Uppgjord (även faktaansvarig om annan) - *Prepared (also subject responsible if other)*		Nr - *No.*		
ETX/D/A Pablo Cebrian				
Dokansv/Godkänd - *Doc respons/Approved*	Kontr - *Checked*	Datum - *Date*	Rev	File
ETX/D/A Eric Lin		1999-10-25	PA1	report.fm

SUPPORT FOR DIFFERENTIATED SERVICES IN MULTI-PROTOCOL LABEL SWITCHING

Abstract

Multi-Protocol Label Switching and Differentiated Services are two emerging technologies which address problems such as traffic engineering and provisioning of QoS. This thesis approaches the question of how to design an MPLS network with support for Differentiated Services.

In order to provide other forwarding treatments than best effort, MPLS needs to dedicate resources in the network to providing preferential virtual paths to forward non-BE packets. Three solutions to the management of QoS resources in an MPLS network are proposed, examined and compared. The implementation of one of the solutions on Ericsson's AXI 51x Access Router is described in detail.

Abbreviations and Acronyms

ATM	Asynchronous Transfer Mode
BGP	Border Gateway Protocol
DiffServ	Differentiated Services
DS	Differentiated Services
DSD	Differentiated Services Database
FEC	Forwarding Equivalence Class
IETF	Internet Engineering Task Force
IP	Internet Protocol
LAN	Local Area Network
LER	Label Edge Router
LSP	Label Switched Path
LSR	Label Switch Router
MPLS	Multi-Protocol Label Switching
OSPF	Open Shortest Path First
PHB	Per Hop Behaviour
PS	Premium Service
PVC	Permanent Virtual Circuit
RADIUS	Remote Authentication Dial In Service
SoHo	Small office / Home office
QoS	Quality of Service
TCP	Transport Control Protocol
TOS	Type of Service
UBR	Unspecified Bit Rate
UDP	User Data Protocol
VCI	Virtual Circuit Identifier
VPI	Virtual Path Identifier

Contents

1 INTRODUCTION

Two new technologies are currently under development at Ericsson and other companies: Multi-Protocol Label Switching and Differentiated Services. Ericsson's AXI 51x Edge Router includes both of them, but uses them separately, not supporting DiffServ when running MPLS.

1.1 MULTI-PROTOCOL LABEL SWITCHING

A promising IP forwarding technique that combines the performance characteristics of layer 2 networks while maintaining the wide connectivity of IP (layer 3) addressing is *Multiprotocol Label Switching* (MPLS). The basic concept is that by prepending a label to each packet, packets can be forwarded along a Label Switched Path (LSP) at faster rates via the labels instead of performing an IP address lookup at each node. Increase in performance is obtained by the forwarding function being greatly simplified since layer 3 routing table lookup is bypassed. With the Ericsson MPLS implementation over ATM, ATM switching technology can be used as high capacity IP forwarding engines in label switching networks.

The label prepended to a packet as it enters the MPLS domain identifies uniquely the path the packet will traverse through the MPLS network. Classification and mapping of packets onto virtual paths can be performed based on information contained in the packet's layer-3 and layer-4 information that is accessible by the ingress node. Inside the core network, a packet is forwarded based exclusively on the contents of the label.

With the separation of layer-3 routing, this provides the foundation for the deployment of advanced traffic engineering features (e.g. explicit routing). It also allows enhanced security because of the following:

- The full IP packet can be encrypted since all the information required to forward the packet is in the MPLS label. Thus, the only part of the packet which remains unencrypted is the label.
- The label has only local significance, it does not convey any information on the packet's final destination.
- Other forwarding parameters (e.g. QoS parameters) can be binded to the label, without having to be explicitly coded in the packet's headers and therefore invisible to packet sniffers.

1.2 DIFFERENTIATED SERVICES

One of the today's most pressing challenges in designing IP networks is the provisioning of Quality of Service (QoS). The current Internet operates in a best-effort manner, which is considered insufficient for QoS demanding applications, e.g. video conferencing, and mission-critical transactions. In addition, as the Internet migrates to commercial enterprise, providing reliable QoS may well become a crucial factor in influencing the customer's propensity to pay for network services.

One approach to introducing QoS is *Differentiated Services* (DiffServ). DiffServ provides a framework in which QoS mechanisms can be developed such that "service differentiation" can be achieved for IP service classes. In practice, the basic idea is to classify IP streams into service classes and forward packets from each class differently.

DiffServ [8] is an IETF-driven paradigm for providing scalable QoS across an IP-based network. The conerstone of DiffServ is the usage of the Type of Service (TOS) byte field in the IP(v4) header. The basic concept is relatively straight-forward: edge nodes mark the TOS field with a particular Per Hop Behaviour (PHB) setting for packets inbound to the core network based on L3/L4 packet information. With the TOS field marked, nodes involved in the packets' forwarding can handle (queue/schedule/shape) the packets in a manner that provides service differentiation.

1.3 PROBLEM STATEMENT: COMBINING DIFFSERV AND MPLS

In Ericsson's implementation of MPLS, all packets are treated in the same best effort manner. The aim of this thesis is to design an MPLS system which supports Differentiated Services, providing packets with different forwarding treatments according to the classification and marking performed by the edge router.

Integrating the DiffServ approach into the MPLS system would produce a network able to provide paths to the same destination offering different service classes. Furthermore, the support of DiffServ by MPLS would allow an MPLS network to be a part of a DiffServ-compliant domain, guaranteeing the enforcement of the PHB.

The problem behind the integration of MPLS and DiffServ is that DiffServ nodes rely on examining the TOS field in the IP header (layer-3) to decide which forwarding treatment must be given to a packet. MPLS core nodes do not look at the layer-3 packet header to forward the packet (forwarding is only based on the MPLS label), and therefore cannot make an independent decision on which scheduling treatment must be given to each packet. Hence, mapping of layer-3 information to MPLS labels must be performed at the ingress MPLS node, by allocating different Label Switched Paths to different forwarding treatments.

By doing this, MPLS core nodes can be signalled that a specific label (i.e. Label Switched Path) maps to a certain destination subnet and forwarding treatment. When a packet arrives, the core nodes will look at the label prepended to the IP-header, perform a lookup in their routing table, and determine through which interface the packet is to be forwarded and which scheduling treatment it must receive. This information is recorded in the routing database when the LSP is established.

The problem addressed by this thesis is how these preferential LSPs should be managed, i.e. when and why they must be established and torn down, and how to coordinate this with users' requirements without being wasteful of resources.

1.4 CONTENTS OF THESIS AND ORGANIZATION OF REPORT

An overview of the current state of the Ericsson's DiffServ and MPLS systems is given in Section 2, at the same time providing the background necessary to understand the MPLS and DiffServ technologies. Section 3 addresses the question of the integration of both systems and introduces specifically the research problem. Three solutions are presented and one discarded, Section 4 and 5 explaining in detail the other two.

Section 6 presents a comparison analysis of both solutions and Section 7 discusses the choice of a solution and future work.

Appendix A in Section 8 provides a proof of concept, showing details of the implementation of the solution chosen.

2 REVIEW OF THE SYSTEM

This thesis considers a general network scenario like the one shown in Figure 1. A network provides connectivity between end users and access to the Internet. The network is made up of a connection oriented core (ATM) and IP routers (with ATM interfaces) at the edge.

Uppgjord (även faktaansvarig om annan) - *Prepared (also subject responsible if other)*		Nr - *No.*		
ETX/D/A Pablo Cebrian				
Dokansv/Godkänd - *Doc respons/Approved*	Kontr - *Checked*	Datum - *Date*	Rev	File
ETX/D/A Eric Lin		1999-10-25	PA1	report.fm

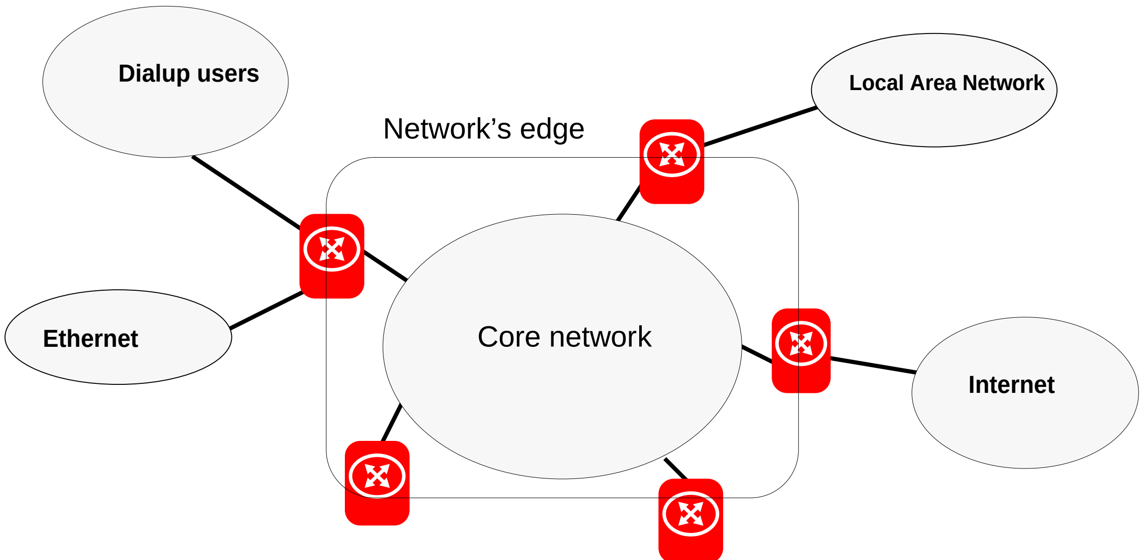


Figure 1 - General network scenario

The edge of the network (i.e. points of access for the users) is owned and administered by a Service Provider. The core connection-oriented infrastructure is usually shared with other ISPs.

