Diffserv over MPLS QoS Policy Management

Yin Ling Liong\textsuperscript{1}, Roberto Barnes\textsuperscript{2}, Man Li\textsuperscript{1}

\textsuperscript{1} Nokia Research Center, 5 Wayside Rd
Burlington, MA 01803, USA
{Yin-ling.Liong, Man.M.Li}@nokia.com
\textsuperscript{2} Nokia Research Center, Itämerenkatu 11-13
00180 Helsinki, Finland
Roberto.Barnes@nokia.com

Abstract. Policies are a set of rules to administer, manage and control access to network resources\cite{1}. Policy based management employs a policy server to manage a network as a whole. It translates business goals or policies into configurations of network resources and automates the configurations across multiple network elements. In this paper, we investigate how to manage a Diffserv/MPLS network with policy-based management. Specifically, we review the industrial standard of how to support Diffserv over MPLS network. Then, we investigate how to construct policies to support Diffserv/MPLS internetworking, and how to provide MPLS traffic engineering using explicit routing and constraint based routing. Finally, we describe a policy server prototype and a test-bed that includes four Cisco routers to demonstrate the feasibility of policy based Diffserv/MPLS management.

1 Introduction

With the rapid growth of the Internet and the strong demand for differentiated customer services, Differentiated Services (Diffserv) emerged as a technology that offers scalable solutions to the network service providers in offering services with different quality-of-service (QoS) objectives. At the same time, reliability and flexible routing control are becoming more and more important. Therefore, Multi-protocol Label Switching (MPLS) has evolved from a packet-forwarding technology into an advanced traffic engineering and link protection tool. It gives network operators a high degree of control over the paths taken by packets on their networks. This enables operators to better utilize the network resources while providing a wide range of services to users. However, MPLS does not define or contain QoS services. Thus, combining MPLS with Diffserv will enable MPLS with IP QoS support.

Despite so many merits, the use of Diffserv over MPLS also brings complexity to the management plane. Since they are designed separately for different purposes, the interworking requires certain compromises and future extensions. The current management for Diffserv/MPLS mainly involves direct configuration of individual elements. Thus, it is difficult for the operators to maintain consistency across multiple network elements. In view of this, policy based management is proposed to support
Diffserv over MPLS. A 3-tier architecture that includes policy server, policy targets and a central database is used. The policy server translates business goals or policies into configurations of network resources and deploys the configurations across multiple network elements and different technologies (e.g. MPLS and Diffserv). It also simplifies service creations and shortens service delivery cycles.

In recent years, Diffserv/MPLS policy based management has attracted a lot of attention. The TEQUILA project ([2], [3]) is one of the examples. It focuses on the architectural framework for supporting end-to-end QoS with Diffserv and MPLS traffic engineering [2]. The framework includes three main components: Service Level Specification Management (SLSM), Traffic Engineering (TE) and Policy Management (PM), with the assumption of certain monitoring and data plane functionalities. The network dimensioning policies under TE are explained in [3]. However, the details of other functional blocks are still under development.

The TEAM automated manager for Diffserv/MPLS networks is another interesting example [4]. It is a network management tool that embeds with new algorithms to provide dynamic bandwidth and route management of MPLS tunnels. The automated manager consists of three main components: Measurement/Performance Evaluation Tool (MPET), Traffic Engineering Tool (TET) and Simulation Tool (ST). The TEAM automated manager develops and integrates many start-of-the-art algorithms to provide an automated network management platform. A test bed that consists of high-end Cisco routers has also been built to evaluate the algorithms and demonstrate the automated manager’s capabilities.

In contrast to TEAM, our work does not focus on devising and integrating new algorithms for automated management. Instead, based on the current capabilities of the network elements, we propose a framework that uses policy-based management to support Diffserv/MPLS interworking and MPLS traffic engineering. The framework includes the relevant policy elements and the construction of service and network policies to support the objectives. Then, we designed a policy server prototype to prove the concept and built a testbed that consisted of four Cisco 3620 routers to demonstrate the capabilities. In the following, Section 2 will give a technology overview of Diffserv, MPLS and MPLS traffic engineering and identify their policy elements. Section 3 will describe the challenges of supporting Diffserv/MPLS management. Section 4 will describe the policy server prototype and explain the components that support Diffserv/MPLS interworking, explicit routing and constraint based routing. Finally, Section 5 will conclude the paper.

