will be explained in Section 4.3. The codebook is built from the network topology database, which is assumed to be available from other network provisioning systems. To meet the scalability requirement, which is the most critical requirement for an AMS, a few effective practices we have deployed can be summarized as follows:

- **Correlate at each level, section, and layer:** Alarm correlation is performed not only at each level of the FMS hierarchy, but also at each section of a network or a layer of a protocol suite. In the end-to-end backbone VPN network, as shown in Fig. 1, the responsibility of network monitoring is either horizontally and or vertically divided, or any kind of combinations. One example is shown in Table 1, in which the network is separated by the three layers, regardless of which section of the network. Another example is shown in Table 2, in which the network is separated by sections, regardless of which layers of the network. In our deployment of the proposed FMS architecture, Layers 1–3 are managed by separated EMS's independently. The NMS that manages different sections also correlates the traps independently. Only those alarms that have not been resolved by the EMS and NMS are fed to the AMS to reduce the overlapped trouble-shooting time.
- **Correlate locally:** Alarm correlation is performed as local as possible, before the fault propagates to a larger area. For example, for a protocol error, we try to identify the fault source to a layer as low as possible. Otherwise, it often takes lots of time to trouble-shoot the fault at upper layers while the fault source is really at a lower layer.
- **Correlate efficiently:** The correlation engine at each level, section, or layer makes correlation decision as efficient as possible. A high-quality alarm generated by one level will greatly reduce the burden on its upper levels. That is why we use an expanded trap format that carries topology information to enable efficient correlations, which will be explained in Section 3.3.

Table 1
Case 1: Network is separated by layers

Routers and layers	CPE	Edge	MPLS core
Layer 1 (SONET)	NMS-L1	NMS-L1	NMS-L1
Layer 2 (ATM)	NMS-L2	NMS-L2	NMS-L2
Layer 3 (IP)	NMS-L3	NMS-L3	NMS-L3

Table 2
Case 2: Network is separated by sections

Routers and layers	CPE	Edge	MPLS core
Layer 1 (SONET)	NMS-1	NMS-2	NMS-3
Layer 2 (ATM)	NMS-1	NMS-2	NMS-3
Layer 3 (IP)	NMS-1	NMS-2	NMS-3

2.3.3. Alarm reporting

The AMS reports fault source to the work-flow management and ticket generation systems. Normally, only correlating and suppressing the excessive traps may not be sufficient. The work-flow management and ticket generation systems may also need to know the impacted network devices or services, in addition to the fault sources. We suggest that minimally the following information should be included in an alarm database:

- Primary Alarm: the alarm that is related to the object of a possible fault source. Normally, this alarm is associated with the object in a lower level of topology or protocol hierarchy.
- Secondary Alarm: the alarm that is related to some objects that are indirectly impacted. These

alarms may not directly help resolving a problem but indicate all the impacted NE's.
- Status of Impacted NE: the condition of the impacted network equipment right after the traps being sent out. A few simple CLI commands are sent out by the FMS to confirm the NE condition to help trouble-shooting.
- Impacted Customers: a list of the impacted service and the subscribed customers.

After received the above information, the workflow management system assigns a trouble-shooting task to the responsible work force and at the same time creates a ticket to track the problem.

3. MPLS monitoring and alarms generation

The tools for MPLS alarm generation are in the categories of OAM mechanisms defined by IETF and ITU-T organizations [24]. Among the various OAM mechanisms proposed for MPLS, both by IETF and ITU-T, our goal is to choose one suitable for the fault management of backbone networks. Our fault management framework for large backbone networks covers three different areas of OAM on MPLS: failure detection, alarm correlation and network monitoring. The first is covered by IETF's MPLS ping/traceroute, which provides a good solution for determining and alerting the affected routers about different LSP and node failures. The second area, which includes alarm correlations among different protocol layers, gives the FMS an important mechanism to identify root causes upon failures. This will be covered in next section. Finally, using MPLS MIBs to monitor the routers of the backbone at the different levels, gives a good network monitoring solution.

3.1. OAM tools for MPLS monitoring

The OAM mechanisms can be divided into two competing categories. One is IP-related mechanism, such as MPLS ping/traceroute, corresponded to those in IP. Another is IP-independent mechanism, such as FEC-CV, e.g., the MPLS-specific OAM packets defined in ITU-T's Y.17xx Recommendations on MPLS, which are used to verify that LSPs maintain connectivity and tells affected routers about failures. The MPLS WG adopts the IP-related mechanism. VRF-aware ping/traceroute, Bidirectional Forwarding Detection (BFD)/VC Connectivity Verification (VCCV), and Label Switch Router (LSR) Self-Test are other methods for failure detection and diagnosis. MPLS SNMP MIBs give operational mechanisms.

3.1.1. MPLS ping/traceroute

As a simple and efficient mechanism, MPLS ping can be used to detect data plane failures in MPLS LSPs. It has a "ping mode" and a "traceroute mode" for testing MPLS LSPs. The ping is used for connectivity checks while the traceroute is used for hop-by-hop fault localization as well as path tracing.

3.1.2. MPLS SNMP MIBs

The IETF Network WG has developed a few drafts that describe managed objects for modeling MPLS. Among them, MPLS-LDP MIB, MPLS-VPN MIB, Label Switch Router MIB, and Traffic Engineering MIB are the MIBs related to our work in fault management and alarm correlation. Another example of an MPLS MIB is FEC-To-NHLFE MIB (FTN MIB). There are MIBs proposed by IETF, for instance, PW-MPLS, PW-ATM, PW-FR MIBs for PWE3 (MPLS & Pseudo Wire Emulation Edge-to-Edge). There are also some enterprise implemented MIBs, such as CISCO-VPN, CISCO-TE, and CISCO-MPLS-SSO, etc.

3.1.3. FEC-CV (MPLS-specific OAM packets)

ITU-T has published the recommendations for user-plane OAM functionality in MPLS networks. The user-plane refers to the set of traffic forwarding components through which user traffic flows. User-plane OAM tools are required to verify that LSPs maintain correct connectivity, and are thus able to deliver customer data to target destinations according to both availability and QoS guarantees, given in SLAs [25]. There are six types of OAM packets proposed: Connectivity Verification (CV), Performance, Forward Defect Indicator (FDI), Backward Defect Indicator (BDI), Loopback Request, and Loopback Response. So far, these are only recommendations.