The considered system presents the following capabilities:

- It is able to route IP packets in a Best Effort manner, thus acting as a normal IP network. This means that the core nodes have full TCP/IP routing capability.

- It can offer Quality of Service using IETF's Differentiated Services approach. Section 2.2 gives an overview of the DiffServ paradigm.

- It can forward packets along virtual paths through the core, examining the IP header only at the ingress and egress nodes, and performing label swapping in the core, according to MPLS. Section 2.1 provides a summary of the most relevant concepts of this technology.

- The considered network domain spans one routing or administrative domain and runs a common intradomain routing protocol. **In this thesis, we assume OSPF to be the intradomain routing protocol**.

The solutions described in this thesis are to be implemented in the routers located at the edge of the domain. It is assumed that the core supports the functionality described above and no further modifications are necessary.

2.1 MULTIPROTOCOL LABEL SWITCHING

Figure 2 illustrates the topology of an MPLS network. An MPLS *domain* is a connected set of nodes which operate MPLS routing and forwarding and which are also contained within the same routing and administrative domain. Edge MPLS nodes are called *Label Edge Routers* (LERs), and core MPLS nodes *Label Switching Routers* (LSRs). An MPLS edge node connects the MPLS domain with a node outside the domain which does not run MPLS or with an MPLS node which belongs to another routing domain.

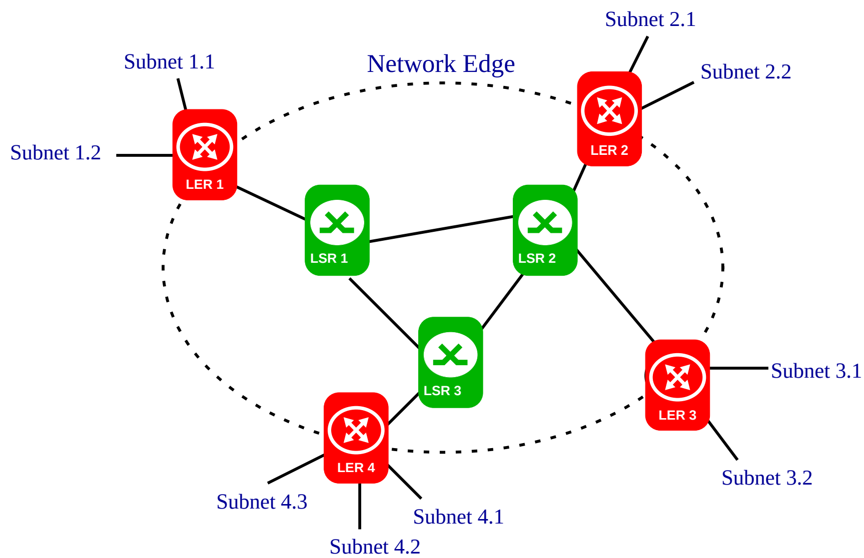


Figure 2 - An example configuration of a MPLS network

2.1.1 Labels and Label Switched Paths

A *label* is a short, fixed-length, locally significant identifier which is used to identify an MPLS virtual path. An incoming packet is assigned a label at the ingress node. The label is swapped at every core node and stripped off at the egress node, as the packet exits the MPLS domain.

The IETF's MPLS specification document [2] defines a 32-bit long MPLS label, which is prepended to the IP packet as shown in Figure 3 below.

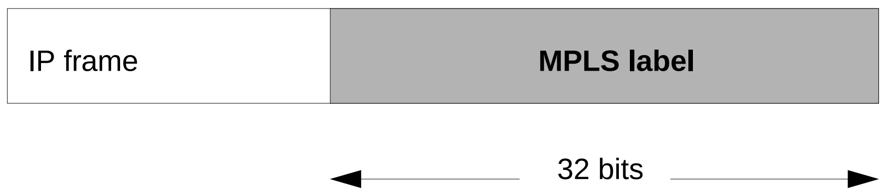


Figure 3 - MPLS label

A *Label Switched Path* (LSP) is a virtual path established through the MPLS core network defining an ingress to egress forwarding path. An LSP is uniquely identified by a label at the ingress LER. All packets prepended the same label will be forwarded along the same LSP and therefore follow the same path through the MPLS domain.

LSPs are simplex, i.e. labels are assigned considering the direction of the packet. There are no guarantees that a packet following an LSP from the ingress router LER1 to the egress point LER2 will follow the same path from LER2 to LER1, as it will not use the same LSP. Another LSP must be set up for packets going from LER2 to LER1.

2.1.2 Forwarding Equivalence Class

With a connectionless network layer protocol such as IP, each node of the network makes an independent forwarding decision for each packet it receives. Each router analyzes the incoming packet's IP header, performs a lookup in its routing table, and determines the next hop for the packet.

The routing module is composed of two different functional components, the control component and the forwarding component. The control component runs routing protocols to exchange information with other routers to build and maintain a forwarding table. When packets arrive, the forwarding component searches the forwarding table maintained by the control component and makes a routing decision for each packet.

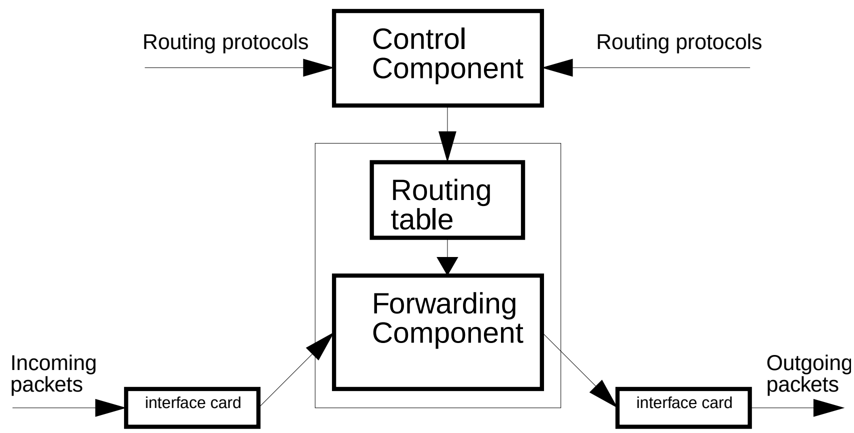


Figure 4 - Routing architecture

The forwarding function partitions the entire set of possible packets into a set of *Forwarding Equivalence Classes* (FECs), and maps each class to a next hop. The MPLS Framework document [1] defines an FEC as a "*group of layer-3 packets which are forwarded in the same manner (e.g. over the same path, with the same forwarding treatment)*". Packets which map to the same FEC are thus identically treated by the forwarding component, and will follow the same path (or the same set of paths, if multi-path routing is used) being given the same scheduling treatment at every hop. Hence, each FEC has at most one LSP assigned to it[i].

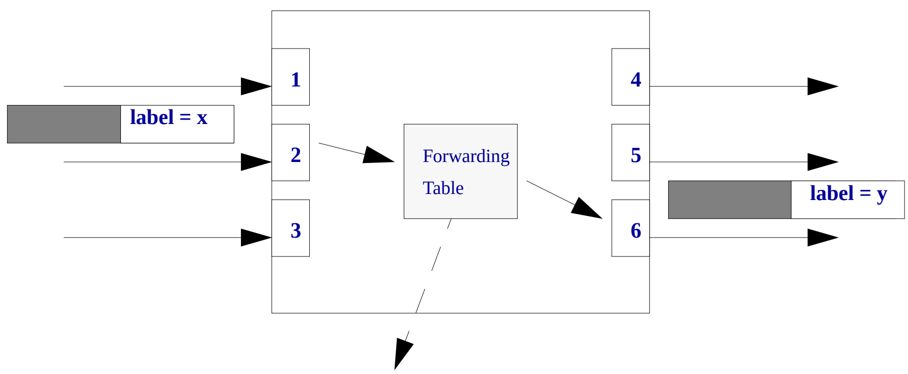
2.1.3 MPLS Forwarding process

Referring to Figure 2, when LER1 receives a packet, it performs a longest-match lookup on its routing table, maps the packet to an FEC and sends the packet out through the appropriate interface with a newly assigned label. In this case, since Ericsson's proprietary implementation of MPLS does not support provision of QoS, **an FEC equals an IP destination prefix**. The FEC to which a packet maps is determined by a longest-match lookup in the router's forwarding table to determine the IP prefix of the packet's destination.

