Using Policy-based MPLS Management Architecture to Improve QoS on IP Network

Ruey-Shun Chen[1,*], Yung-Shun Tsai[2], K.C. Yeh[2], and H.Y. Chen[2]
[1] Department of Information Management, China University of Technology,
530 Sec. 3 Chung Shan Road, Hukou, Hsinchu, Taiwan
[2] Institute of Information Management, National Chiao Tung University,
1001 Ta Hsueh Road, Hsinchu, Taiwan
[*] E-mail: rschen@cc.nctu.edu.tw

Abstract: - Multi-Protocol Label Switching (MPLS) is in the process of standardization by the Internet Engineering Task Force (IETF). It is regarded as a technology for traffic engineering and QoS in IP-networks. We proposed an IETF Policy-based Network Management Framework and policies with MPLS specific classes. It uses a three-level policy architecture, which includes managing on device, network, and service level using policies for supporting Inter-serve and Diff-serve based end-to-end QoS in the Internet. A prototyping of policy-based management system for MPLS Traffic Engineering is operating on MPLS network elements. Several experiments illustrate the efficiency and feasibility in this architecture. The results show it can reduce the time of the setup of MPLS traffic engineering tunnel over hops and MPLS traffic engineering tunnel deletion. The proposed integrated policy based management architecture will allow network service providers to offer both quantitative and qualitative services while optimizing the use of underlying network resources.

Key-Words: - Multiple Protocol Label Switching, Traffic Engineering, Quality of Service, Policy-based Management

1 Introduction

The Internet Engineering Task Force (IETF) has proposed a number of QoS models and supporting technologies, including the integrated services (IntServ) and differentiated services (DiffServ) frameworks [1]. The latter has been conceived to provide QoS in a scalable fashion. Instead of maintaining per-flow soft state at each router, packets are classified, marked, and policed at the edge of a DiffServ domain. In order to achieve QoS guarantees, control plane mechanisms have been used to reserve resources on demand, but management plane mechanisms are also necessary to plan and provision the network, and manage requirements for service subscription according to available resources [2]. QoS frameworks such as IntServ and DiffServ have so far concentrated in control plane mechanisms for providing QoS. However, it would not seem possible to provide QoS without the network and service management support, which is an integral part of QoS-based telecommunications networks. Considering in particular the DiffServ architecture, a key issue is end-to-end QoS delivery. The Diff-Serv architecture suggests only mechanisms for relative packet forwarding treatment to aggregate flows, traffic management, and conditioning; by no means does it suggest any architecture for end-to-end QoS delivery. In order to provide end-to-end quantitative QoS guarantees, DiffServ mechanisms should be augmented with intelligent traffic engineering functions.

Multi-Protocol Label Switching (MPLS) is a new technology to be standardized by the IETF. The technology enables the setup of Label Switched Paths (LSPs) through an IP network. Initially, the idea of IP label switching was to speed up the packet forwarding in routers via simple table lookups instead of longest-matching prefix algorithms [3].

In this paper, we propose enhancing the IETF Policy Framework in two directions. First, we incorporate the management of Multi-Protocol Label Switching (MPLS) networks into the framework. MPLS is currently seen as a technology to influence the routing of IP networks in order to engineer the traffic with appropriate tools. QoS services are more easily and more flexible deployed in an IP-based network, because MPLS allows a network manager to pin down a route for an aggregate of flows. However, MPLS per se does not have QoS features nor mechanisms, but MPLS together with Differentiated Services (DiffServ) is the favored approach by the IETF [6] for providing

IP QoS. The second enhancement of the policy framework is dealing with network-level and service-level management in IP networks.

Using MPLS networks, the notion of a Label Switched Path (LSP) brings network-level concepts into the framework which has not been dealt with in the device-level policy framework. Furthermore, using traffic engineered network, new kinds of IP services are possible. E.g., a service guaranteeing low packet loss probability can be offered using MPLS as mechanism to traffic engineer the IP network in a way that traffic is routed around hot spots. However, quantifying the service quality is very difficult. Additionally, having mechanisms for guaranteed services in place, advanced IP services need to be specified, configured, and controlled.

The rest of this paper is organized as follows: Section 2 reviews MPLS technique. Section 3 describes the IETF Policy Framework. The framework of the proposed network model including different resource classes and paths is outlined in Section 4. Section 5 describes the actual implementation and performance evaluation of the proposed system. Finally, Section 6 presents conclusions.

2 MPLS Technology

2.1 MPLS Label Stacking

Figure 1 illustrated the MPLS label stack. When a label is added to a packet, this means that at minimum a 4 byte "shim" has been added to the packet. This shim is added between the layer 3 header and layer 2 headers. Therefore, an IP packet on Ethernet would add the shim before the IP header but after the Ethernet header. MPLS forwarding is currently defined for the following implementation of layer 2: Ethernet, packet over SONET, ATM, and Frame-relay. MPLS has also been defined for any medium that PPP runs on top of. On most of the layer 2 implementation a label consists of a 20 bit number. The shim that is added to the packet contains more than just a label. Here is a diagram of a MPLS shim [13].