3.2. MPLS Connectivity Monitor (CMON)

The architecture of CMON is shown in Fig. 2. A service provider's backbone is comprised of the provider (P) routers and PE routers. By periodically generating and sending out short messages (such as MPLS pings) to designated PE routers, CMON allows network operators to quickly assess that a router or connection is up and running. If the ping fails, CMON generates specific SNMP alarms corre-

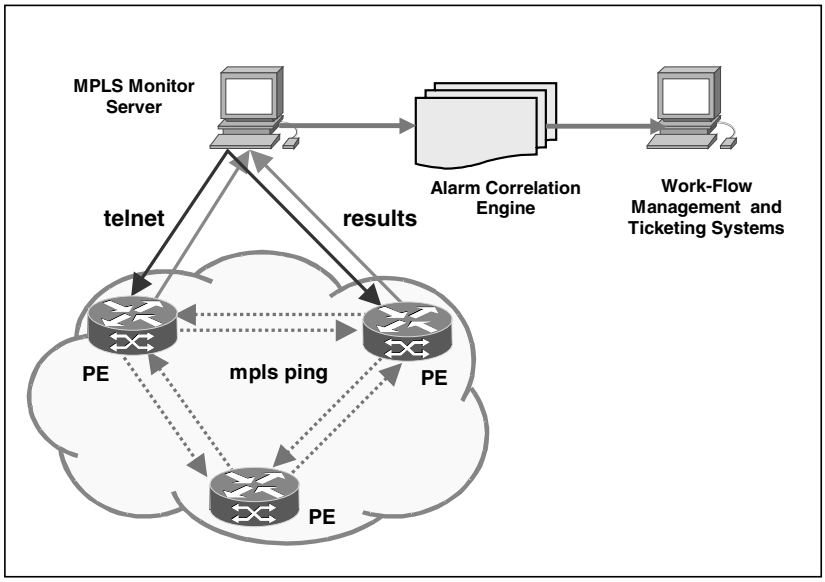

Fig. 2. MPLS connectivity monitor.

sponding to the results received from the network elements. These alarms will be then forwarded to the upstream alarm correlation engine for investigating the root causes of the faults.

The functions of CMON include automatically establishing telnet sessions to PE routers, generating and sending MPLS ping commands to check reachability of the PE routers, analyzing the responses of the above commands to determine the status changes, generating SNMP alarms when the status is changed from normal to failure or from failure to normal, sending alarms to NMS for graphical display and further processing.

The utilities used in CMON, include *ping vrf* and *ping mpls* (LSP ping), and *traceroute*, are provided by switch or router vendors and are now VPN aware and can detect VPN specific faults. There are a variety of these utilities available for different vendor products, such as *ping vrf* for Cisco routers, *ping vrf* and *ping mpls* (LSP ping) for Juniper routers, etc.

3.2.1. VPN Routing and Forwarding (VRF) diagnostic

The monitoring of an MPLS VPN backbone relies on the appropriate OAM tools, not only LSP ping/traceroute, but also the VRF-aware ping/traceroute.

A virtual private network (VPN) is a network in which customer connectivity to multiple sites is deployed on a shared infrastructure with the same administrative policies as a private network. In MPLS, a VPN generally consists of a set of sites that are interconnected by means of an MPLS-enabled backbone network. MPLS VPN provides the ability that the routing information about a particular VPN be present only in those PE routers that be attached to that VPN. A key element in the MPLS VPN technology is the VPN Routing and Forwarding (VRF) table. A VRF is a routing table instance on a PE. By assigning unique VRFs to each customer's VPN, user's traffic separation occurs without tunneling or encryption because it is built directly into the network.

The global routing table and the per-VRF routing table are independent entities. The diagnostic utilities such as ping and traceroute, and telnet—all invoke the service routines that deal with the global IP routing table. A local VRF interface on a PE is not considered a directly connected interface in a traditional sense. To diagnosis an interface on a PE participated in a particular VRF/VPN, the utilities such as ping and traceroute, and telnet need to be VRF-aware, i.e., capable of dealing with a local VRF interface and displaying routes connecting customer sites in a particular VPN. These utilities provided by switch vendors are typically in the form of *ping vrf*, *traceroute vrf*, and *telnet vrf*. As one example, we can issue a standard telnet command from a CE router to connect to a PE router. However, from that

PE, we must issue the VRF-aware commands to connect from the PE to the CE: *telnet vrf vrf-name*, and then utilize *ping vrf-name*, and *traceroute vrf-name* commands in a VRF context similarly.

3.2.2. Alarms generated by CMON

The alarms generated by CMON are summarized in Table 3.

In addition to periodically pinging a VRF or FEC to ensure connectivity, CMON can be also integrated with other utilities such as traceroute. If the ping fails, one can then initiate a traceroute VRF or FEC to determine where the fault lies. One can also periodically traceroute VRFs or FECs to verify that forwarding matches the control plane; however, this places a greater burden on transit LSRs. The current design is to run CMON as an independent server. The fault location function will be separately put on an upstream FMS after correlating all the alarms from CMON, LDP syslog messages, current NMS alarms for MPLS interfaces, and PE-CE link Up/Down alarms from the IF-MIB by together using some other CLI and NMS tools.

The frequency of periodically sending each Ping packet is dependent on the probability of how often a VRF or LSP may fail. The time it takes to detect a failure, plus the time it takes to alert affected routers, may be too long for other applications such as protection switching. A direct way to improve the LSP failure detection time is to increase the frequency of the Ping packets, which may use quite amount of bandwidth.

3.3. Trap format and MPLS alarms

We extend the Internet-Standard approach for network fault management in two aspects. First, the extension includes various event types in a multi-level hierarchy, which describe the fault data obtained from different monitoring modules. Second, the extension adds topology information to the trap messages so that alarm-processing units will have sufficient information to make more efficient decisions.

Table 3
Alarms generated by CMON

ID	Alarm	Severity
1	IpFailure	Critical
2	IpPartialFailure	Major
3	IpFailureClear	Normal
4	VrfFailure	Critical
5	VrfPartialFailure	Major
6	VrfFailureClear	Normal
7	LspFailure	Critical
8	LspPartialFailure	Major
9	LspFailureClear	Normal
10	TelnetFailure	Major
11	TelnetFailureClear	Normal

3.3.1. New trap format

Expanding trap format to contain topology information may speedup correlations at each level of an FMS. In [26], the authors use an expanded format to generate causal information needed for correlation and apply it to a wireless system. We use a similar format to integrate various fault events to fit for our hybrid correlation scheme.

As defined in the standard, the SNMPv2 trap PDU format is shown in Fig. 3. The expanded trap PDU format is also shown in the figure. Here, the topology field is the topology information related to a component failure or protocol error. This topology information can be provided by the individual switch or router, or by a separate topology database. The event type and error code together indicate the reason for the event. The network element and object names identify the source of the event. The reason field provides additional information about the reason for the event in some cases. Each alarm-processing unit in an FMS can also add its parsed result here for further processing. The time field indicates when the event was generated.

3.3.2. Alarm summary

The alarms can be categorized into MPLS alarms, which include MPLS protocol and service alarms, such as LDP, MP-BGP, VPN; network equipment alarms, which include alarms generated by network elements and transmission links, such as switches, routers, and SONET/SDH; synthetic alarms generated by network monitoring or status polling tools, such as NMS (e.g., HP OpenView), CMON, and alarms converted from syslog messages from network devices. These alarms are parsed from the corresponding traps that can be either defined by device MIBs or ad hoc mechanism such as ping.