In the core of the network, label switches ignore the packet's network layer header and simply forward the packet based on the label.

i. Note that multiple FECs *could* share the same LSP. Features like traffic aggregation or the case when two classes of service are treated equally by the MPLS domain are examples.

Figure 5 below shows how label-based switching works.When a packet arrives at a Label Switching Router, the forwarding component uses the input port number[ii] and label to perform an exact match search of its label forwarding table. When a match is found, the forwarding component retrieves the outgoing label and the outgoing interface, replaces the incoming label with the outgoing label and directs the packet to the appropriate outgoing interface.



label IN	Port IN	label OUT	Port OUT
x	2	y	6
a	1	m	4

Figure 5 - Label swapping as performed by MPLS interior nodes

LSRs will generally not look at the IP header of labelled packets and in fact can forward a non IP frame. Packets labelled with a pre-specified **default** label will have their network layer header examined to determine their destination, thus allowing normal IP routing to take place. See section 2.1.6 for further information.

When the labelled packet arrives at the last hop in the domain, the forwarding component does not find an outgoing label because the next hop is not an MPLS hop, and removes the current label, performing conventional IP routing to forward the packet.

---

ii. Using the input port number to perform a search for the outgoing port and label enables the router to use the same label in different interfaces, increasing the label range.

2.1.4 Signalling Protocol: Label Distribution Protocol

The set of procedures by which one MPLS node informs another of the label/FEC bindings it has made comprises the label distribution protocol. The MPLS architecture document [2] does not specify the use of a given signalling protocol. For the purpose of this thesis and in accordance with Ericsson's implementation, it will be assumed that the *Label Distribution Protocol* (LDP) [4] is used. This protocol was designed specifically to be used for MPLS signalling. Other protocols, like RSVP [29], are currently being used by other companies for distribution of labels and reservation of resources when using MPLS.

Two MPLS nodes which use LDP to exchange mapping information are known as LDP Peers with an LDP Session established between them. A single LDP session allows peers to learn of each other, i.e. the protocol is bidirectional.

There are four kinds of LDP messages:

1) Discovery messages - used to announce and maintain the presence of an MPLS node (be it LSR or LER) in a network.
2) Session messages - used to establish, maintain, and terminate sessions between LDP peers.
3) Advertisement messages - used to create, change, and delete label mappings for FECs.
4) Notification messages - used to provide advisory information and to signal error information.

Discovery messages provide a mechanism whereby LERs and LSRs indicate their presence in a network by sending the "Hello" message periodically. This is transmitted using UDP to the LDP port at the "all routers in this subnet" multicast address. When another LSR wishes to establish a session with another discovered LSP, it uses the LDP initialization procedure over TCP. Correct operation of LDP requires reliable and orderly delivery of messages. To satisfy these requirements LDP uses TCP for all message types except discovery messages.

2.1.5 Topology driven MPLS

In the first Ericsson proprietary implementation of LDP, the protocol is *topology driven*, and the set up and tearing down of LSPs are triggered by updates in the ingress edge router's routing tablesl[iii]. See Section 2.1.9 for examples of topology driven LSP setup and tearing down.

---

iii. Note that this does not need to be the case as other mechanisms can drive the setup/tearingdown of LSPs, e.g. traffic engineering tools.

This means that MPLS LSPs correspond to hop-by-hop forwarding paths that are determined by OSPF. Hence, FECs are defined solely by destination host/subnet addresses (entries in the ingress router's routing table). Two packets that result in the same lookup entry fall in the same FEC and therefore are assigned the same label.

2.1.6 Default Label

In order to be able to perform normal IP routing, the MPLS network uses a special default path, identified by the label **default.** Incoming packets carrying this label are passed up to the IP layer and routed by looking at their layer-3 destination address. The default label is also used for OSPF and LDP signalling traffic.

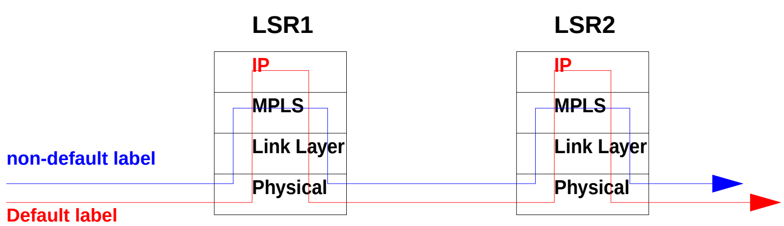


Figure 6 - Default-labeled packets are IP routed

All MPLS nodes perform IP routing on packets labelled **default.** Packets labelled in this manner do not follow a pre-established path, but are routed on a per-hop basis.

When the MPLS network is initialized, the default paths established among all other neighbouring MPLS nodes are used to transmit LDP information in order to establish LDP sessions between peers and set up LSPs, and used by OSPF to distribute routing information and to build routing tables at every node.

2.1.7 Label Information Base

Each edge node keeps information about its LSPs. This information is stored in the *Label Information Base* (LIB). Two such databases are kept, one for LSPs which originated at the LER, and another for LSPs which end at the LER. For the purpose of this thesis, only the LIB related to ingress functionality is relevant.

label	State	RT entries	CoS
34/21	WAITING	190.23.40	Not used
43/23	ESTABLISHED	223.30	Not used

Figure 7 - LIB example (relevant fields shown only)

The most relevant fields are:

- The **label** identifies uniquely an LSP at the ingress point.
- The **state** field contains information about the establishment phase of the LSP, an LSP is in WAITING state when it is being established.
- The **RT entries** field contains a list of the entries of the routing table which map to this LSP. For instance, all packets which after a longest-match lookup map to the IP prefix **190.23.40** will be labelled with label **34/21**. If no aggregation of routes is supported, each routing table entry will map to a different label, constituting an FEC by itself.
- The **CoS** field is not used by the current system. In an MPLS system with support for QoS features, it would contain the traffic service provided by the corresponding LSP.

When LDP is triggered to set up an LSP, it creates the LIB entry and it sets the **state** field to WAITING. Then, it issues a message requesting a label binding for the LSP to be established and waits for a response. Upon the reception of a positive response, the corresponding fields in the LIB and the routing table are updated.

2.1.8 MPLS over ATM

The label-swapping algorithm used by MPLS core nodes is identical to that used by connection-oriented switching networks such as ATM or Frame Relay. An ATM [26] switch forwards cells between interfaces based on the value of their VPI/VCI (Virtual Path Identifier/Virtual Circuit Identifier) field.

This thesis takes Ericsson's MPLS implementation as a reference and **considers the switching core to be ATM-based**. Ericsson's AXI 512 Edge Routers are used as Label Edge Routers and AXD 301 ATM Switches as core nodes.

When using ATM as a core technology for an MPLS network [27], the following applies:

- ATM breaks IP packets into fixed length cells and forwards these through the network by looking solely at the cell header's contents.