2 Technology Overviews

In this section, we will give a brief overview of the key concepts concerning Diffserv and MPLS, and will review the standard practice of Diffserv/MPLS interworking. The policy elements of the different technology will also be identified.
2.1 Differentiated Services (Diffserv)

Differentiated Services (Diffserv) provides a scalable QoS support to IP networks. Traffic is classified and conditioned at the edge of the network by edge devices such as edge routers or gateways. A Diffserv code point (DSCP) is then applied to the IP header of each packet. Core routers of the network handle packets in different traffic streams by forwarding them with different per-hop behaviors (PHB). The PHB to be applied is indicated by the DSCP in each packet header. The advantage of Diffserv is that many traffic streams can be aggregated to a small number of behavior aggregates that are each forwarded using the same PHB at core routers. This approach simplifies the processing and storage requirements at core routers.

A Diffserv QoS policy to be applied to router interfaces consists of one or more cascading Traffic Conditioning Blocks (TCB). Fig. 1 shows the elements of a TCB. Some elements in a TCB, e.g., meter, can be unspecified. Please refer to [5] and [6] for details of Diffserv QoS policy.

Fig. 1. A Traffic Conditioning Block (TCB)

To support Diffserv/MPLS, RFC 3270 [7] recommends the use of one set of Diffserv queues at each interface. In other words, all LSPs will share the same queue if they have the same PHB Scheduling Class (PSC). PSC is a set of one or more PHB(s) applied to the behavior aggregates that share an ordering constraint. For example, AF1x is a PSC comprising the AF11, AF12 and AF13 PHBs [7]. Thus, no per-LSP queuing will be employed. This provides scalability to large number of LSPs. The uniformity of queue structure also simplifies the network management.

2.2 MPLS and Traffic Engineering

Multi-protocol Label Switching (MPLS) is a packet-forwarding technology that gives network operators a high degree of control over the paths taken by packets on their networks. Ingress routers at the edge of an MPLS domain classify incoming IP packets into Forward Equivalent Classes (FEC) or groups of packets that must be forwarded in the same manner (i.e., follow the same path) inside the domain. The unidirectional path taken by packets belonging to a particular FEC is defined as a Label Switched Path (LSP). They can be setup by using Label Distribution Protocols, such as LDP [8], RSVP-TE [9], and CR-LDP [10]. A setup priority and a holding priority may be associated with an LSP. The setup priority defines the priority of the LSP during setup whereas the holding priority defines the pre-emption priority of the LSP.

MPLS traffic engineering allows operators to establish routes based on information other than shortest path, such as delay and bandwidth available along the path. In
contrast, Interior Gateway Protocols (IGP), such as OSPF and IS-IS, use the destination address of the packets and the shortest path to reach the destination to route packets. Operators may want to establish routes for certain customers based on information other than the shortest path. This provides the ground for MPLS traffic engineering, which can relieve congestion and maximize bandwidth utilization by allowing multiple paths between source and destination. There are other benefits for MPLS such as fault tolerance and link protection. However, this study only focuses on managing the MPLS traffic engineering capability offered by Explicit Routing and Constraint-Based Routing (CBR), which will be explained as follows.

**Explicit Routing.** The operator explicitly specifies the optimal path to be traversed by the LSP and configures the ingress router to set up the path with a label distribution protocol. The operator may determine the optimal path with the help of a network-planning tool for example. This approach is suitable for off-line traffic engineering.

**Constraint-Based Routing (CBR).** The operator does not specify the path but depends on a constraint based routing mechanism that has been implemented in the network to determine the path. With CBR, every router advertises traffic-engineering attributes (e.g., maximum bandwidth, unreserved bandwidth, etc.) for its interfaces to all other routers. OSPF and IS-IS protocols have been extended for that purpose ([11], [12]). As a result of the advertisement, every router obtains a “traffic engineering database” in addition to its regular routing database. When an LSP with QoS requirements needs to be established, the ingress router will compute the optimal path based on its traffic-engineering database and a path selection algorithm. It then signals the path establishment with a label distribution protocol. The explicit path selected is conveyed through the signaling protocol.