The MPLS alarms are those generated according to the related MPLS MIBs, which include LDP-MIB, LSR-MIB, VPN-MIB, TE-MIB and MP-BGP-MIB, which give us the possibility to configure and monitor different parameters concerning MPLS

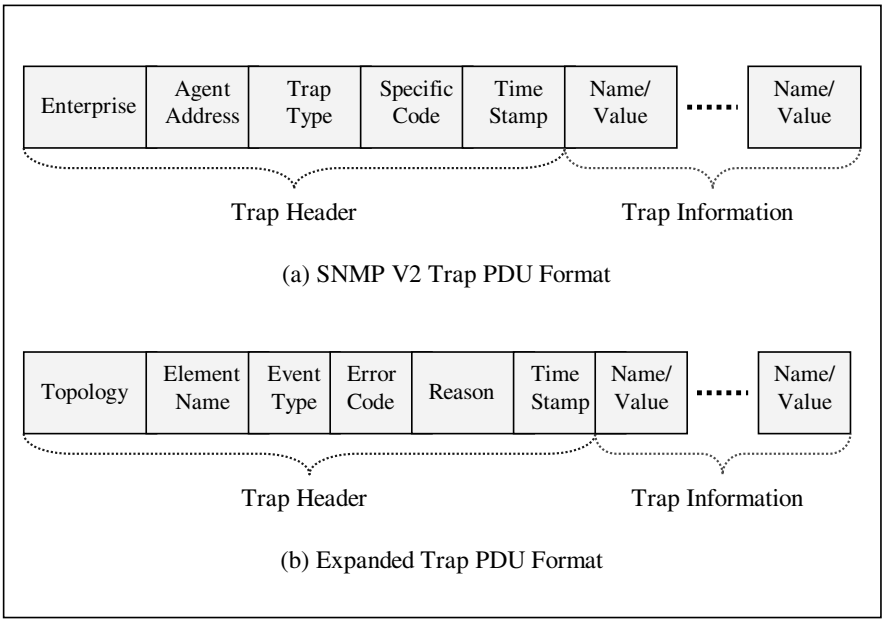

Fig. 3. SNMP trap PDU formats.

Table 4
MPLS alarm summery

ID	Alarm	MIB	Severity
1	mplsLdpSessionUp	LDP-MIB	Normal
2	mplsLdpSessionDown	LDP-MIB	Warning
3	mplsLdpFailedISTE	LDP-MIB	Warning
4	mplsLdpPVLMismatch	LDP-MIB	Warning
5	mplsInSegmentUp	LSR-MIB	Normal
6	mplsInSegmentDown	LSR-MIB	Warning
7	mplsOutSegmentUp	LSR-MIB	Normal
8	mplsOutSegmentDown	LSR-MIB	Warning
9	mplsVrfIfUp	VPN-MIB	Normal
10	mplsVrfIfDown	VPN-MIB	Major
11	mplsNumVrfRouteMidTE	VPN-MIB	Warning
12	mplsNumVrfRouteMaxTE	VPN-MIB	Warning
13	mplsNumVrfSecILTE	VPN-MIB	Warning
14	mplsTunnelUp	TE-MIB	Normal
15	mplsTunnelDown	TE-MIB	Warning
16	mplsTunnelRerouted	TE-MIB	Warning
17	mpbgpSessionUp	MPBGP-MIB	Normal
18	mpbgpSessionDown	MPBGP-MIB	Major

LSRs and the MPLS LSPs. These alarms are summarized in Table 4.

The NE alarms are those generated by network elements. The important are those generic SNMP alarms defined in RFC 1907: coldStart, warmStart, linkDown, linkUp, authenticationFailure, and egpNeighborLoss. The non-generic alarms vary vastly for different equipment. The equipment can be P, PE, and CE routers from different equipment vendors. The alarms related to transmission links are SONET/SDH Section/Line/Path failure and clear.

The synthetic alarms are generated by NMS such as HPOV, which can be set up to automatically poll any of the device MIBs directly. Some of the alarms from interface polling, such as ifUp, ifDown and ifStatusClear, are duplicate to the SONET/SDH interface traps generated by connected network devices, and thus need to be filtered out directly. This type of alarms can be also extracted from parsing a device's syslog messages and sent to a management station for further processing. The important alarms in this category are bgpNeighborUp/Down, bgpPeerClose, etc. These syslog generated BGP traps can be correlated with the bgpPeerLastError (idle/established) traps.

4. MPLS alarms correlation

As we discussed in Section 1.3, the rule-based reasoning and codebook approach are the two most promising schemes for event correlation of large-scale FMS.

The codebook scheme has the advantages of high speed, resilient to high rates of symptom loss if using error-correction coding approach, once the codebook has been generated. But the size of the codebook grows exponentially with the percentage of correlated alarms. Also, the codebook scheme

only supports the correlation of causal events, not temporal events [14]. A new event may result in the calculation of a new codebook, instead of directly adding an entry to the existing correlation matrix.

On the other hand, the rule-based scheme is more efficient than other methods in handling the common events frequently happened in a restricted domain in real-world networks. Another advantage is that it supports the correlations of both causal and temporal events in the same way. But building rules for network-wide events in a large-scale may be complicated and time-consuming.

In this section, we propose a hybrid method that consists of the rule-based methods, an event aggregation method, and the codebook method. For events within a network device or protocol and are managed by an EMS, or the events among different protocol layers but have simple relations, simple rules are developed to correlate the most frequent alarms. Thus, the propagation of alarms are restricted close to the place where they are generated. For specific events within a network domain and are managed by an NMS, specific rules are developed to aggregate the related events. In this way, not only are the inter-network impacts decoupled, but also the size of correlation problem reduced. For events across different networks, the codebook scheme can be applied for the reduced size of problem in a more efficient way.

4.1. Correlation by event aggregation

In general, events can be classified as being primitive or composite. Primitive events are directly observable, while composite events are formed by composing primitive or other composite events [27]. An object is an n-dimensional tuple with components being functions of time. Each component is an attribute associated with the object. An event represents state changes of managed objects in a system. Thus, an event can be represented as a tuple of these attributes. For example, the attributes of an alarm event or trap can be the type and name of the trap, etc. An event may occur multiple times during an observation period. Each occurrence of an event is called an instance of the event. Therefore, an instance can be represented as a tuple of values of the attributes.

Formally, an event can be defined as follows:

$$E := \langle e_E, a_E, t_E, i_E \rangle,$$

where e_E is the event name; a_E is the attribute of the event, which can be specified as a tupe of [*name*, *type*]; t_E is the occurrence time of the event; i_E is the ith instance of the event. A specific value of a, t, and i can be denoted by a_n, t_n, and i_n, where $n = 1, 2, \ldots$

To get the values of a, t, and i, we define the functions

$$G_a(E) = a_E, \quad G_t(E) = t_E, \quad G_i(E) = i_E.$$

Examples of the get functions are the get operations defined in SNMP. Similarly, we can define set functions to set the value of a, t, and i:

$$S_a(E) = a_n, \quad S_t(E) = t_n, \quad S_i(E) = i_n,$$

where $a_n = [name_n, type_n]$ is the attribute value assigned to event E.