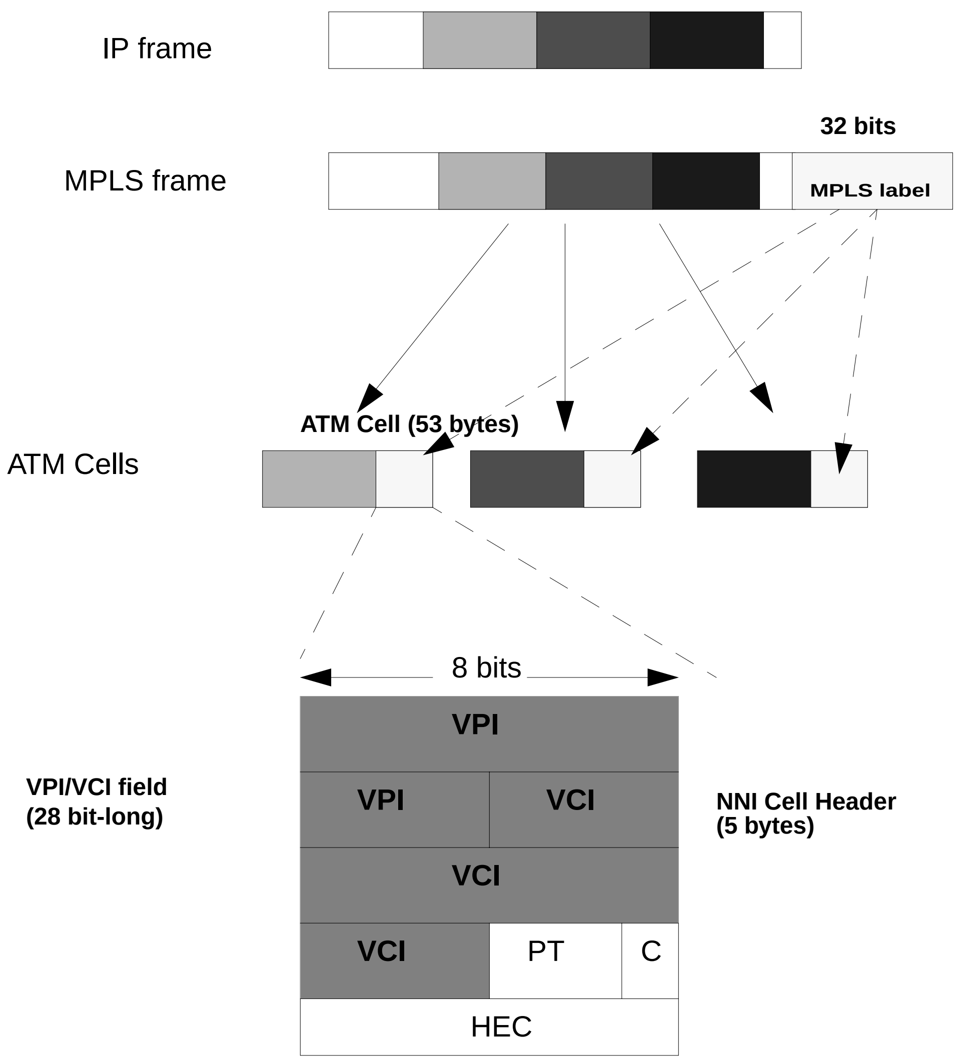


Figure 8 - ATM operation

- Labels are encoded into the VCI/VPI field of the ATM cell header (see Figure 8). This implies that the 32-bit long IETF-defined MPLS label is encoded into an 28 bit long field. Hence, in MPLS over ATM implementations, the four least significant bits of the MPLS label are 0 (see Figure 9 below).

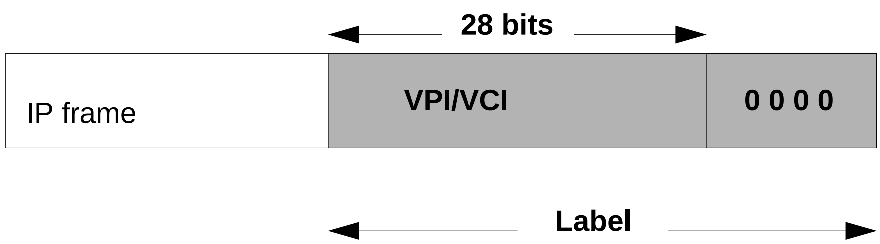


Figure 9 - Encoding of the label in MPLS over ATM implementations

- Labels are still stored as 32-bit long in order to comply with the MPLS specification.
- ATM switches forward fixed length cells swapping VPI/VCI values as LSRs swap labels.
- ATM Permanent Virtual Circuits act as Label Switched Paths. The number of PVCs a node can handle is limited by the hardware. Reasonable values are 2000 PVCs for edge routers and 32000 for core nodes.
- ATM switches have IP router capability. Cells coming in through the **default** path (labeled as **0/32**) are recombined into an IP packet and the packet is passed up to the IP layer.
- Label binding is handled by the MPLS signalling protocol and no ATM signalling is necessary for the operation of MPLS.
- All MPLS cells are treated as Unspecified Bit Rate traffic. Ericsson's first implementation does not contemplate the use of bandwidth reservation for MPLS connections.

The fact that an ATM switching core is used imposes one important design constraint: the label must be encoded in the 28 bit VPI/VCI field in the cell header, and that is all the information LSRs can rely on to perform switching. As it is shown in Section 3.1.3, this means that to provide different forwarding treatments, the binding **{label, destination host/subnet}** must be enhanced to **{label, destination host/subnet, forwarding treatment}**, since there is no CoS field present in the cell header. This means that the way the system interprets an FEC matches the original definition given in the introduction: a "*group of layer-3 packets which are forwarded in the same manner (e.g. over the* *same path, with the same forwarding treatment)*". **An FEC is defined now by the destination host/subnet of the packet and the DiffServ forwarding treatment**.

2.1.9 Examples of MPLS usage

2.1.9.1 Establishment of an LSP

The following steps show the setup procedure for an LSP to subnet 2.1, as shown in Figure 2. The label distribution procedure shown is called *downstream on demand ordered control* and is specified in the LDP specification [4]. This mechanism is being used by Ericsson's and most early MPLS implementations.

The setup procedures are triggered by the addition of a new subnet connected to the edge of the MPLS domain. This subnet will be advertised using a routing protocol (e.g. OSPF) by the edge router to which the subnet is connected, and the routing tables of all nodes in the network will be updated. Only ingress edge nodes will start the signalling necessary to set up LSPs to the new destination. How this is accomplished follows:

1. LER1's routing protocol learns about a new subnet (2.1) and updates the routing table.

2. The routing table recognizes that an LSP does not exist for this destination and triggers LDP to set up a new LSP.

3. LER1 creates the LIB entry for the LSP and the **label** field is initialized as **default** both in the LIB and in the routing table. Packets using this LSP before it is established will be routed using IP routing along the default path. The **state** field in the LIB entry is set to **WAITING**, and a timer is started. If no response is received in a given time, the LER will issue another label request message.

4. LER1 sends an LDP message requesting a label for subnet 2.1 to its next hop router for this destination - i.e. to LSR1.

5. LSR1 forwards the request message for subnet 2.1 to its next hop for destination 2.1-i.e. to LSR2, which in turn forwards it to LER2.

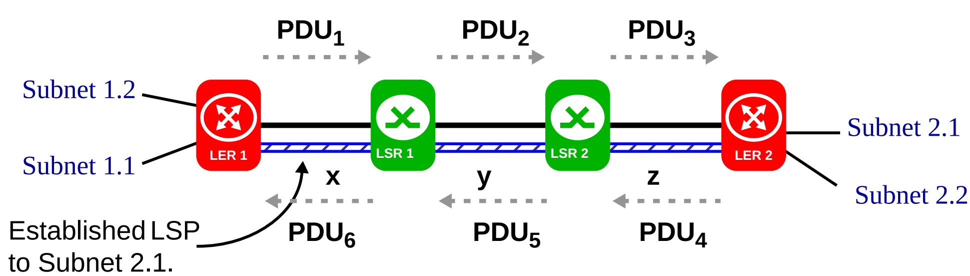
6. Upon reception of the message requesting a label, LER 2 allocates a label (label **z**) to be used by LSR2 when forwarding packets for subnet 2.1, and sends back a message specifying the label binding. Therefore, all packets from LSR2 to LER2 for subnet 2.1 will now use the label **z**.

7. LSR 2 and LSR1 similarly allocate labels (label **y** and **x** respectively) to be used and send corresponding messages reporting the binding.

8. When LER1 receives the message informing of LSR1's label binding, the LSP is established. It updates the LIB entry and the routing table with the new label. LER1 will mark all subsequent packets destined to subnet 2.1 with label **x** and forward them to LSR1.

8. When LSR1 receives packets with label **x**, it will simply "swap" label **x** with label **y** and forward the packets and so forth until the packets reach LER2.

9. LER2 will strip off the label and perform standard IP routing to forward the packet.



PDU_1 LABEL_REQUEST message for destination Subnet 2.1
PDU_2 LABEL_REQUEST message for destination Subnet 2.1
PDU_3 LABEL_REQUEST message for destination Subnet 2.1

PDU_4 LABEL_BIND message, use label **z** for Subnet 2.1
PDU_5 LABEL_BIND message, use label **y** for Subnet 2.1
PDU_6 LABEL_BIND message, use label **x** for Subnet 2.1

Figure 10 - LSP setup procedure

2.1.10 Failure to establish an LSP

It is possible that LDP will not manage to set up an LSP for a given FEC. Possible causes to this are link failures, out-dated routing information, a host momentarily unreachable, network congestion, packet loss, and so forth.

LDP starts a retry timer when the proceedings leading to the setup of an LSP begin. If a label binding message has not been received after a certain period of time, the binding request is re-issued. The timer has an exponential back-off factor, and after a certain number of attempts, LDP will finally quit, erasing the corresponding LIB entry.