As you can see the label is 20 bits. This value is used to determine how a packet will be label switched. The next 3 bits are called the EXP bits. They are currently reserved for experimental purposes. The next bit is referred to as the "bottom of stack bit" (S bit). Due to the fact that MPLS adds a shim to the packet, a LSR needs to know if what follows this top shim is the layer 3 header or another shim (Multiple shims are called a label stack. The purpose of a label stack will be explained later). The S bit signifies that what follows this shim is the layer 3 header. For typical single shim MPLS forwarding the S bit is on. Finally the shim contains the Time To Live (TTL) counter. This is used to allow current layer 3 functions to occur even though an LSR cannot use the layer 3 header. Some examples of these are trace-route, loop detection, and multicast domains [13].

When an LER adds a shim to a packet, it is feasible that i can add more than one shim. This concept is called Label Stacking. The stack of shims is treated just as its name sake data structure. A POP

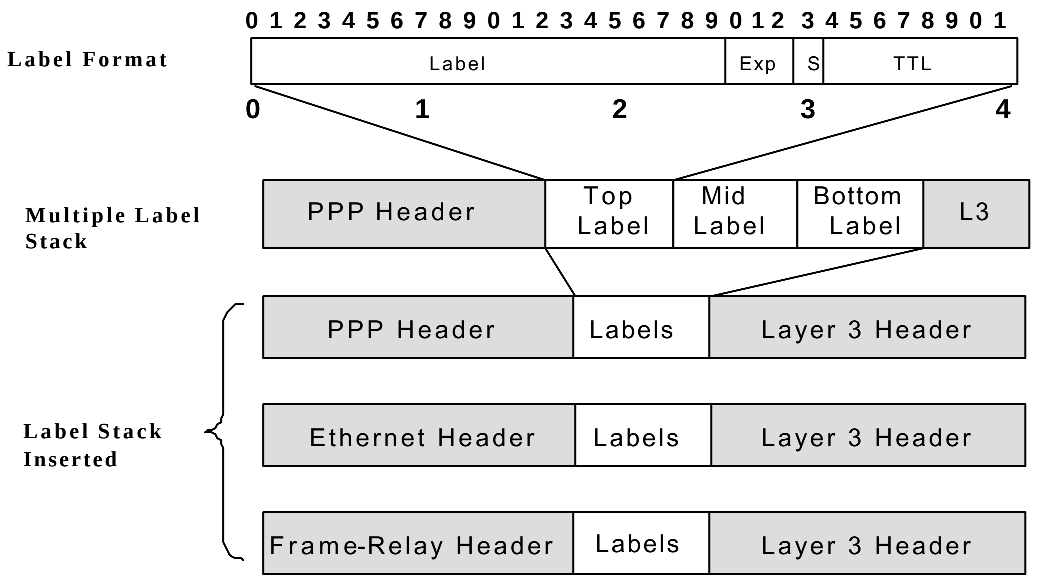


Fig. 1 Label stack

means that the top shim is removed, exposing either another shim or the layer 3 header (determined by the S bit). A PUSH adds a new shim to the top of the stack or on top of the layer 3 header. Therefore, standard label swapping is defined as a POP followed by a PUSH. In some cases a labeled packet may need to be tunneled across another MPLS network. In the case the labeled packet gets another shim pushed on top without POPing the original shim off. This results in a label stack of size 2. This operation can occur multiple times by separate LSRs, or a single LSR could add more than one shim. In general any labeled packet has a label stack although most have a label stack of size 1 [13].

2.2 MPLS Diff-Serv-aware Traffic Engineering

MPLS traffic engineering allows constraint-based routing of IP traffic. One of the constraints satisfied by CBR is the availability of required bandwidth over a selected path. Diff-Serv-aware Traffic Engineering extends MPLS traffic engineering to enable you to perform constraint-based routing of "guaranteed" traffic, which satisfies a more restrictive bandwidth constraint than that satisfied by CBR for regular traffic. The more restrictive bandwidth is termed a sub-pool, while the regular TE tunnel bandwidth is called the global pool. (The sub-pool is a portion of the global pool.) This ability to satisfy a more restrictive bandwidth constraint translates into an ability to achieve higher Quality of Service performance (in terms of delay, jitter, or loss) for the guaranteed traffic [5].

For example, DS-TE can be used to ensure that traffic is routed over the network so that, on every link, there is never more than 40 percent (or any assigned percentage) of the link capacity of guaranteed traffic (for example, voice), while there can be up to 100 percent of the link capacity of regular traffic. Assuming QoS mechanisms are also used on every link to queue guaranteed traffic separately from regular traffic, it then becomes possible to enforce separate "overbooking" ratios for guaranteed and regular traffic. Also, through the ability to enforce a maximum percentage of guaranteed traffic on any link, the network administrator can directly control the end-to-end QoS performance parameters without having to rely on over-engineering or on expected shortest path routing behavior. This is essential for transport of applications that have very high QoS requirements (such as real-time voice, virtual IP leased line, and bandwidth trading), where over-engineering cannot be assumed everywhere in the network [5].