Note that composite events can be used to correlate events, with correlation rules defined in the composite events. In [28], event operators, such as *AND*, *OR*, *SEQUENCE*, etc., are proposed to specify the relations among the component events. To precisely express occurrence semantics of events, time constraints are introduced in [27] to specify composite events, in addition to using general *if–then* conditions. We adopt a similar approach for composite alarm events to specify causal and temporal relations. In particular, the attribute, time, and instance values are specified for each instance of an alarm event, which is called aggregated event.

define aggregated event
  $AE := \langle e, a, t, i \rangle$
if condition
  CONDITION is true
then
  $S_a(AE) = a_n$, $S_t(AE) = t_n$, $S_i(AE) = i_n$.

In the above definition, *CONDITION* can be specified by using the event operators such as *AND*, *OR*, *SEQUENCE*, etc., on the attribute, time, and instance values of the component events.

As an example, a cross-network scenario is shown in Fig. 4, in which two switches in different locations and managed by different NMS stations are connected by two PVCs. If one side of the trunk or a port of the trunk is faulty, the ports in the far-end switch will be also faulty. Each NMS station locally correlates the alarms it received. The AMS station globally correlates the alarms from the two NMS stations. A sample causality graph is shown

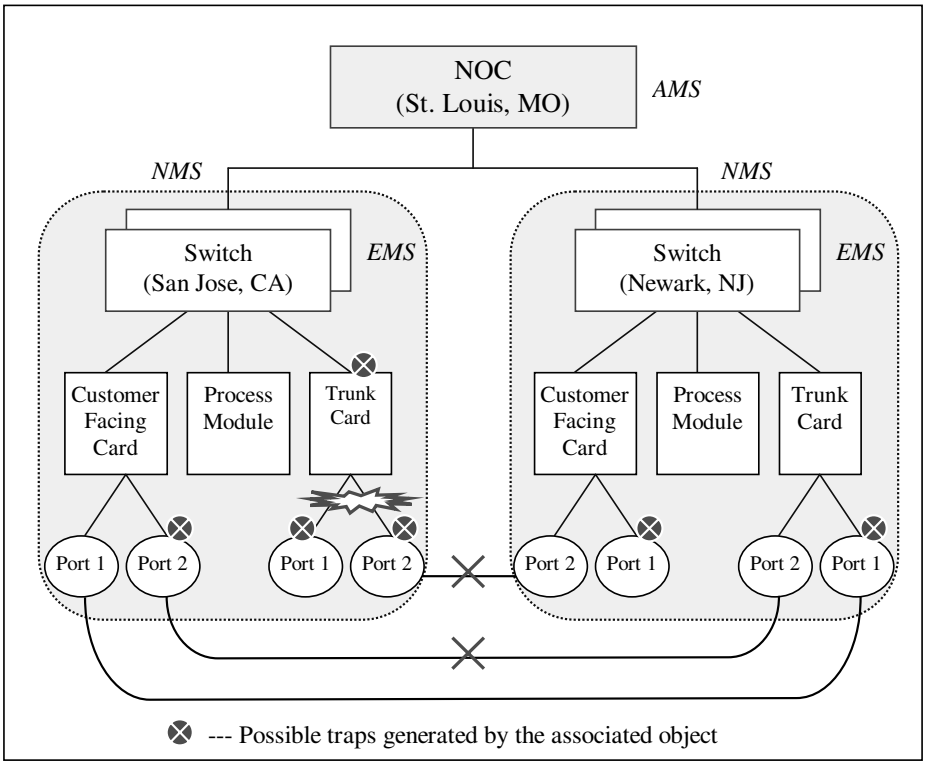

Fig. 4. A simple correlation model.

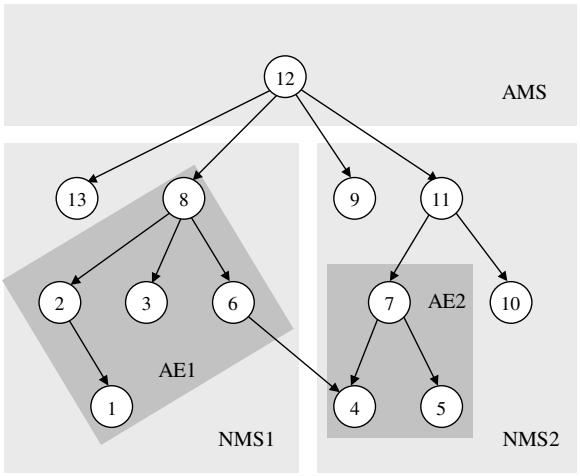

Fig. 5. A example of causality graph.

in Fig. 5. To correctly identify the root cause, the AMS needs to collect all the alarms from the two NMS stations. By using event aggregation, we can define two aggregated events, i.e., $AE1$ and $AE2$, to represent events $\{1,2,3,6,8\}$ and $\{4,5,7\}$, respectively. For example, we assume that $\{1,2,3,6\}$ are symptoms and $\{8\}$ is the problem, we can define:

define aggregated event
$AE1 := \langle e, a, t, i \rangle$
if condition
$\{e_4 \mid e_1\ AND\ e_3\}$ *is true*
then
$S_a(AE1) = a_8,\ S_t(AE1) = t_8,\ S_i(AE1) = i_8.$

In the above definition, the condition is that if both events e_1 and e_3 are observed then define $AE1$, which will be treated as e_8 on the condition that e_4 is observed. In this way, the AMS only needs to collect the aggregated events, $AE1$, from one NMS and the symptom e_4 from another NMS, instead of all the events from the two NMS stations. Thus, the correlation in AMS will be more efficient based on a reduced size of correlation problem.

4.2. Rule-based correlation

In the alarm correlation hierarchy shown in Fig. 1, the correlation mechanisms implemented in EMS and NMS Levels try to filter out most redundant alarms locally, instead of sending them to AMS. The correlation in EMS level deals with a

single network element across different slots and ports, such as the different modules and ports in a router. The correlation in NMS level deals with a single network platform across different network elements, such as the routers in an IP network. For these correlations, a simple choice is to use rule-based correlation scheme.

In [4,5], the general process of event correlation is summarized into six operations among the observed event alarms (symptoms) and possible root causes (problems): compression, suppression, count, Boolean patterns, generalization, and specialization. In this paper, we develop rules in terms of specific device or protocol characteristics. The correlation rules can be categorized as: topology-based correlation, connectivity-based correlation, container-based correlation, protocol-based correlation, cross-platform correlation (SONET), bouncing interface correlation, and cross-event correlation. It has been noticed that different correlation level may have to use different correlation rules. Also, there may exist one or more rules applicable to one fault event and each may result in different reduction of redundant alarms.

We illustrate some of these important rules by examples. Depending on specific applications, a root cause is reported as a primary alert and generate a trouble ticket to track this failure. At the same time, the symptoms are grouped as secondary traps for information only. The alarm correlation engine may just suppress all the symptoms, depending on the implemented reporting policy.