While LDP tries to set up an LSP for a given FEC, packets destined for that FEC will be labelled **default** and routed on the basis of their network layer header.

2.1.11 Removal of an LSP

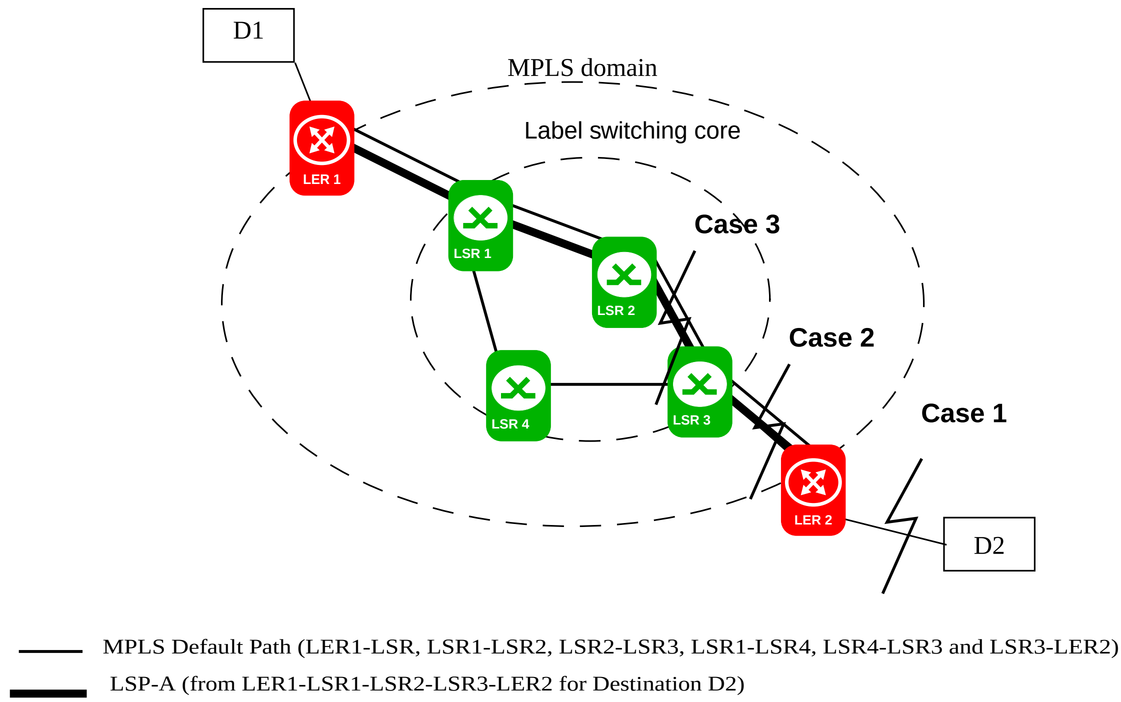


Figure 11 - Reference network for a link failure. Three cases

Figure 11 shows three network scenarios which will cause the removal of routing table entries and their corresponding Label Switched Paths.

2.1.11.1 Case 1: Destination D2 becomes unavailable

In this case, the routing protocol will remove the entry corresponding to D2 from all routing tables in the domain. Since D2 is the only destination using LSP-A, this LSP becomes unnecessary and must be removed.

LER2 will deallocate resources dedicated to LSP-A and send a message informing of the label release to LSR3 which will be forwarded along the path, freeing all resources allocated within the network.

2.1.11.2 Case 2: The link between LER2 and LSR3 goes down

Either the routing protocol will learn that the link is down and will remove all routes through the failed link or else the LDP session established between LER2 and LSR3 will time out. Once alarmed, LDP will take actions to remove all LSPs traversing such a link.

2.1.11.3 Case 3: The link between LSR2 and LSR3 goes down

Since there exists an alternative route to destinations D2 and D1 (through LSR4), it will be assumed that no LSPs will be taken down. The switching core will update LSR1 and LSR2 routing tables and perform a “local repair”, routing LSP A through the alternative route. If there existed no alternative path, all LSPs along the link would be torn down.

2.1.12 Aggregation

Aggregation is the ability of the network to use the same LSPs for every connection within the same ingress and egress points. If aggregation is supported, several destination subnets will map to the same Forwarding Equivalence Class, and therefore use the same label.

Referring to Figure 2, a system supporting aggregation would imply that Subnets 2.1 and 2.2 would be assigned the same labels along the path from LER1 to LER2, and thus any packet going from LER1 to LER2 would use the same LSP, regardless of whether its destination is Subnet 2.1 or 2.2.

For this thesis, it is assumed that aggregation is not supported.

2.2 DIFFERENTIATED SERVICES

The Differentiated Services paradigm aims at providing QoS on a hop-by-hop basis. Traffic entering a DiffServ-compliant network is classified into different behaviour aggregates. A behaviour aggregate is comprised of all packets which request the same forwarding treatment. Each hop within the network identifies the behaviour aggregate that incoming packets belong to, and tries, to the best of its capability, to provide the appropriate scheduling treatment, or Per Hop Behaviour.

Each PHB is identified by a Differentiated Services codepoint, encoded in the first six bits of the IPv4 Type of Service field or in the Traffic Class octet in the case of IPv6 (see RFC2474). This field is called the DiffServ-byte by the IETF DiffServ working group.

Version	IHL	Type of Service	Total Length
Identification		Flags	Fragment Offset
Time to Live		Protocol	Header Checksum
Source Address			
Destination Address			
Options			Payload

Figure 12 - IP Header (TOS field shadowed)

2.2.1 DiffServ compliant node architecture

A DiffServ compliant node or DiffServ node is a router able to apply different forwarding treatments to packets, i.e. able to provide different PHBs based on the packet's DiffServ codepoint value.

A DiffServ domain is a connected set of DiffServ nodes which operate with a common service provisioning policy and a set of PHB groups implemented on each node. A DiffServ domain has a well defined boundary consisting of DiffServ boundary nodes which classify and possibly condition ingress traffic to ensure that packets which transit the domain are appropriately marked to select a PHB from one of the PHB groups supported within the domain. Both boundary and interior nodes must be able to apply the appropriate PHB to a marked packet but only edge nodes perform classification and marking of packets.

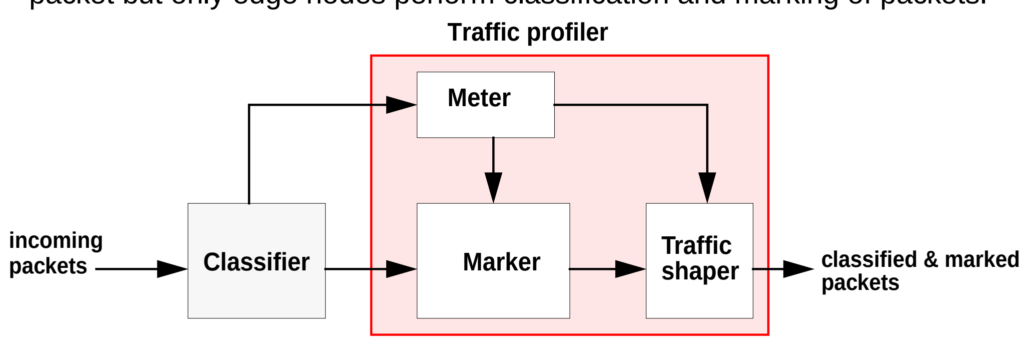


Figure 13 - Architecture of a DiffServ boundary node

Figure 13 above shows the general architecture of a DiffServ boundary node. The packet classifier steers packets matching some specified rule to the traffic conditioner. Two types of packets classifiers are defined in the DiffServ architecture (RFC 2475), the Behaviour Aggregate Classifier classifies packets based on the DiffServ codepoint only. The Multi Field classifier selects packets based on the value of a combination of one or more header fields. The classifier implemented in a boundary node, might take into consideration other information about the connection or the user to classify the packet.

The traffic marker is closely associated with the classifier, and ensures that the DiffServ field or packets at an access point is set appropriately. Meters, policers, and shapers measure and verify that the incoming traffic meets certain requirements, and that outgoing traffic conforms to service agreements with other DiffServ domains.

2.2.2 Service Level Agreement

According to the Differentiated Services Framework [8], a service is an *"overall treatment of a defined subset of customer's traffic within the DiffServ domain"*. **It could be said that a service is defined by the combination of all the forwarding treatments the packet will receive along its path**.

Providers and customers negotiate agreements with respect to the services to be provided by the provider to the user. This agreements take the form of Service Level Agreements, which specify the overall features and performance which can be expected by a customer purchasing a given service.

2.2.3 Services supported

Three user traffic classes are supported in Ericsson's implementation of DiffServ:

- **Best Effort (BE)** - standard default service.