The policy elements that support CBR include link attributes and affinity. The link attributes is associated to each network interface and are advertised by the IGP to form the traffic-engineering database. The affinity is attached to a LSP. It is used by the IGP to choose the preferred link attributes when routing LSPs, and form part of the path selection algorithm. There are other algorithms that require additional parameters for better resource and bandwidth management. The challenges are discussed in Section 3.2.

### 2.3 Diffserv over MPLS

There are two solutions in supporting Diffserv/MPLS inter-working: EXP-Inferred-PSC LSP (E-LSP) and Label-Only-Inferred-PSC LSP (L-LSP) [7]. An E-LSP uses the EXP field in the MPLS shim header to infer the PHB of a packet. In contrast, an L-LSP uses the MPLS label to carry the PSC, and the EXP field to convey the PHB. In order to support these two solutions, operators have to map between Diffserv and MPLS policies.

There are two essential policy elements that serve the Diffserv/MPLS inter-working: EXP-to-PHB mapping and Tunneling Mode. To support E-LSP, the opera-
tors should define the EXP-to-PHB mapping to ensure rule consistency across a network. Then, this mapping can be configured into the routers directly, or through RSVP-TE or CR-LDP signaling during tunnel set-up. To support L-LSP, the label-to-PSC and EXP-to-PHB mapping has to be defined. Since the MPLS labels are swapped at every hop, operators have to rely on signaling to set the label and EXP mapping.

Tunneling mode defines the way to translate the Diffserv information in the MPLS headers (labels and EXP field) into the DSCP value in the encapsulated IP header when packets exit the MPLS network. There are two modes of tunneling: Pipe Mode and Uniform Mode. For Pipe Mode, the egress router would keep the DSCP of the encapsulated IP header and discard any changes made to the behavior aggregate membership within the MPLS cloud. For Uniform Mode, the egress router would overwrite the original DSCP with the Diffserv information in the MPLS network. Details of the EXP-to-PHB mapping and tunneling modes could be found in [7].

3. Challenges of Diffserv/MPLS management

There are many challenges in supporting Diffserv/MPLS management due to different recommendations from the standards, and the limited capabilities supported at the network elements. In the following, we will discuss the challenges of supporting L-LSP, CBR and implicit null labels in policy based management.

3.1 Challenges of supporting L-LSP

The support of L-LSP requires explicitly signaling the label-to-PSC and EXP-to-PHB mapping associated with an L-LSP during label establishment. However, the required protocol extensions were not complete at the time of study. Therefore, the detailed support of L-LSP by policy-based management is not evaluated. Currently, the IETF Internet Traffic Engineering (tewg) working group has active discussion of supporting DiffServ-aware Traffic Engineering (DS-TE). According to [13], DS-TE allows the traffic to utilize resources available to the given class on both shortest paths and non-shortest paths and to follow paths that meet its engineering constraints by mapping the traffic from a given Diffserv class on a separate LSP. In other words, DS-TE requires the use of L-LSP. The protocol extension to support DS-TE (and therefore L-LSP) is an Internet Draft currently [14] and can contribute to the future work of the study.

3.2 Challenges of supporting CBR

The support of CBR is reduced to the configuration of link attributes and affinity in the policy server. The first challenge here is how to use the link attributes and affinity to achieve the goals of the operators. Some device implementation uses 16-bit variables to represent link attributes and affinities. However, it is not clear how to inter-
pret the 16-bit variables, and how to enforce the same definition across a network. Moreover, manually configuring hundreds of these attribute and affinity variables in a large network is error prone and difficult to debug. So, there is a need for the policy server to define the semantics for the attributes and affinities supported across the network. We will explain our solution in Section 4.4. The second challenge is how to provide better resource and bandwidth management by CBR, such as enforcing different bandwidth constraints for different Diffserv classes. There are active researches and industrial activities in this area, such as DS-TE ([13], [14]) and TEAM automated manager [4]. The support for these new mechanisms can be a future study.