Example 1. Container-based Rule

Primary Alert:
Router PE1 is DOWN!

Secondary Alerts:
Link down from source P1 to destination PE1!
VRF Down at vrf1 of PE1!
MPLS VPN connectivity failed from CE1 to PE1!

Rule: If a router is down, then report the Router Down alarm as a primary and group all the resulted alarms from the router as secondary.

Example 2. Topology-based Rule

Configuration:
P1: 1/36/1 — PE2: 1/36/1
P1: 1/36/2 — PE3: 1/36/1

Primary Alert:
Slot 1/36 at P1 was DOWN!

Secondary Alerts:
Link Down from PE2 to P1!
Link Down from PE3 to P1!
MPLS LDP session from P1 to PE2 is DOWN!
MPLS LDP session from P1 to PE3 is DOWN!

Rule: If one slot was down, then group all the alarms from the far-ends of links that have near-ends connected to this slot. Report the slot down as primary alarm and other topology related alarms as secondary alarms.

Example 3. Protocol-based Rule

Primary Alert:
Link POS 1/38/2 at P1 has SONET AIS Failure!

Secondary Alerts:
Link POS 1/38/1 at P1 has SONET LOF Failure!
MPLS LDP Session was down at switch P1!
MPLS LDP InitSessionFailed Exceeded Threshold at P1!

Rule: If the lower protocol layer is down, then group all the resulted higher protocol layers' alarms. Report the lower protocol layer alarm as primary and the higher protocol layer alarms as secondary.

Example 4. Connectivity-based Rule

Primary Alert:
Router PE1 was DOWN!

Secondary Alerts:
IP Connectivity Failure from P1 to PE1!
Telnet Failure from MPLS Connectivity Monitor Server to PE1!

Rule: If connectivity fails, then group the alarms resulted from the loss of this connectivity and report them as secondary. Report the underlying connectivity failure as primary.

Example 5. Bouncing Interface Rule

Primary Alert:
VRF Bouncing at vrf1 of PE1!

Secondary Alerts:
VRF Down at vrf1 of PE1!
VRF Up at vrf1 of PE1!

Rule: If more than ten interface Up/Down event pairs are received within ten minutes, then report a bouncing interface alarm as primary, and group all the interface Up/Down events as secondary.

There are of course many other correlation rules. In our testing network, more than 200 rules have been developed to correlate over 90% of the common alarms.

4.3. Codebook approach

The correlation in AMS level is to handle the enterprise network alarms across different network platforms, such as IP, ATM, Frame Relay, and DSL, which comprise the MPLS-enabled backbone network. The correlated alarms will be tracked by trouble-shooting tickets, which will be handled by work force. In practice, resolving the tickets is a costly process. Therefore, the correlation mechanism implemented in AMS level must be able to handle a large volume of all types of alarms generated by different network elements in different networks, as complete as possible.

After two levels of correlations, which have filtered out most of the alarms locally within each individual EMS and NMS, the codebook scheme is adopted in AMS level. Its advantages of high speed and resilient to high rates of symptom loss can be fully utilized for a codebook of reduced size. Its disadvantage of not supporting the correlation of temporal events has been compensated by using events aggregation. The chance of recomputing codebook due to new events has been reduced by using rule-based scheme in lower levels of correlation.

In the codebook scheme, events are divided into symptoms and problems. Symptoms are observable events and problems non-observable events. By converting causality graph into correlation graph, which is a bipartite graph to describe the relations among the problems and symptoms, a correlation matrix can be formed. The correlation matrix can be reduced into a codebook, which can be further optimized based on the requirement of error-correction capability for the codes. The methods of building causality graph, converting to correlation graph, and selecting optimal codebook, can be found in [29] and are commercially available.

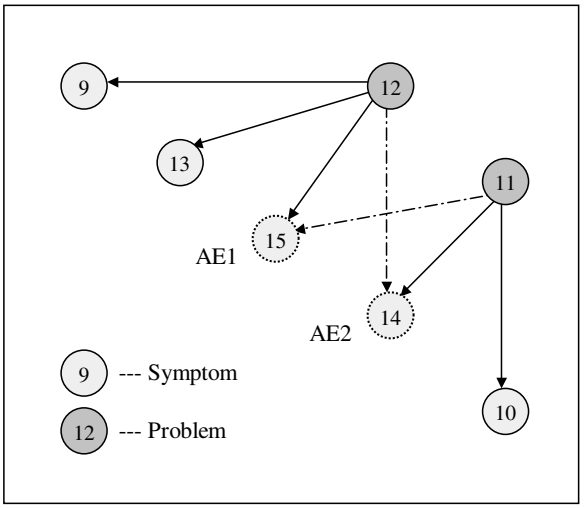

Fig. 6. A example of correlation graph.

As an example, the causality graph shown in Fig. 5 can be converted into the correlation graph shown in Fig. 6. By defining two aggregated events, labeled as $\{14, 15\}$ in the graph, and choosing $\{11, 12\}$ as the problems, others as symptoms, the correlation matrix becomes a 2×5 matrix, as compared to the original 3×7 matrix [14]. For a minimum Hamming distance of 2, the size of the codebook is reduced from 3×4 to 2×2. Clearly, events aggregation results in a smaller codebook and more effective coding/decoding process since there are fewer comparisons for each event.

It is worth noting that the scalability and changing topology issues can not be resolved by directly applying the existing schemes. The proposed hybrid scheme under the hierarchical architecture, in which different correlation schemes are used in different phases of alarm processing and correlations, is such a promising one that may resolve the issues.

5. Experimental studies

The testbed network is shown in Fig. 7, in which there are seven routers deployed over a major backbone network. These routers are labeled as Router A, B, C, D, E, F, and G. When there is a fault event occurred in the network, each impacted network device will send out traps independently, based on the condition it knows from its directly associated neighbors. The AMS takes a network-wide view and correlates all the related traps.

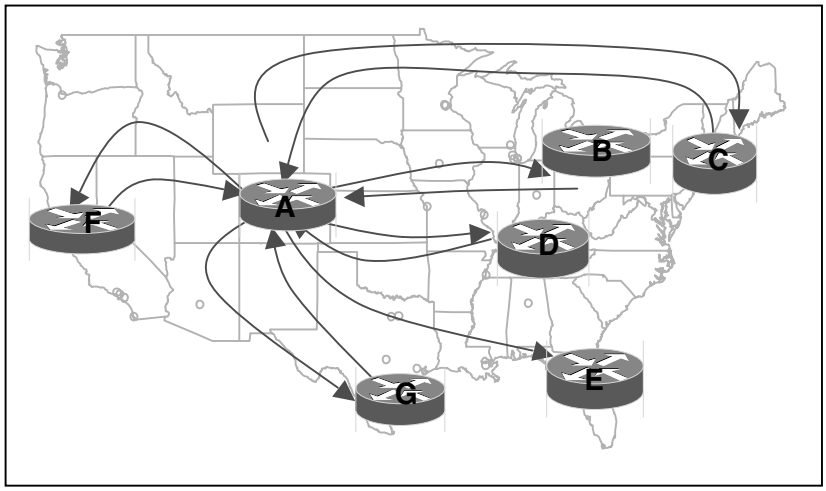

Fig. 7. A testbed of MPLS-enabled Network.