- **Assured Service (AS)** - aimed to have a better assurance of timely delivery of data than the BE service.

- **Premium Service (PS)** - intended to be used where low delay, jitter, and probability of loss are required (e.g. delay-sensitive real-time applications). This service is expected to be expensive and with a limited number of simultaneous subscribers, thus avoiding the degradation of the service to Best Effort.

How the enforcement of different forwarding behaviors is to be implemented is not specified by the IETF DiffServ specification [10]. In the system considered by this thesis, each service class is assigned a different queue in every network interface. Treatment of incoming packets is handled using algorithms like Random Early Detection (RED) and RIO [12]. Once a packet has been classified and properly queued a customized Virtual Clock is used to schedule among the

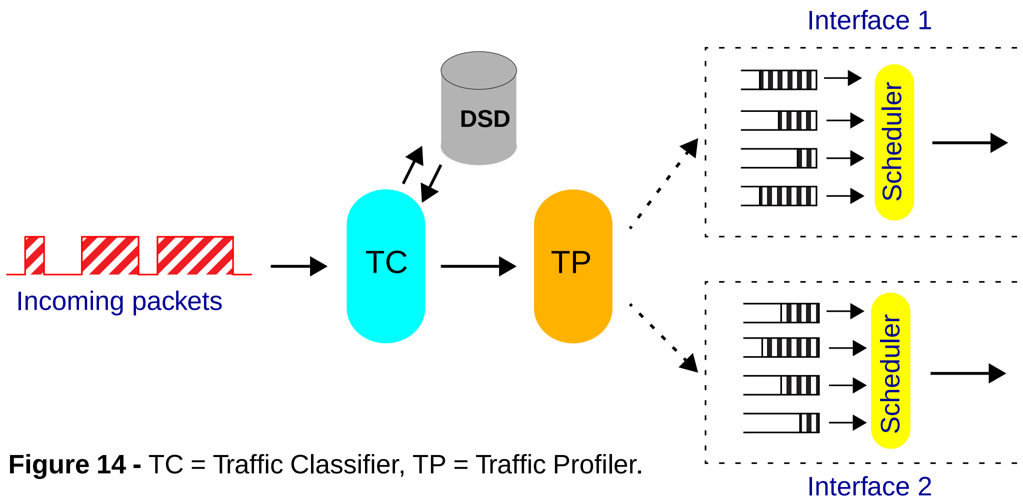


Figure 14 - TC = Traffic Classifier, TP = Traffic Profiler.

three queues with different priorities. Figure14 illustrates how packets are classified and queued accordingly.

2.2.4 DiffServ database

Users purchasing services from a Service Provider specify their preferences in terms of forwarding treatments to be given to different destinations and protocols, this information is stored in the Differentiated Services database. Figure 15 illustrates how the DiffServ database can be organized and what kind of information is used to determine the forwarding treatment to be given to a packet.

User ID	221.1.28.14			221.1.28.18
Appl. Type	Telnet	FTP	SMTP	Telnet
Dest. Address	221.1.29.1	123.223.211.1	130.40.207.x	191.20.45.x
DS Class	PS	AS	AS	PS

Figure 15 - Example of a DS database

When packets arrive at the DiffServ classifier the information contained in the DiffServ database is used to assign a PHB to different connections. It will be assumed that a multi-field classifier uses information from the IP header to find an exact match in the DiffServ database, coming up with the appropriate PHB. If multiple matches are possible, the highest class match (PS over AS) is used.

2.2.5 Walk-through

Recalling Figure 14, as an IP packet enters the edge DiffServ node, it is first classified by the *Traffic Classifier* (TC), which:

- reads the necessary fields in the IP header
- performs an exact match lookup in the DiffServ Database
- determines the PHB to be assigned to the packet
- marks the packet accordingly (by coding the PHB into the DiffServ codepoint in the DiffServ byte)
- passes the packet to the *Traffic Profiler* (TP).

The TP module is responsible for traffic flow metering and accounting. From TP, packets are routed to the appropriate output interface. Each interface maintains each own set of queues and delivers the appropriate scheduling treatment according to the PHB assigned to the packet.

In an intermediate node there is no packet classification or traffic conditioning, the DiffServ module looks at the value of the DiffServ byte and assigns the packet to the correct queue in the appropriate outgoing interface.

3 MPLS AND DIFFSERV

Both MPLS and DiffServ perform classification of packets once at the edge of the domain. The results of this classification are respectively encoded in the MPLS label prepended to the IP-header, and the DiffServ-byte in the layer-3 packet header.

Core nodes do not perform any further classification, and simply look at either the label or the PHB to treat the packet according to the way it was initially classified.

The problem is that, since both technologies mark packets at different layers, MPLS core nodes are not able to read the results of the DiffServ classification because they will not inspect the contents of layer-3 headers. In order to overcome this and achieve a working implementation of MPLS with DiffServ support, the information provided by the label to the core switches must be enhanced, mapping a label (i.e. an LSP) to a forwarding treatment as well as a destination subnet. That is, each FEC (defined by a destination IP prefix and a forwarding treatment) maps to one LSP, which is uniquely identified by a label.

3.1 DIFFSERV SUPPORT IN MPLS

An LER maps incoming packets to Forwarding Equivalence Classes, which are serviced by LSPs identified uniquely by the ingress label. An FEC defines the path the packet will follow through the network and the forwarding treatment it will receive at every node along such path.

In the current system (i.e. MPLS with no DiffServ support) an FEC is equivalent to an IP subnet prefix, and every routing table entry at the ingress router constitutes a different FEC. Interior nodes perform forwarding of packets based only on the label coded in the VPI/VCI field of the cell header. Using LDP, labels are bound to destination subnets, and an LSP is established as a set of labels. All packets with the same destination subnet will be forwarded along the same LSP.

In an MPLS system with DiffServ support, an FEC is defined both by the destination subnet and the PHB assigned to the packet by the DiffServ Classifier. Therefore **two packets with the same destination IP address but classified differently by the DiffServ module will map to two different Forwarding Equivalence Classes and hence two different LSPs**.

3.1.1 LSP allocation without DiffServ support

In an MPLS system which does not support DiffServ, one LSP is allocated to each IP subnet prefix in the edge router's routing table. This means that all packets belonging to the same FEC (i.e. following the same path through the network) are labelled in the same way.

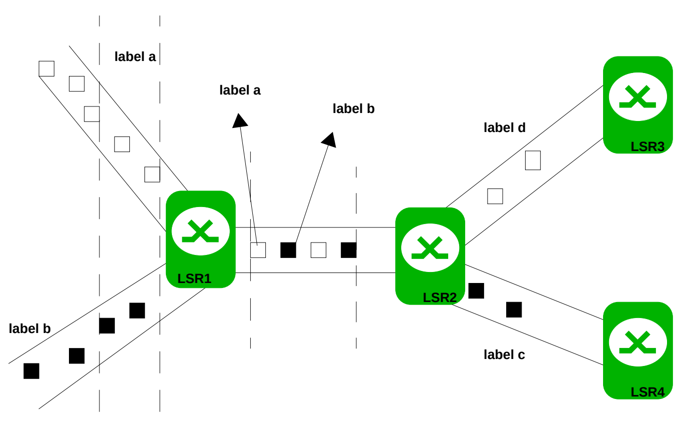


Figure 16 - Conventional MPLS

Figure 16 illustrates how cells belonging to two different connections (i.e. carrying different labels), enter LSR1 through two different interfaces, and exit through the same one. LSR1 puts the cells in the outgoing interface queue in

FIFO manner. When cells arrive at the next LSR, they are directed towards different outgoing interfaces by looking at their labels.

3.1.2 Why are new LSPs needed for DiffServ support?

Edge routers mark the DiffServ byte of incoming packets and direct them to the correct queue in the outgoing interface. Packets come out of an Edge Router in the order dictated by their PHB.

Let us assume that all packets destined for the same subnet are assigned the same label, regardless of their PHB. In Figure 17, packets come out of LER1 and LER2 correctly ordered, and enter the LSR through different interfaces, exiting through the same outgoing interface. Both LERs have access to the contents of the IP header, and can use the value of the DiffServ field to determine how outgoing packets must be scheduled to provide DiffServ.