3.3 Difficulty of using implicit null labels

The use of implicit null labels at penultimate hop (next-to-last router) can pose difficulty to Diffserv/MPLS interworking. MPLS standards propose the use of implicit null labels between the penultimate and egress routers of an MPLS LSP, which means the MPLS shim header is completely removed at the penultimate hop. This lifts the burden off the egress router of having to do either one MPLS label lookup and an IP address lookup, or two MPLS label lookups, before forwarding a packet. However, by using the implicit null mechanism, the EXP field in a packet’s shim header is lost before reaching the egress router, losing any changes done to the behavior aggregate membership of a packet within the MPLS domain.

The lost of EXP information can be solved by using explicit null labels. A penultimate hop that implements explicit null forwards an MPLS packet to the egress router with a shim header where the value of the label has been set to zero. Upon receiving a packet with label value of zero, the egress router knows it can discard the label, and does not need to perform MPLS label lookup. However, the egress router does receive an intact EXP field with the packet, and the problem of propagating the Diffserv information from within the MPLS domain is solved.

There are certainly other mechanisms to implement the same behavior. Most of them rely on having the penultimate hop do any MPLS-Diffserv to IP-Diffserv processing (for example, the EXP-to-DSCP conversation, in the case of the uniform tunneling mode). This option can be undesirable because it moves some of the IP processing to devices within the MPLS cloud, hence this approach does not provide a clear separation between (possibly) MPLS-only devices and MPLS/IP devices. Therefore, the prototype supports explicit null labels in configuring the routers.

4. Diffserv/MPLS QoS Policy Server Prototype

We have implemented a Diffserv/MPLS policy server and built a testbed as shown in Fig. 2. The test bed consists of four Cisco 3620 routers, and is built to demonstrate the capabilities of the policy server prototype in configuring Diffserv/MPLS interworking and MPLS traffic engineering. Two routers that connect to the customers serve as edge routers, and the other two routers serve as core routers. The policy server can construct multiple LSPs between the edge routers through explicit route or
constraint based routing, as indicated by the arrows in Fig. 8. It can also map both customer X and Y traffic into one LSP or split them into two LSPs by service policies. In addition, the policy server can assign different applications of a customer into different Diffserv classes by service policies, and allocate the resources accordingly by network policies. A sniffer is connected to the test bed to verify the EXP to DSCP mappings.

![Diagram of constraint based routing](https://via.placeholder.com/150)

**Fig. 2.** DiffServ/MPLS policy based management testbed

The architecture of the policy server is shown in Fig. 3. It follows a 3-tier policy management architecture that consists of policy server (or Policy Decision Point), policy targets (or Policy Enforcement Points) and a central database. The policy server consists of Service Application (SA), Central Processing Facility (CPF) and Policy Consumer (PC). They are all connected to a central database through a database access interface. SA is a user interface that allows operators to add, delete or modify policies. CPF translates high-level business policies into detailed policy pa-

![Diagram of policy server architecture](https://via.placeholder.com/150)

**Fig. 3.** The architecture of the DiffServ/MPLS policy server prototype
rameters and stores them in the database. It also conducts policy verification, conflict
detection and resolution. PC provides an interface for the CPF to communicate with
the policy targets. It translates the device-neutral policies stored in the database into
the device-specific commands, and deploys them to the policy targets.

As shown in Fig. 4, SA consists of six objects: Services, Applications, Customers,
Devices, MPLS Tunnels and Network Policies. The first three objects are related to
the customer or service configuration while the other three objects are related to net-
work configuration. The Services object defines the services to be provided to cus-
tomers. Each service is defined as a mapping between a service name and any of the
fourteen DSCP numbers defined by the IETF [15][16]. The Applications object con-
tains the classification rules for different applications, such as protocol, source port
and destination port ranges. The Customers object contains the service policy infor-
mation for each of the customers. It also contains host groups, metering profiles and
policies associated with the customer. The Devices object contains information about
the network elements or policy targets that will receive the policies. Most of the in-
formation in the Devices object should be directly imported into the policy server
database from element management database. The MPLS Tunnels object is used to
specify the MPLS and Diffserv/MPLS policies. Its details will be explained in Sec-
tion 4.4. Finally, the Network Policies object defines the network policies that specify
the queuing and scheduling treatment of specific service and must be attached to
specific interfaces. The detailed policy elements and the key design concept are ex-
plained in the following subsections.