5.1. Case 1: MPLS alarm generation

In this case, a SONET line status change has been reported. In particular, a SONET line Alarm Indication Signal (AIS) Failure is reported on link POS 1/38/2. The purpose is to test if CMON generates MPLS VPN alarms and LDP alarms based on the LDP MIB. The alarms generated are shown in details in Fig. 8, in which the alarms are put in a readable format, with fields of alarm number, date, time, EMS that handles the alarm, router IP address, type of message, trap name and severity, etc. Compared to the alarms listed in Tables 3 and 4, it can be seen that the MPLS alarms are correctly generated in time.

5.2. Case 2: Alarm correlation for OSPF neighbor status changes

In this case, the AMS has received 7 traps regarding the OSPF neighbor status change, from the 7 routers as shown in Fig. 7. After correlation, only one trouble ticket is generated with 1 primary alarm and 11 secondary alarms. The trouble ticket is shown in Fig. 9.

5.3. Case 3: Alarm correlation for MPLS-enabled network

In this case, a week period of alarms received by the AMS are collected for study, in which most of them are NE alarms, such as linkUp/Down, protocol alarms, such as BGP Session Failure, security alarms, such as Authentication Failure, and the MPLS alarms, as summarized in Tables 3 and 4. In this testing, three correlation schemes are applied: the basic alarm correlation at EMS level, such as aging, counting, container-based rules, connectivity-based rules, and bouncing interface rules, etc.; the rule-based correlation scheme at NMS level, such as topology- and protocol-based rules, for events within the same network, and protocol layer; and finally the codebook scheme at AMS level, by using a small codebook based on the aggregated events that are defined in EMS and NMS levels. The total number of traps initially generated by the network is 30,711. After filtering unnecessary traps, such as minor, informational, and certain clear traps, there are 1325 critical traps that need to be correlated.

The alarm correlation results are summarized as follows:

- Basic Correlation Scheme: among the 1325 total critical alarms, 731 alarms are identified as primary alarms, 594 of them are identified as secondary alarms. The reduction rate of the critical alarms is 44.83%.
- Rule-based Correlation Scheme: among the 731 critical alarms reported by the basic correlation scheme, 643 of them are further identified as primary alarms, 88 of them are identified as secondary alarms. The further reduction rate of the critical alarms is 12.04%.
- Codebook Correlation Scheme: among the 643 critical alarms reported by the rule-based correla-

Sample MPLS Alarms

```
1140 01/07/2003 00:13:40 EMS500_013: Incoming Alarm:
routera.net1.abc.net: Traps|mplsLdpSessionDown: 1|

bd00 01/07/2003 00:13:45 EMS500_013: Incoming Alarm:
routera.net1.abc.net: Interfaces|Avici Other Error:
routerb.net2.abc.net: pos1/33/2 is reporting SonetSectionEventStatus
32; Sonet Loss Of Signal (LOS) Failure on link POS 1/33/2. :: ifindex
= 2162690|

bb80 01/07/2003 00:14:09 EMS500_013: Incoming Alarm:
routera.net1.abc.net: Interfaces|Avici Other Error:
routera.net1.abc.net: pos1/38/2 is reporting SonetSectionEventStatus
32; Sonet Loss Of Signal (LOS) Failure on link POS 1/38/2. :: ifindex
= 2490370|

aec0 01/07/2003 00:15:39 EMS500_013: Incoming Alarm:
routera.net1.abc.net: Interfaces|Avici Other Error:
routera.net1.abc.net: pos1/38/2 is reporting SonetLineEventStatus 32;
Sonet Line Alarm Indication Signal (AIS) Failure on link POS 1/38/2.|

9ac0 01/07/2003 00:15:47 EMS500_013: Incoming Alarm:
routera.net1.abc.net: Traps |
mplsLdpFailedInitSessionThresholdExceeded: 8|

cd30 01/07/2003 00:16:04 EMS500_013: Incoming Alarm:
routera.net.abc.net: MPLS Monitor Traps|VrfFailure on connectivity
from routera.net1.abc.net:pos1/38/2 to routerb.net2.abc.net: pos1/33/2
of VRF 143d|
```

Fig. 8. Alarms generated for MPLS LDP and VPN.

Trouble Ticket

Primary Alarm : Router A :Router B 03/27/2003 01:20:51 - OSPF Neighbor Status Change

Secondary Alarms:

Router E : Router A	03/27/2003 01:20:15 OSPF Neighbor Status Change
Router C : Router A	03/27/2003 01:20:21 OSPF Neighbor Status Change
Router B : Router A	03/27/2003 01:20:47 OSPF Neighbor Status Change
Router F : Router A	03/27/2003 01:20:47 OSPF Neighbor Status Change
Router D : Router A	03/27/2003 01:20:49 OSPF Neighbor Status Change
Router G : Router A	03/27/2003 01:20:49 OSPF Neighbor Status Change
Router A : Router B	03/27/2003 01:20:51 OSPF Neighbor Status Change
Router A : Router C	03/27/2003 01:20:59 OSPF Neighbor Status Change
Router A : Router F	03/27/2003 01:21:07 OSPF Neighbor Status Change
Router A : Router G	03/27/2003 01:21:09 OSPF Neighbor Status Change
Router A : Router D	03/27/2003 01:21:09 OSPF Neighbor Status Change

Fig. 9. Trouble ticket for OSPF neighbor status change.

tion scheme, 359 of them are further identified as primary alarms, 284 of them are identified as secondary alarms. The further reduction rate of the critical alarms is 44.17%.

These results are plotted in Fig. 10. Overall, among the 30,711 alarms, only 359 alarms need to have tickets created and resolved by work force. The total alarm reduction ratio is 98.83%, which means only 1.17% of the alarms have to be resolved by the work force. Clearly, the testing results have demonstrated the effectiveness of the proposed FMS architecture and correlation scheme.

5.4. Comparison of alarm correlation schemes

In this section, we compare the performance of the hybrid scheme to those of the other correlation

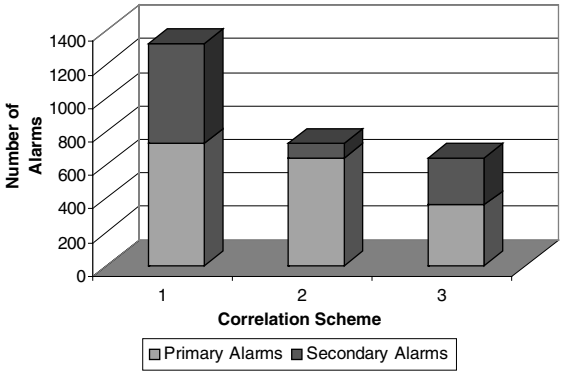

Fig. 10. Reduction of MPLS alarms.