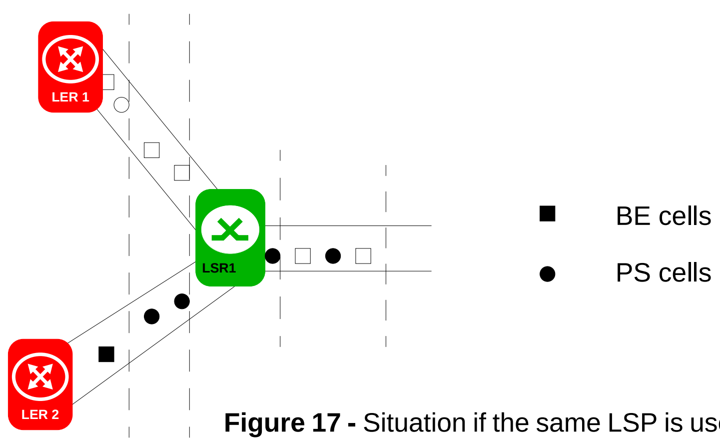


Figure 17 - Situation if the same LSP is used for all PHBs

However, the network layer header of incoming cells is invisible to LSRs, since the forwarding performed by these is based solely on the label. LSR1 cannot distinguish which packets should be assigned which forwarding behaviour, because all packets from the same connection carry the same label. Therefore, it will simply choose in a round robin manner between both connections, not providing any special treatment to PS packets over BE packets.

The conclusion is that **separate LSPs are needed to provide support for the different DiffServ classes at the core network**.

3.1.3 LSP establishment

Since an LSP now maps to a destination subnet and a PHB (i.e. to an FEC), both must be specified when setting up the path. This is accomplished by using a Class of Service field in the label request messages. A label is now requested for a certain destination IP prefix and a certain forwarding treatment. The core switches map an incoming label to an outgoing label and interface and a scheduling behaviour.

3.1.4 Modification of the routing table

In the MPLS system without DiffServ support, one routing table entry mapped to one single FEC and therefore was assigned only one label. When DiffServ classes are supported, each entry maps to as many FECs as services supported (three in this thesis).

The routing table is modified to accommodate the new labels, as shown in Figure 18 below.

If the LSP corresponding to a given FEC has not been established, is under establishment or could not be established, the label present in the appropriate routing table field will be **default.**

Host/Subnet	Next hop	BE	AS	PS
221.1.29	LSR1	label a	label c	label d
193.4	LSR2	default	default	default

Figure 18 - Two new fields are added to the routing table

3.1.5 Walk-through

How is a packet forwarded in an MPLS system with DiffServ support? The packet's path through the system can be broken down in three stages: how it is treated as it enters the domain, what happens inside the domain, and how it is delivered by the edge egress node, as it leaves the MPLS domain.

3.1.5.1 Ingress edge router

When an IP packet comes into the MPLS-DiffServ domain, the following steps take place at the ingress router:

- The DiffServ module at the edge router determines the PHB to be assigned to the packet and marks the DiffServ-byte. It does this by doing an exact match lookup on the DiffServ database. Assume, for instance, that the source IP address of the incoming packet is **221.1.28.14**, and the destination port is the **telnet** server port. The match found could look like the entry in Figure 19.

User ID	**221.1.28.14**
Appl. Type	**Telnet**
Dest. Address	**221.1.29.1**
DS Class	**PS**

Figure 19 - Example of a DSD entry

- The PHB to be assigned to the packet is Premium Service (**PS**). The DiffServ module marks the DiffServ byte in the packet's IP header with the corresponding DiffServ code point.

- The IP module performs a longest-match lookup on the packet's destination address and determines the routing table entry to which the packet's destination IP address maps to. The FEC the packet belongs to is completely known now, being defined by the routing table lookup and the DiffServ classification outcome.

Subnet	Next hop	BE	AS	PS
221.1.29	**LSR1**	**a**	**c**	**d**

Figure 20 - Each subnet maps to three labels

- This routing table entry maps to three labels (see Figure 20), one per PHB supported by the system (BE, AS, PS). The correct label is determined by taking into account the contents of the DiffServ byte in the IP header. Referring to Figure 20, the packet is labeled with **label d**.

- The packet is passed down to the link layer. The appropriate link layer header with the packet's label encoded is attached and the packet is put in the queue corresponding its PHB in the outgoing interface

3.1.5.2 Core LSR mesh

In the middle of the LSR cloud, the intermediate LSR only look at the incoming VPI/VCI of an incoming cell to determine:

- Outgoing interface
- Outgoing label
- Outgoing scheduling queue[iv]

3.1.5.3 Egress edge router

When the packet arrives at the egress LER, the label is stripped off and normal IP routing is performed. The egress LER can still read the value of the DiffServ codepoint and enforce the appropriate Per Hop Behaviour.

3.2 RESEARCH QUESTION

3.2.1 Problem statement

In order for a packet to be forwarded along a Label Switched Path, this path must exist. Best Effort paths providing connectivity are established at system boot-up, as the routing protocols build the routing tables in the edge routers. Setup and tearing down of these BE LSPs is completely topology driven. LSPs offering other services (preferential LSPs, non-BE LSPs) must also be allocated and deallocated to service packets from users requesting and paying for preferential treatments. **The problem addressed by this thesis is how the non-BE or preferential LSPs should be managed**.

3.2.2 Proposed solutions

Three feasible solutions to non-BE LSP management problem are presented in the following sections. In all of them, the management of BE LSPs providing basic connectivity is topology-driven. Each solution is briefly described and evaluated.

---

iv. ATM switches can now support DiffServ by performing VC scheduling

3.2.2.1 Topology driven LSP management

One solution is to establish and tear down PS and AS LSPs when BE LSPs are established and torn down. This means that there will be three times more LSPs than in the current system without DiffServ support.

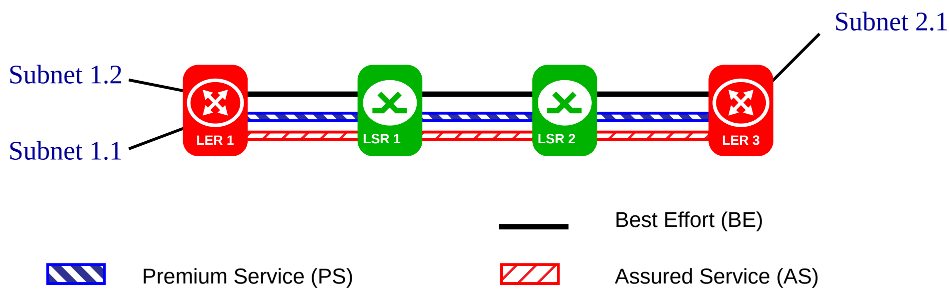


Figure 21 - Three new LSPs are established to LER3

With this solution, all three label fields present for every entry of the routing table will map to a valid LSP or all three will be **default.**

This solution is easy and straight-forward to implement, but presents serious scalability and network utilization problems. Many of the LSPs established from each Edge Router will never be used, but reduce available network resources[v] anyway, besides wasting the label space and taking up memory. As the number of services supported and destinations attached grows so will the number of static LSPs, independently of traffic requirements.

This solution is discarded in this thesis, which aims to achieve a more efficient management scheme, less wasteful of network resources.

3.2.2.2 user profile driven LSP management

With user profile driven LSP management, the contents of the DiffServ database determine the network configuration. When the system is started the DiffServ database is empty. Entries are added as users connect to the network and their DiffServ profiles are downloaded from a central repository on to the edge router.

v. The AXI 512 ATM interface supports only up to 2000 VPI/VCI pairs.

At system boot up, after OSPF has built all routing tables and LDP set up all corresponding LSPs, and before any user has been connected to the network, all edge router's routing table entries look like the ones shown below:

Host/Subnet	Next hop	BE	AS	PS
221.1.29	LSR1	label a	default	default
193.4	LSR2	label b	default	default

Figure 22 - Routing table at system boot-up

BE LSPs have been set up for all routing table entries, to provide connectivity to all destinations connected to the MPLS network edge. Management of BE LSPs remains topology driven, and the setup or tearing down events are triggered by OSPF updates on the egress router's routing table.

When an entry is added to the Edge Router's DiffServ database, its contents are expressed in terms of FECs to which packets sent by the subscriber of the entry will map. If the LSPs servicing those FECs do not exist, they are set up. It must be underlined that an FEC in an MPLS system with DiffServ support is defined by a destination subnet and a forwarding treatment.

An LSP is torn down when, after the removal of an entry from the DiffServ Database, it turns out that the LSP is not being needed by any subscriber.

3.2.2.3 Traffic-driven LSP management

When using Traffic driven LSP management, the system will not set up any LSPs in advance. At boot-up, BE LSPs will be set up for all BE-FECs in a topology driven way (i.e. as entries are added to the routing table by OSPF), and the **label** fields corresponding to AS and PS will be set to **default**. The routing table will look like the one shown in Figure 22.

When the system's forwarding function retrieves the label corresponding to an FEC from the routing table, two results are possible:

- A label corresponding to a valid non-BE LSP is retrieved. The packet can be forwarded according to its FEC, that is along an LSP providing the appropriate PHB and leading to the appropriate egress point.