DS-TE involves extending OSPF (Open Shortest Path First routing protocol) so that the available sub-pool bandwidth at each preemption level is advertised in addition to the available global pool bandwidth at each preemption level. DS-TE modifies constraint-based routing to take this more complex advertised information into account during path computation. The MPLS traffic engineering Internet Protocol (IP) explicit address exclusion feature provides a means to exclude a link or node from the path for an MPLS traffic engineering label-switched path (LSP). The feature is accessible via the IP explicit-path command that allows you to create an IP explicit path and enter a configuration submode for specifying the path. The feature adds to the submode commands the exclude-address command for specifying addresses to exclude from the path [5].

If the exclude-address for an MPLS traffic engineering LSP identifies a flooded link, the constraint-based shortest path first (CSPF) routing algorithm doesn't consider that link when computing paths for the LSP. If the exclude-address specifies a flooded MPLS traffic engineering router ID, the CSPF routing algorithm doesn't allow paths for the LSP to traverse the node identified by the router ID. However, in the meantime traffic engineering and QoS in IP networks became the dominant driving force behind MPLS [5].

Assuming the deployment of MPLS, the key question arises: how do we manage large MPLS networks? We decided to apply policy-based management concepts to managing an MPLS network because we considered this an appropriate way of dealing with large sets of managed elements. Using policy-based management for networks and systems has become very popular since the early work on policies [2], [3], [4]. Nowadays, some commercial products are available, which use some form of policies to configure and control networks. In the IETF there is a Policy Framework Working Group [5], which aims at resolving issues related to policy-driven management of IP networks. It includes the definition of a policy framework and information models for DiffServ, IntServ, and IP Devices.

3 IETF Policy Framework

The IETF Policy Framework is under development by the IETF Policy Framework working group. The framework consists of Policy Enforcement Points (PEP), Policy Decision Points (PDP), management console, and a directory to store policies together with user/network resource information as Figure 2.

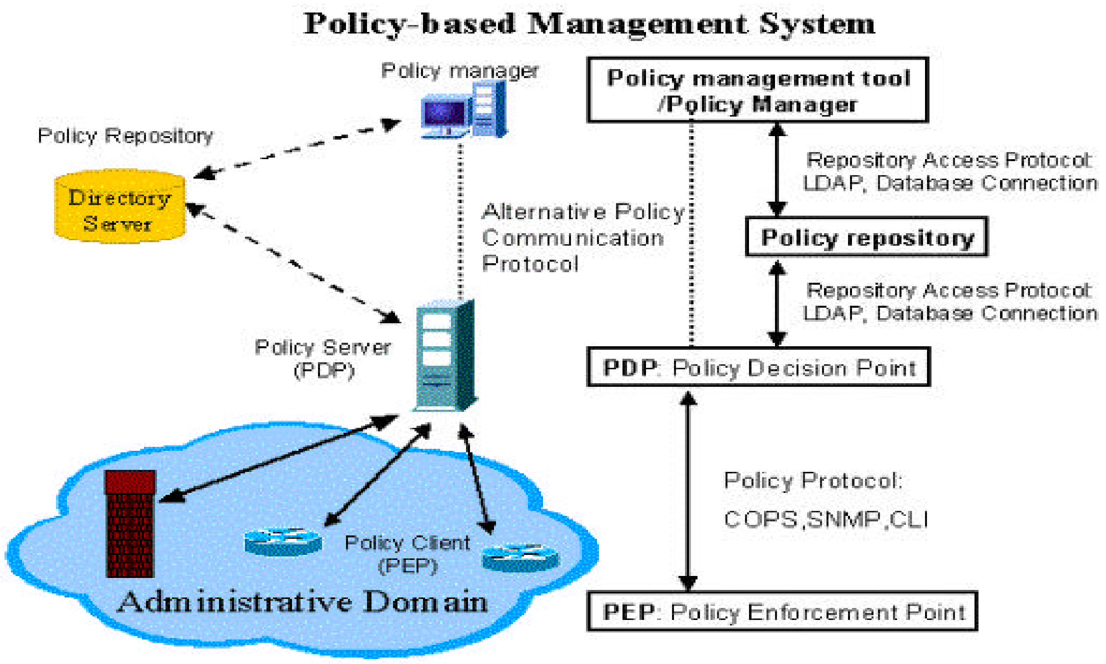

Policy-based Management System
Policy manager
Policy management tool /Policy Manager
Policy Repository
Directory Server
Repository Access Protocol: LDAP, Database Connection
Alternative Policy Communication Protocol
Policy repository
Repository Access Protocol: LDAP, Database Connection
Policy Server (PDP)
PDP: Policy Decision Point
Policy Protocol: COPS,SNMP,CLI
Policy Client (PEP)
Administrative Domain
PEP: Policy Enforcement Point

Fig. 2 IETF policy framework

PEPs are basically network elements and the PDP is typically referred to as the Policy Server (PS). The components are linked by the following protocols and languages.

The Common Open Policy Service (COPS) protocol [7] is used to forward requests from PEPs to the central policy server and to pass back corresponding policy decisions and support for reliability using TCP and keep-alive messages. Note that just recently an initiative to use policies in the SNMP framework has been started in the Configuration Management with SNMP (SNMPconf) working group [9]. A Policy Definition Language (PDL) is used to define new policies in terms of policy rules with condition and action lists [15].