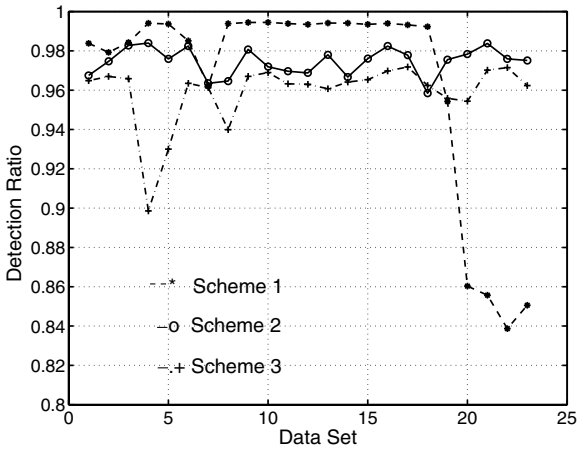

Fig. 11. Alarm detection ration.

schemes. There are 23 data sets, each was collected over a day period when there were alarm storms due to some special network-wide events. After filtering most unnecessary traps, and applying the basic correlation scheme at EMS level, the three different schemes are applied to the collected data off-line, respectively:

- Scheme 1: At both AMS and NMS levels, the codebook scheme has been applied.
- Scheme 2: At AMS and NMS levels, the codebook and rule-based schemes are applied respectively, i.e., the hybrid scheme.
- Scheme 3: At both AMS and NMS levels, the rule-based scheme has been applied.

Let N_{tot} represent the total number of alarms generated by all the network devices in the network. The number of alarms that are output by the AMS is denoted by N_{ams} for manual processing by the work force, among which N_{act} of them are proven to be actual network problems, and N_{spu} of them are spurious alarms that are generated by the alarm correlation systems. Thus, we can define:

$$D_{\text{r}} = \frac{N_{\text{tot}} - N_{\text{act}}}{N_{\text{tot}}} \tag{1}$$

and

$$F_{\text{p}} = \frac{N_{\text{spu}}}{N_{\text{tot}} - N_{\text{act}} + N_{\text{spu}}}, \tag{2}$$

where $N_{\text{spu}} = N_{\text{ams}} - N_{\text{act}}$; D_{r} and F_{p} are called the alarm detection rate and false positive rate, respectively.

The detection rates of the three schemes are plotted in Fig. 11. It can be seen that Scheme 1, i.e., pure codebook method, has a higher detection rate when the network topology is relatively stable, for the case of static codebook without error correction capability. When topology suddenly changes, such as the last four data sets, in which there are a few lost alarms due to some unreachable routers. For these data sets, the rule-based related, such as Scheme 3 has higher detection rate. It can be also seen that Scheme 2, i.e., the hybrid scheme, has a relatively stable detection rate, which is in between the detection rates of Schemes 1 and 3.

The false positive rates for the three schemes are shown in Fig. 12. We can see that Scheme 1 has a relatively stable false positive rate, while Scheme 3 has a highly varying false positive rate. Clearly, Scheme 2 is in between the two schemes by false positive rate.

The real advantage of Scheme 2 is in the correlation time, as shown in Fig. 13. It can be seen that

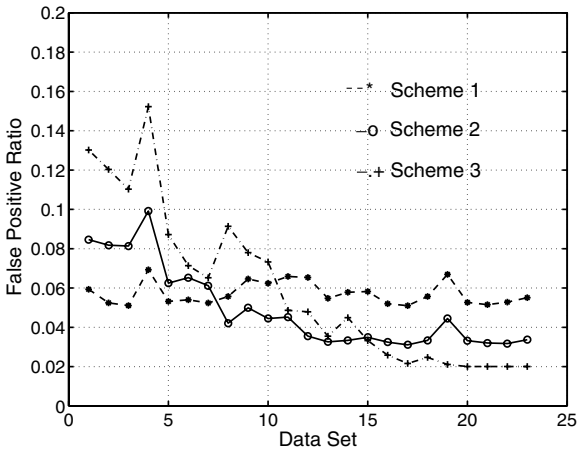

Fig. 12. False positive alarm ratio.

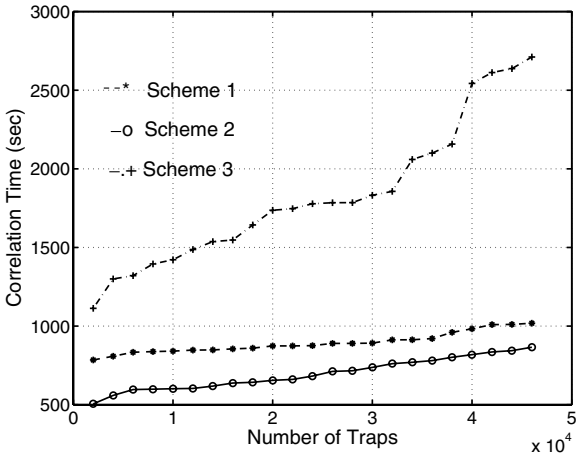

Fig. 13. Correlation processing time.

Scheme 3 is more time consuming than the others. It increases quickly as the number of traps becomes large. On the other hand, Scheme 1 has a relatively stable correlation time. Here, the correlation time only includes the decoding time and does not include the codebook generation time. Clearly, Scheme 2 may have an even smaller correlation time, as compared to Scheme 1. This is due to a much small size of the codebook. Also, the pattern match part conducted by the rule-based scheme is fast when the search domain is limited to within an NMS.

It is estimated that the average correlation time per data set is between 347 and 2869 s, by using a Dell PowerEdge 1800 machine, with dual Xeon CPU of 2.8 GHz, ram of 2 GMB. It is possible to apply the hybrid scheme in real-time alarm processing. In summary, the hybrid scheme outperforms either rule-based or codebook scheme alone, in detection rate, false positive rate, and correlation time.

6. Conclusions and remarks

MPLS technology will be the future platform for sending IP packets through backbone networks. In this paper, we have proposed a hierarchical architecture and hybrid correlation scheme for fault management of MPLS-enabled backbone networks.

Within the Internet-Standard SNMP framework, the use of hierarchical architecture for alarm generation, collection, and correlation, based on MPLS MIBs, is recommended as fault management approach for MPLS-enabled networks. The hybrid scheme gives the FMS a critical capability that will be needed in developing service automation processes. Compared to existing schemes, the hybrid scheme scales well to large-scale problem, quickly adapts to topology changes, and supports both causal and temporal events.

In order to further reduce network operation cost and accelerate new service deployment, FMS must be fully end-to-end automated. For this reason, our future work would be developing methods for automatic rules discovery, more complete MPLS alarm correlation rules, under all kinds of fault events, including the proactive performance monitoring alarms. Also, further research has to find proactive monitoring mechanisms to enhance the availability and reliability of the backbone networks by integrating MPLS VPN and QoS mechanisms.

Acknowledgements

The authors would like to thank the editors and anonymous reviewers for their constructional comments and suggestions.