![Fig. 4. The construction of the Service Application (SA)](image)

### 4.1 Role and rule approach for scalability

In a large network, it is likely that the same set of rules is applied to multiple routers
or router interfaces. To achieve better scalability, policy based management adopts a
“role and rule” approach. Router interfaces are assigned role names. Policies or rules
are then specified and associated with roles. Interfaces with the same role names will
get the same set of rules or policies. This role and rule approach has been imple-
mented in the policy server prototype.
4.2 Separation of service and network policies

Service policies refer to rules that govern the treatment of individual customer traffic, such as traffic classification, metering and marking. Network policies refer to rules that govern the treatment of aggregated traffic, such as queue and scheduler configurations for behavior aggregates. When a new Service Level Agreement (SLA) is created or when an existing SLA is modified, an operator is likely to modify service policies with the policy server. When new facilities are added into a network or when traffic patterns are changed significantly, an operator is likely to conduct a sequence of network planning steps. The operator is then likely to modify network policy according to the results of network planning with the policy server.

![Fig. 5. The construction of the service policies in the prototype](image)

The service policies we proposed are shown in Fig. 5. The service policies consist of rules that specify the treatment of the individual customer traffic. The classification rule consists of the application profile and source/destination host groups. The metering rules use the simple leaky bucket algorithm specified by the traffic profiles. The service name selects the Diffserv class that supports the classified traffic. In addition, the service policies define the non-conformance action (e.g. marking or dropping) after metering and map the traffic into the desired MPLS tunnels and tunneling mode. Finally, the service policies use the role names to specify the device interfaces that will receive the policies.

![Fig. 6. The construction of the network policies in the prototype](image)

The network policies we propose are shown in Fig. 6. It defines the PHB that is used to support a particular service or Diffserv class, such as queuing and scheduling.
The network policies are then deployed to the device output interfaces through the associated role name. In each interface, all LSPs will share the same queue if they have the same PSC, and no per-LSP queuing will be employed.

4.3 Managing in multi-vendor environment

An important element in policy management is the so-called common information model. In a device independent way, it describes policies as relationships between network objects. The information model is used to derive database schema for policy server database so that the policy stored in the database should also be device independent.

In a multi-vendor environment where routers from different vendors are configured through different protocols, e.g., COPS, SNMP, CLI, a policy server adapts the policy stored in the database into device specific formats and delivers the resulted policy through the specific protocol. This approach enables a consistent provision of policies across multi-vendor networks.

The policy server prototype adapts the policy stored in the database into Cisco router configuration commands and delivers those commands through command line interfaces (CLI).

4.4 MPLS Tunnel Group policy elements

![Diagram of MPLS Tunnel Group policy elements](image)

The policy elements of the MPLS Tunnels object we propose are shown in Fig. 7. They consist of Tunnel Group, Tunnel Characteristics, Explicit Route, Route Affinity and EXP-to-Service Map. A Tunnel Group is a set of tunnels that share the same properties and form a certain topology, such as mesh or star. It is used to create, maintain and delete MPLS tunnels. The advantage of introducing the concept of tunnel group is that operators do not need to create the tunnels one by one. Instead, they only need to specify the end-point routers and the inter-connecting topology. The internal routes traversed by a tunnel will be determined by the underlying routing
protocols, such as OSPF or IS-IS. In the case of star topology, the hub and the spokes are specified explicitly.