What language to use is very controversial, and the IETF has not reached consensus in standardizing a Policy Definition Languages. Basically, each implementation defines its own language. The simple version of the X.500 directory access protocol called Light-Weight Directory Access Protocol (LDAP) is used by the policy server to retrieve information from the repository. Note that any other database may be used, but the working group decided to only provide a mapping of the policy model to a LDAP schema. The IETF policy framework activities are on one hand limited to DiffServ/IntServ based networks, and on the other mainly dealing with device configuration.

In this study, we proposed enhancing the IETF policy framework in two directions. First, we incorporate the management of Multi-Protocol Label Switching (MPLS) networks into the framework. MPLS is currently seen as a technology to influence the routing of IP networks in order to engineer the traffic with appropriate tools. QoS services are more easily and more flexible deployed in an IP-based network, because MPLS allows a network manager to control a route for an aggregate of flows [14].

However, MPLS per se does not have QoS features nor mechanisms, but MPLS together with Differentiated Services (DiffServ) is the favored approach by the IETF for providing IP QoS. The second enhancement of the policy framework is dealing with network-level and service-level management in IP networks. Using MPLS networks, the notion of a Label Switched Path (LSP) brings network-level concepts into the frame-work which has not been dealt with in the device-level policy framework. Furthermore, using traffic engineered network, new kinds of IP services are possible. E.g., a service guaranteeing low packet loss probability can be offered using MPLS as mechanism to traffic engineer the IP network in a way that traffic is routed around hot spots [15].

One of the key issues in the framework is how policy rules are triggered by state transitions or events. A Policy Decision Engine (PDE) is typically used to handle requests. For instance, in COPS for RSVP, the PEP issues a COPS request and the policy server returns a decision on whether to permit or deny the RSVP Path Message. A table of enabled policy rules is traversed at the PDE in order to find the matching rules for a request. The scenario is fairly clear for QoS signaling using the Resource Reservation Protocol (RSVP). RSVP path and reservation messages arriving at an RSVP daemon running on an IP router are converted by a COPS client component into COPS requests and sent to the COPS server component at the policy server [14].

The RSVP-related COPS request will be forwarded to the decision engine which makes a decision based on the applicable rules. A similar scenario can be described for the Differentiated Services approach, where the PDE may be triggered by a request for (initial) configuration issued by network elements. The requests contain a description of the element's capabilities, which are used in the PDE to decide on the configuration to load to the element. The similarity lies in the entity, which initiates the communication with the policy server. However, in the DiffServ case, the re-configuration of network elements may be triggered by new service level requests (SLS) or an operator manually re-configuring parts of the network. Both scenarios communicate in a push structure different from the RSVP scenario mentioned above. The second issue to be dealt within the policy server is the information model used to represent the policy information as well as network information [15].

4 Design a Policy-based Framework

4.1 Network Topology

We deployed a MPLS network using Cisco router for the testing network plane. The infrastructure of this inters-AS MPLS VPN for Diffserv Qos testing was showing in Figure 3. MPLS network include P (Provider) router that is responsible for label swapping in MPLS backbone network and PE (Provider Edge) router that is responsible for insert or pop label in the edge of MPLS network connecting with CE (Customer Edge) router which is in customer network and ASBR(Autonomous System Border Router) that is connected with other network service provider with MPLS network backbone. We use AS no. to distinguish with each other.

As shown in Figure 4, the tunnel configuration involves at least three devices including tunnel head, midpoint, and tail. On each of those devices one or two physical interfaces must be configured, for traffic ingress and egress. Figure 4 is a Sample Tunnel Topology for one link failure explicit routing for backup.

Figure 5 is a sample tunnel topology for unequal-cost load-sharing solution. The tunnel configuration involves at least three devices including tunnel head, midpoint, and tail. On each of those devices one or two physical interfaces must be configured for traffic ingress and egress. We use packet generator