References

[1] U. Black, Network Management Standards: SNMP, CMIP, TMN, MIBs, and Object Libraries, second ed., McGraw-Hill, New York, 1994.
[2] Y.A. Nygate, Event Correlation using Rule and Object Based Techniques, in: Integrated Network Management IV, Chapman and Hall, London, 1995, pp. 278–289.
[3] A. Hanemann, M. Sailer, D. Schmitz, Assured service quality by improved fault management: service-oriented event correlation, in: ICSOC'04, November 15–19, 2004, New York, USA. Available from: <http://www.mnm-team.org/pub/Publikationen/hss04a/PDF-Version/hss04a.pdf>.
[4] G. Jakobson, M. Weissman, Alarm correlation, IEEE Network Magazine 6 (7) (1993) 52–59.
[5] G. Jakobson, M. Weissman, Real-time telecommunication network management: extending event correlation with temporal constraints, in: Proceedings of the Fourth IEEE/IFIP International Symposium on Integrated Network Management, May 1995, pp. 290–301.
[6] M. Klemettinen, H. Mannila, H. Toivonent, Rule discovery in telecommunication alarm data, Journal of Network and Systems Management 7 (4) (1999) 395–423.
[7] T. Oates, D. Jensen, P.R. Cohen, Automatically acquiring rules for event correlation from event logs, Technical Report, 97-14, Computer Science Department of Umass, 1997.
[8] R. Sterritt, Discovering rules for fault management, in: 8th Annual IEEE International Conference on the Engineering of Computer Based Systems (ECBS), April 2001.
[9] I. Katzela, M. Schwartz, Schemes for fault identification in communications networks, IEEE/ACM Transactions on Networking 3 (6) (1995) 753–764.

[10] B. Gruschke, Integrated event management: event correlation using dependency graphs, in: Proceedings of the 9th IFIP/IEEE International Workshop on Distributed Systems: Operations and Management (DSOM'98), IEEE/IFIP, October 1998.
[11] M. Gupta, A. Neogi, M. Agarwal, G. Kar, Discovering dynamic dependencies in enterprise environments for problem determination, in: Proceedings of the 14th IFIP/IEEE Workshop on Distributed Sytems: Operations and Management (DSOM'03), IFIP/IEEE, October 2003.
[12] D.W. Guerer, I. Khan, R. Ogler, R. Keffer, An Artificial Intelligence Approach to Network Fault Management, SRI International, 1996.
[13] M. Steinder, S. Sethi, Probabilistic fault localization in communication systems using belief networks, IEEE/ACM Transactions on Networking 12 (5) (2004) 809–822.
[14] M. Hasan, B. Sugla, R. Viswanathan, A conceptual framework for network management event correlation and filtering systems, in: Integrated Network Management VI, IEEE Press, New York, 1999.
[15] J. Huard, Probabilistic reasoning for fault management on XUNET, Technical Report, AT&T Bell Labs, 1994.
[16] C.S. Hood, C. Ji, Proactive network-fault detection, IEEE Transaction on Reliability 46 (3) (1997).
[17] S. Kliger, S.A. Yemini, Y. Yemini, D. Ohsie, S. Stolfo, A coding approach to event correlation, Integrated Network Management (IM) (1995) 266–277.
[18] S.A. Yemini, S. Kliger, E. Mozes, Y. Yemini, D. Ohsie, High speed and robust event correlation, IEEE Communications Magazine 34 (5) (1996).
[19] M. Gupta, M. Subramanian, Preprocessor algorithm for network management codebook, in: Proceedings of the Workshop on Intrusion Detection and Network Monitoring, USENIX, Santa Clara, CA, 1999.
[20] L. Lewis, Event Correlation in SPECTRUM and other commercial products, Technical Note, ctron-lml-99-05, 1999.
[21] Agilent Technologies, NetExpert, http://www.agilent.com/comms/OSS.
[22] L. Lewis, A case-based reasoning approach to the management of faults in communications networks, in: Proceedings of IEEE INFOCOM'93, IEEE Press, New York, 1993.
[23] W. Stallings, Snmp, Snmpv2, Snmpv3, and Rmon 1 and 2, Addison Wesley, Reading, MA, 1999.
[24] T.D. Nadeau, MPLS network management: MIBs, tools and techniques from an implementer's perspective, in: MPLS Japan 2003, Tokyo, Japan, October 2003.
[25] A. Habib, S. Fahmy, S.R. Avasarala, V. Prabhakar, B. Bhargava, On detecting service violations and bandwidth theft in QoS network domains, Computer Communications 26 (8) (2003) 861–871.
[26] M. Albaghdadi, B. Briley, M. Evens, R. Sukkar, M. Petiwala, M. Hamlen, A framework for event correlation in communication systems, in: E.S. Al-Shaer, G. Pacifici (Eds.), MMNS 2001, LNCS 2216, Springer-Verlag, Berlin, Heidelberg, 2001, pp. 271–284.
[27] G. Liu, A. Mok, E. Yang, Composite events for network event correlation, in: Integrated Network Management VI, IEEE Press, New York, 1999.
[28] M. Hasan, The management of data, events, and information for presentation for network management, Ph.D. Thesis, Dept. of CS, University of Waterloo, Canada, 1995.
[29] A. Mayer, S. Kliger, D. Ohsie, S. Yemini, Event modeling with the MODEL language, in: Integrated Network Management V, Chapman and Hall, London, 1997.

Ming Yu received his Doctor of Engineering from Tsinghua University, Beijing, in 1994, and Ph.D. from Rutgers University, New Brunswick, NJ, in 2002, all in Electrical and Computer Engineering. He joined the Operation Technology Center, AT&T, Middletown, NJ, in July 1997 as a Senior Technical Staff Member. Since 1999, he was with the Dept. of ATM Network Service, AT&T Labs. From December 2002, he worked for the Dept. of IP/Data NMS Engineering, AT&T Labs. As of August 2003, he joined the Dept. of Electrical and Computer Engineering, State University of New York, Binghamton, NY, as an assistant professor. His research interests are in the areas of routing protocols, traffic engineering, and fault management of networks. He was awarded the IEEE Millennium Medal on May 2000.

Wenjui Li received his Ph.D. in Operation Research from State University of New York, Stony Brook, in 1985. He is a Senior Technical Specialist of AT&T Labs, Middletown, NJ 07748. He has been working on designing tools for SONET Ring planning, design, VLSI placement, and routing design. His current responsibility includes providing Layer 2/3/MPLS network management system requirements for the AT&T edge and backbone network. His research interests include root cause analysis and design algorithms to predict network failures.

Li-jin W. Chung has a B.S. degree from Chung-Yung University, Taiwan, an M.S. degree from State University of New York, Stony Brook, and a Ph.D. degree from North Carolina State University, all in mathematics. She is a technical manager of the AT&T Network Service Assurance Department at AT&T Labs, Middletown, NJ. She is responsible for systems engineering of performance management operations support systems including performance management support of MPLS VPN and VoIP services. She joined AT&T in 1976 and holds four US patents.