The properties of a tunnel group include tunnel characteristics, set-up priority and holding priority. Tunnel characteristics are profiles that specify the parameters for resource reservation and policing per MPLS tunnel. The set-up priority determines which tunnel group can establish first in times of resource contention. If preemption is supported, a tunnel group with high set-up priority can preempt a tunnel group with low holding priority. In the standard, the set-up priority and holding priority are associated with an MPLS tunnel. However, in our prototype, the two priorities are attached to the tunnel group for scalability reason. In the following, we will explain the policy elements we developed with regards to explicit routing and CBR.

**Explicit Routing.** Explicit routing allows the operators to explicitly specify the optimal path for forwarding the packets. According to Fig. 7, the Explicit Route object contains a list of interfaces that an explicit route LSP will traverse through. The list of IP addresses can be a partial or complete route. There can be more than one explicit route list and thus a pointer is necessary to link the list to the Tunnel Group. In addition, each explicit route LSP is unidirectional. Therefore, a bidirectional LSP would require configuration of two explicit route LSPs in opposite direction.

**CBR.** CBR allows operators to route packets other than using shortest path algorithms. There are two steps in configuring a Tunnel Group using constraint based routing: setting the link attributes and attaching the route affinity to the Tunnel Group.

![Policy elements of Device object in the prototype](image)

Fig. 8 shows the policy elements in the Device object that set the link attributes in each interface. As explained in the challenges in Section 3.2, the policy server should define the semantics of the attributes and affinities supported. This is done by the Attribute Bit-Map element in the Device object. It allows operators to define a high-level representation for each attribute bit, such as the specific data rate, link technology or VPN membership. Each attribute definition consists of bit position, attribute names and category. An example of the attribute name is “Ethernet” and the category
is “Link Technology”. The high-level representations will then be stored in the database and shared across the different objects. For instance, the attributes can be added to each device interface. Thus, operators can select the right set of attributes to describe the interface.

The second step involves the use of Route Affinity object in Fig. 7. It defines affinity profiles that select the preferred link attributes in routing. The profile indicates whether each attribute defined in the Attribute Bit-Map is preferred, not preferred or “don’t care”, and is linked explicitly to a Tunnel Group. CPF then converts the affinity profile into device-specific inputs for database storage, which is then used with an extended IGP protocol to support CBR.

4.5 Diffserv/MPLS policy elements

We propose that The Diffserv/MPLS policy elements include the EXP-Service Map (Fig. 7) and the linking of Tunnel Group and Tunneling Mode in service policies (Fig. 5). The EXP-Service Map allows operator to configure the mapping between the EXP field in the MPLS shim header and the services defined in the Service object. Since each service is linked to a DSCP value in the Diffserv network, the mapping also translates the EXP field into a specific DSCP value. Thus, service differentiation is achieved by parsing MPLS shim headers only. In the prototype, this static mapping is distributed to all routers during policy deployment, thus limiting the number of Diffserv classes to eight. In order to support more than eight Diffserv classes in the network, the EXP-Service Map can be attached to a Tunnel Group, as indicated by a dotted arrow in Fig. 7. L-LSP is not currently supported, as explained in the challenges in Section 3.1.

The linking of Tunnel Group in service policies identifies the LSPs that carry the customer traffic. The use of Tunneling Mode decides what DSCP code point should be carried in the IP headers when a packet exits the MPLS network. It can be either Uniform Mode or Pipe Mode, as explained in Section 2.3. The Tunnel Group and Tunneling Mode are used only if operators use MPLS to carry the customer’s traffic. If MPLS is not used, they can be leave as empty.

5. Conclusion

In the paper, we have identified the policy elements required to support the Diffserv/MPLS interworking and MPLS traffic engineering. Moreover, we described the design of the policy server prototype we developed, and the challenges we faced during the design and the development process. In addition, a testbed using four Cisco 3620 routers is built to demonstrate the capabilities of the policy server prototype and the feasibility of Diffserv/MPLS policy management. Other works in Diffserv/MPLS policy management have also been reviewed and compared. This study builds on top of the capabilities of existing routers, and identifies most of the policy elements necessary at this time. A natural extension of the current work is the support
of DS-TE and other state-of-the-art CBR algorithms when they become mature. This will be an interesting work in the future.

6. References