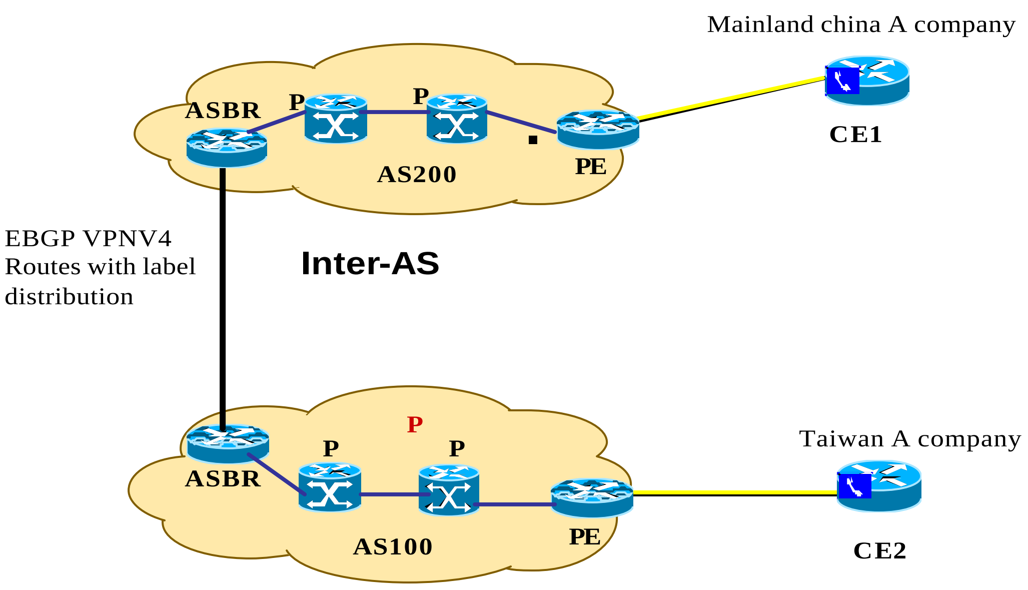


Fig. 3 Diff-serv for inter-AS MPLS VPN

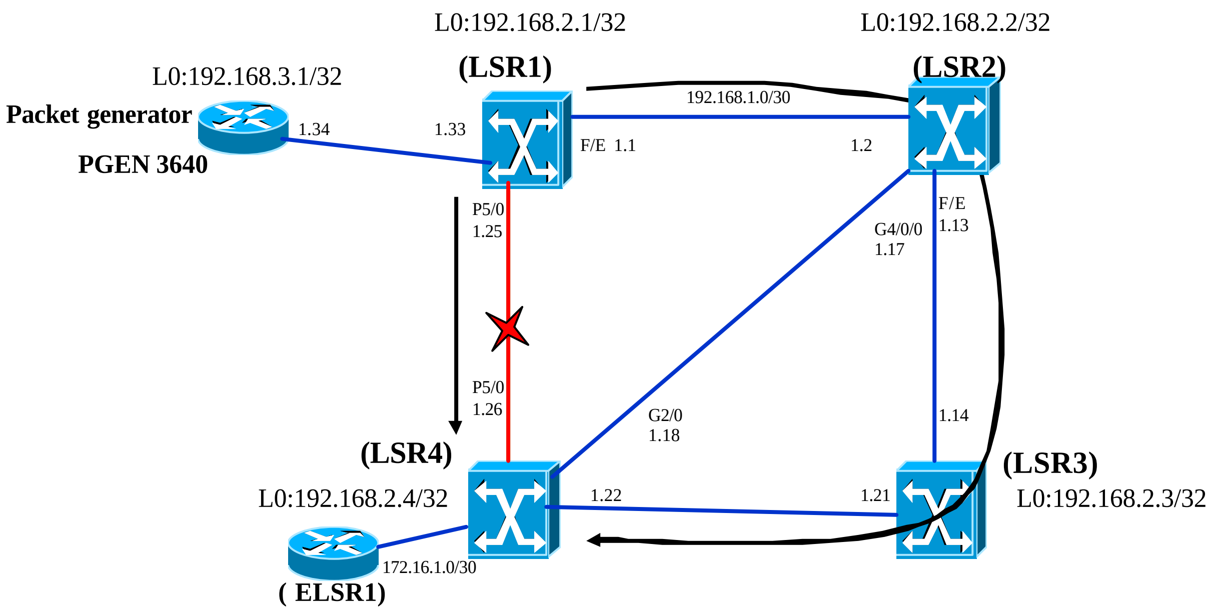


Fig. 4 Inter-serv for MPLS network backbone backup solution

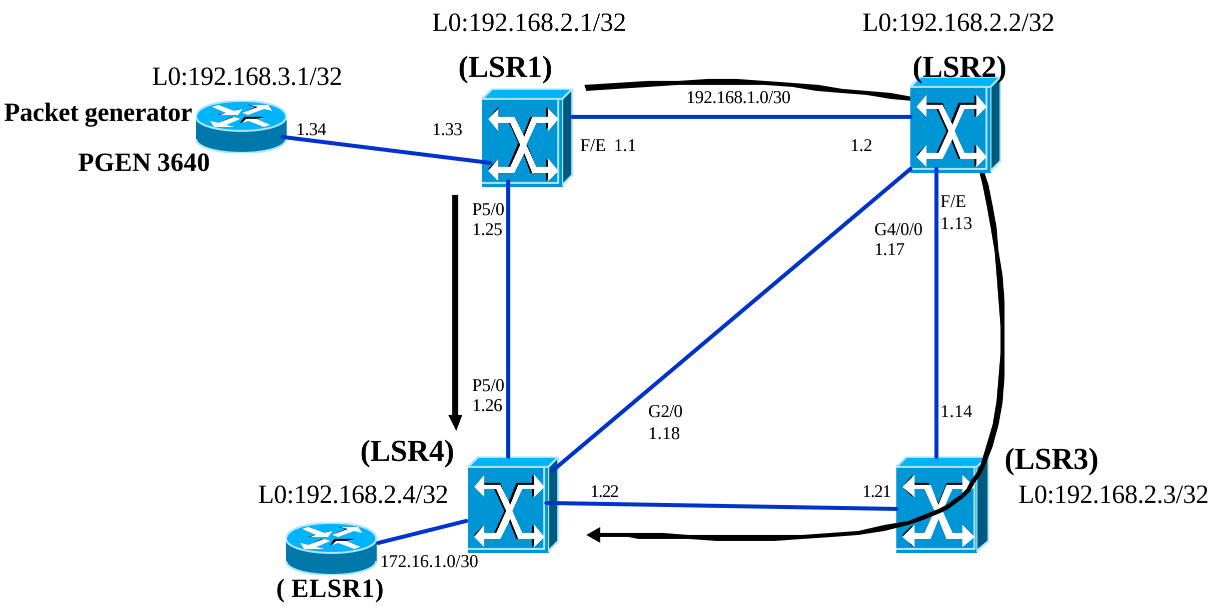


Fig. 5 Inter-serv for MPLS network backbone load-sharing

to create traffic flow into the MPLS backbone network to simulate real network condition. The tunnel configuration is described below.

(1). TE headend configuration

```
interface Tunnel0
ip unnumbered Loopback0
tunnel mode mpls traffic-eng
tunnel destination 192.168.2.4
tunnel mpls traffic-eng autoroute announce
tunnel mpls traffic-eng priority 1 1
tunnel mpls traffic-eng path-option 1 explicit
identifier 1
tunnel mpls traffic-eng path-option 2 explicit
identifier 2
```

(2). Basic options configuration

```
ip explicit-path identifier 1 enable
 next-address 192.168.1.26
!
ip explicit-path identifier 2 enable
 next-address 192.168.1.2
 next-address 192.168.1.14
 next-address 192.168.1.22
interface POS5/0
 description ****connect to LSR4 (1)***
 ip address 192.168.1.25 255.255.255.252
 no ip redirects
 no ip directed-broadcast
 no ip proxy-arp
load-interval 30
crc 16
clock source internal
mpls traffic-eng tunnels
tag-switching ip
ip rsvp bandwidth 75000 75000
```

4.2 Service-Level Agreement (SLA)

In this section we substantiate the notion of SLA. The definition of SLA is the first step toward the provisioning of QoS. Today, QoS-based services are offered in terms of contract agreements between an ISP and its customers. Such agreements will be greatly simplified through a standardized set of SLA parameters. An SLA standard is also necessary to allow for a highly developed level of automation and dynamic negotiation of SLAs between customers and providers.

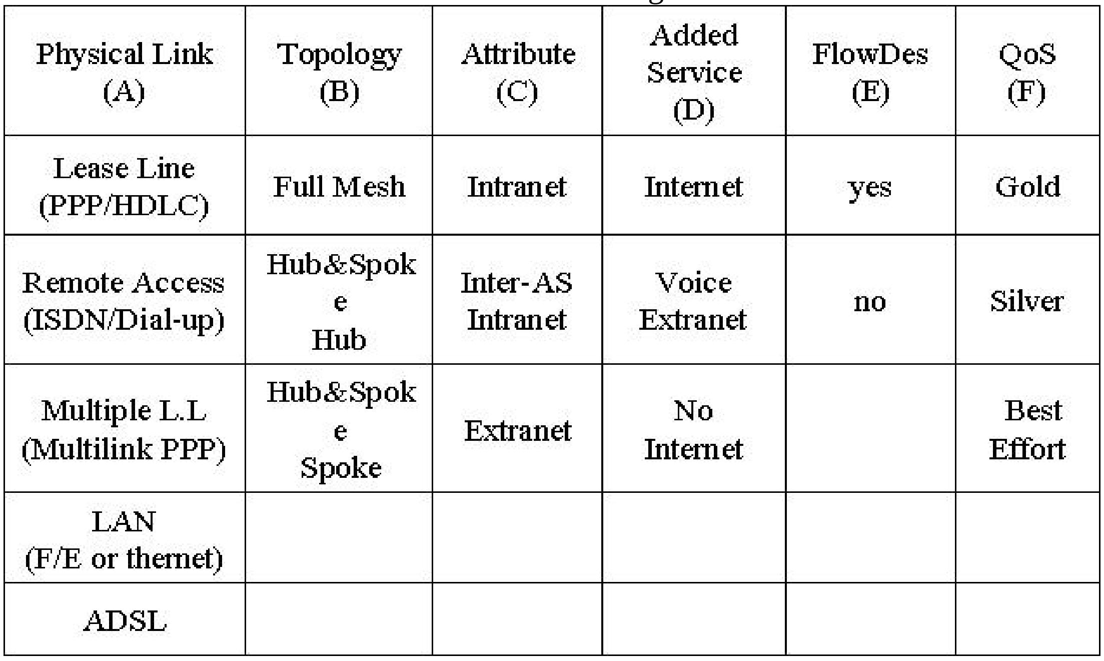
The contents of an SLA include the essential QoS-related parameters, including scope and flow identification, traffic conformance parameters, and service guarantees. More specifically, an SLA has the following fields as shown in Table 1: Physical Link, Topology, Attribute, Add service, FlowDes, Qos, and MPLS backbone network guarantees for performance parameters, service schedule, and reliability.

The scope of an SLA associated to a given service offering uniquely identifies the geographical and topological region over which the QoS of the IP service is to be enforced. An ingress (or egress) interface identifier should uniquely determine the boundary link or links as defined in [1] on which packets arrive/depart at the border of a DiffServ domain. This identifier may be an IP address, but it may also be determined by a layer two identifier in case of, say, Ethernet, or for unnumbered links like in, for example, Point-to-Point Protocol (PPP) access configurations.

The semantics allow for the description of one-to-one, one-to-many, and many-to-one communication SLA models, denoted (1|1), (1|N), and (N|1), respectively. The network service attributes of an SLA associated to a given service offering intranet or extranet or internet indicates for which IP packets the QoS policy for that specific service offering is to be enforced.

An SLA has only one FlowDes, which can be formally specified by providing one or more of the following attributes: FlowDes = (DiffServ information, source information, destination information, application information) Setting one or more of the above attributes formally specifies a SLS FlowDes. The DiffServ information might be

Table 1 MPLS service level agreement table



Physical Link (A)	Topology (B)	Attribute (C)	Added Service (D)	FlowDes (E)	QoS (F)
Lease Line (PPP/HDLC)	Full Mesh	Intranet	Internet	yes	Gold
Remote Access (ISDN/Dial-up)	Hub&Spok e Hub	Inter-AS Intranet	Voice Extranet	no	Silver
Multiple L.L (Multilink PPP)	Hub&Spok e Spoke	Extranet	No Internet		Best Effort
LAN (F/E or thernet)					
ADSL					

the DSCP. The source/destination information could be a source/destination address, a set of them, a set of prefixes or any combination of them. The FlowDes provides the necessary information for classifying the packets at a DiffServ edge node.

The packet classification can be either behavior aggregate (BA) or multifield (MF) based. The traffic descriptor includes traffic envelope and traffic conformance and describes the traffic characteristics of the IP packet stream identified by FlowDes. The traffic envelope is a set of traffic conformance (TC) parameters, describing how the packet stream should be in order to receive the treatment indicated by the performance parameters. The TC parameters are the input to the traffic conformance testing algorithms.

5 Implementation and Results

5.1 System Implementation

Figure 6 illustrated the actual implementation of policy-based management framework. We developed a policy server prototype for the management of MPLS networks in order to prove the feasibility of our architecture. The applicability to large MPLS/DiffServ networks has been shown by using Cisco router. However, at the current stage of the implementation, we can only show a working prototype proving the concept.

For more meaningful results many open issues in the area of service level agreement (SLA) request arrival and duration, traffic models of source etc. The prototype is based on the policy server, which was targeted to the area of IntServ and HTTP. It consists of a policy language together with a policy editor and an interface to LDAP directories. The server is implemented in Java. The policy language is a proprietary simple language, which allows an operator to specify policies in a human readable way.

The mapping of the policy language to the objects in the implementation is straight forward and easy to implement. As interface to policy clients, the policy server uses a protocol adaptor, which abstracts from real policy protocols such as COPS, SNMP for configuration, or our proprietary one to the simulator. According to the policy information model, we extended the IETF framework by MPLS policies.

MPLS policy classes are converted into a LDAP directory schema. Furthermore, we built Cisco router to a MPLS network which offers MPLS functionality. The policy server and policy manager run on different PCs. The interface is implemented

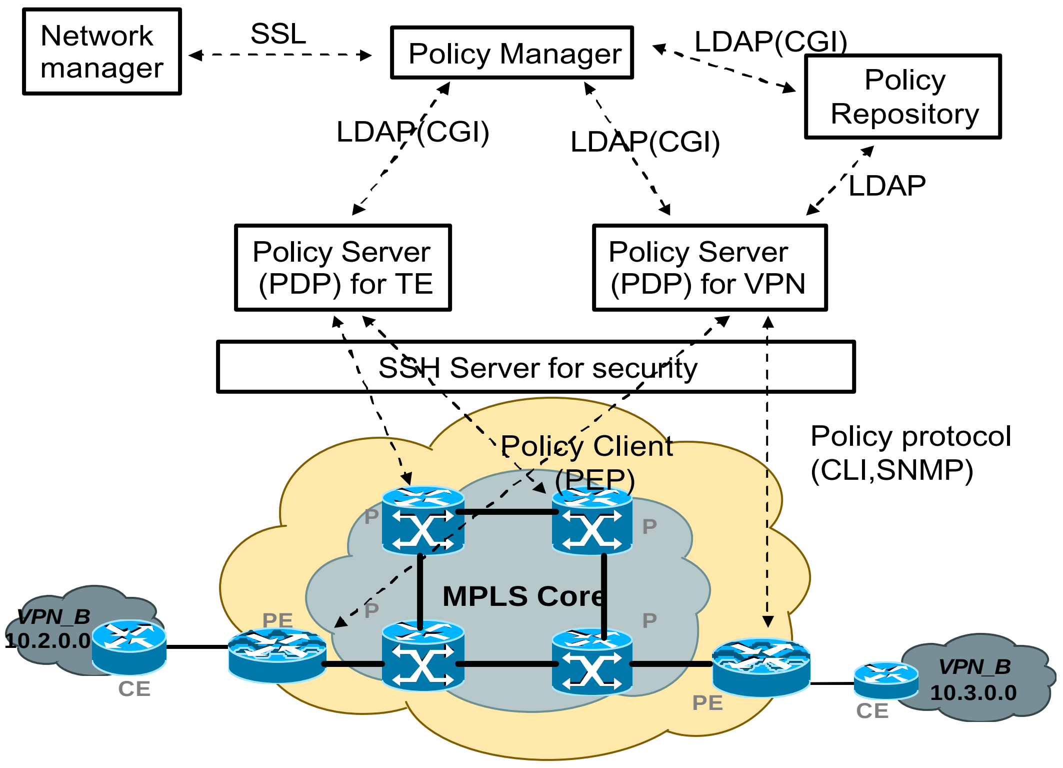


Fig. 6 Policy-based management framework

using a proprietary, COPS-like, text-based protocol between the real policy server and the Cisco router MPLS network using a TCP connection. All management agents send COPS-like messages to the real policy server. The messages from the Cisco router MPLS network to the policy server include always the network element's address and port number of the management agent.

The configuration messages from the policy server take the reverse way. Additionally, a simulation of a SLA requestor issues service requests to the policy server, and generates traffic. The service level requests are sent with the same mechanism described in the above paragraph. The source starts sending in case a SLA Permit decision is communicated back to it. Our policy server supports two kinds of policies: policies for device configuration and policies for deciding on service requests. Device configuration policies are triggered by devices, e.g. at start-up, while the other kind of policies is triggered by arriving service requests.

5.2 Evaluation

We describe a prototypical implementation of a policy-based management system for MPLS Traffic Engineering, operating on MPLS network elements. Several experiment made in our test environment illustrate the general efficiency and feasibility of our architecture. For example, the setup of MPLS traffic engineering tunnel over four hops is performed in one second, and finally, MPLS traffic engineering tunnel deletion also lasts about two seconds, this data is calculated from Cisco router history log file and policy server history log file. Policy repository is using MySQL database software to establish, and policy server is using simulation software of telnet function like manual CLI (Command Line Interface) to send configuration information to Cisco router.

6 Conclusions

We proposed a template for service-level agreement with a functional architecture for supporting the QoS required by contracted SLA, while trying to optimize use of network resources. The management plane aspects of our architecture include SLA subscription, traffic forecasting, network dimensioning, and dynamic resource and route management. All of these are policy-driven. The control plane aspects include SLA invocation and packet routing, while data plane aspects include traffic conditioning and PHB-based forwarding. The management plane aspects of our architecture can be thought of as a detailed of the policy server in the context of an integrated management and control architecture that aims to support both qualitative and quantitative.

We proposed a prototyping of a policy-based management system for MPLS Traffic Engineering, operating on MPLS network elements. Several experiments illustrate the general efficiency and feasibility of our architecture. For example, the setup of MPLS traffic engineering tunnel over four hops is performed in one second, and finally, MPLS traffic engineering tunnel deletion also lasts about two seconds.

Many of the functional blocks of this architectural model are also features of policy server the main difference being that a policy server is seen as driven purely by customer requests whereas in our approach, TE functions continually aim at optimizing the network configuration and its performance.

This system used a number of technologies for communications between the policy manager and policy server, with Lightweight Directory Access Protocol for accessing the SLA and network repositories. The interfaces to the routers are based on the Simple Network Management Protocol and command-line interfaces with an adaptation layer presenting a consistent interface to the management plane, which is independent of whether the underlying router is commercial or experimental.

Finally, the proposed DiffServ-oriented management and control framework was based on MPLS network. As such, we are fairly confident that the proposed architectural framework will result in a workable solution for end-to-end QoS in a DiffServ MPLS-based Internet. Diff-serv-aware MPLS TE is a powerful solution for improving network resource management. Guaranteed Bandwidth Services is assuring value-added services better availability with TE, scalable VPN solution.

References:

[1] S. Blake et al., *An Architecture for Differentiated Services*, RFC 2475, 1998.

[2] P. Georgatsos et al., Technology Interoperation in ATM Networks: The REFORM System, *IEEE Commun. Mag.*, Vol.37, No.5, May 1999, pp. 112-118.

[3] D. Awduche et al., *A Framework for Internet Traffic Engineering*, Working Paper, July 2000.

[4] E. Rosen, A. Viswanathan, and R. Callon, *Multi-protocol Label Switching Architecture*, RFC 3031, Jan. 2001.

[5] P. Aukia et al., RATES: A Server for MPLS Traffic Engineering, *IEEE Network*, Mar./Apr. 2000.

[6] A. Feldmann et al., NetScope: Traffic Engineering for IP Networks, *IEEE Network*, Mar./Apr. 2000.

[7] B. Teitelbaum, *Qbone Architecture (v1.0)*, 1999, http://www.internet2.edu

[8] K. Nichols, V. Jacobson, and L. Zhang, *A Two-Bit Differentiated Services Architecture for the Internet,* RFC2638, July 1999.

[9] D. Goderis et al., *Service Level Specification Semantics and Parameters*, Working Paper, Nov. 2000.

[10] P. Flegkas et al., On Policy-based Extensible Hierarchical Network Management in QoS-enabled IP Networks, *Proc. Policies for Dist. Sys. and Networks*, 2001.

[11] P. Aukia et al., RATES: A Server for MPLS Traffic Engineering, *IEEE Network*, Mar./Apr. 2000.

[12] K. Dave, *Understanding Policy-based Networking*, Willey Computer, 2001.

[13] I. Pepelnjak, and J. Guichard, *MPLS and VPN Architectures*, Cisco Press, 2001.

[14] C.K. Wang, Policy-based Network Management, *WCC-ICCT,* 2000.

[15] J. Guichard, *A Primer on Policy-based Network Management,* HP Company, 1999.