- The **default** label is retrieved. This implies that there is not an LSP servicing the corresponding FEC. There are three possible causes:

1.- The LSP is under establishment. The LDP message specifying the binding {FEC, label} has not been received yet and the label field in the routing table has therefore not been updated.
2.- The LSP's establishment was unsuccessful (e.g. due to a link failure). LDP sets a retry timer with an exponential back-off factor. After a certain period of time it stops trying to set up the LSP.
3.- The LSP was torn down (e.g. due to a topology change or to not being needed longer) and not re-established.
4.- There was never an attempt to establish the LSP and therefore the corresponding routing table field stays as initialized (i.e. **default**)

An LSP under establishment has a LIB entry associated with it, and the **state** field of the entry is set to **WAITING.** In cases 2-4 the LIB entry does not exist, since it has either been removed (cases 2 and 3) or was never created (case 4).

The setup of LSPs will be triggered by incoming packets which are classified into an FEC for which there is not an LSP set up or under establishment. For instance and referring to Figure 22, a packet with destination address **221.1.29.45** and marked by DiffServ as PS will map to the FEC **{221.1.29, PS}** which has a **default** label assigned to it. If the LSP is not under establishment (i.e. the LIB entry does not exist), LDP will be instructed to set it up.

A timer is associated to each LSP, and reset every time a packet is sent along that LSP. When a certain time elapses without the LSP being used, it times out and is removed.

4 USER PROFILE DRIVEN MANAGEMENT SOLUTION

4.1 GENERAL DESCRIPTION

In order to be forwarded along a non-BE LSP that has already been set up, a packet must be labelled properly. The label is chosen taking into account the routing table entry the packet's destination address maps to and the results of the packet classification performed by the DiffServ module.

The contents of the DiffServ database can be expressed as a set of FECs. Each destination address specified maps to a routing table entry and has one or several PHBs associated with it. Thus, the database can be translated into a set of non-BE LSPs needed to service packets sent by the subscribers. The user profile driven solution keeps an updated "translation" of the DiffServ database in terms of FECs (i.e. requested LSPs, since one LSP is allocated per FEC) and ensures that the LSPs necessary to service those are established and remain established as long as they are likely to be used by any subscriber. This translation takes place every time the DiffServ database is modified either by deleting or adding an entry.

A detailed description of the implementation of this solution can be found in Appendix A.

4.2 MODIFICATIONS TO THE SYSTEM

The MPLS system providing DiffServ support must be modified and expanded to be able to manage the allocation of non-BE LSPs. Apart from a new control module, serving as an interface among the different involved modules, existing data structures must be enhanced to keep new necessary state information.

4.2.1 New interface module

A new module serves as an interface between the LDP and DiffServ modules. The interface module contains the criteria determining when non-BE LSPs should be established or removed and it includes the functionality necessary to perform the operation of expressing the contents of the DiffServ database DiffServ in terms of FECs.

The new module responds to events generated by DiffServ and LDP and runs a different algorithm, depending on the incoming event, to generate three possible responses:

- The module does nothing but update state information kept on users and LSPs
- The module instructs LDP to start the proceedings to set up an LSP for a given destination and with a given quality of service and updates state information accordingly.
- The module instructs LDP to tear down an existing LSP and updates state information as needed.

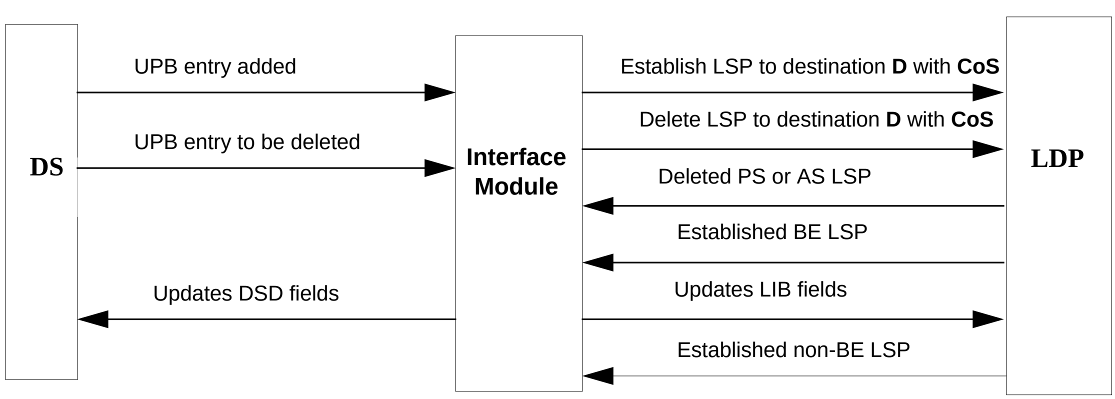


Figure 23 - Module communication

Figure 23 shows the interaction among the three modules. The interface module has access to the data structures managed by DiffServ and LDP and updates them as necessary.

4.2.2 New field in the DiffServ Database

Figure 24 shows the new field which needs to be added to the DS database. A linked list of LSPs used by each user is necessary to keep track of which LSPs must be examined for deletion if the User Profile Entry is deleted.

User ID	221.1.28.14		
Appl. Type	Telnet	FTP	SMTP
Dest. Address	221.1.29.1	123.223.211.1	130.40.207.x
DS Class	PS	AS	AS
LSPs used	Linked list of pointers to LSPs servicing the FECs the user has subscribed to		

Figure 24 - New field added to the DS database

4.2.3 New field in the Label Information Database

Figure 25 shows the new field that must be added to every LIB entry. A list of users needing the LSP is linked off each LIB entry in order to determine who is needing of the existence of the LSP when it is inspected for deletion. **An LSP will be deleted only if there are no users requesting it**.

label	State	FECs	CoS (PHB)	List of users
34/21	Waiting	190.23.40	PS	

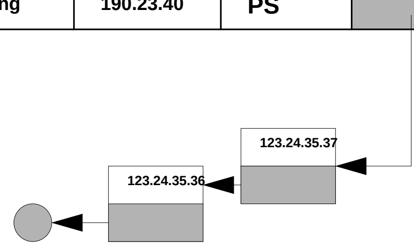


Figure 25 - LIB example (relevant fields shown only)

Uppgjord (även faktaansvarig om annan) - *Prepared (also subject responsible if other)*		Nr - *No.*		
ETX/D/A Pablo Cebrian				
Dokansv/Godkänd - *Doc respons/Approved*	Kontr - *Checked*	Datum - *Date*	Rev	File
ETX/D/A Eric Lin		1999-10-25	PA1	report.fm

4.3 SYSTEM EXAMPLES

When an event generated by the LDP or DiffServ modules is received by the interface module, it triggers the execution of an algorithm which determines the response of the module to the event.The events passed by LDP are:

- Removal of a non-BE LSP
- Establishment of BE LSPs
- Establishment of non-BE LSPs

The events passed by DiffServ are:

- Addition of a new user profile to the DiffServ database
- A user profile is to be deleted from the DiffServ database

The following sections detail the logic of the response of the interface module to these events.

4.3.1 Scenarios triggered by updates in the DiffServ database

4.3.1.1 Addition of a DiffServ entry

Event:

The addition of a new DiffServ entry means the addition of a new authorized user to the network. This user subscribes to traffic services specified in his user profile in the DiffServ database DiffServ. These preferences might require the establishment of new non-BE LSPs.

Actions:

The interface module will parse the new user profile entry and will determine which LSPs are needed to service the user requirements. It will instruct LDP to set up those LSPs which are needed and do not already exist. A more detailed explanation of this algorithm can be found in Appendix A, Section 9.3

Example:

User ID	**221.1.28.14**
Appl. Type	**Telnet**
Dest. Address	**221.1.29.1**
DS Class	**PS**
List of LSPs	**Empty**

Figure 26 - Entry added

Let us assume that the entry shown in Figure 26 is added to the DiffServ database. After the interface module is notified, the following actions take place:

- The entry is parsed to determine which LSPs are necessary to forward packets belonging to the subset defined by the user preferences. In this case, packets from the user **221.1.28.14** destined to the Telnet port in **221.1.29.1** must be assigned Premium Service. This means that an LSP must be established to destination **221.1.28.14** providing **PS**.
- The destination address **221.1.29.1** is used to perform a lookup in the routing table, determining the destination IP prefix the packet belongs to, **221.1.29/24** for instance.
- The "translation" of the entry is now complete.The Interface Module interprets the entry as the **FEC {221.1.29/24, PS}**.
- Is there a PS LSP already assigned to that FEC? If there is not, then instruct LDP to set it up. Otherwise continue.
- Add the user ID to the **list_of_users** contained in the LSP's LIB entry.

4.3.1.2 Imminent deletion of a User Profile entry

Event:

A user profile entry is to be deleted, the DiffServ module notifies the DiffServ/LDP interface module before deletion.

Actions:

The Interface Module will check whether the non-BE LSPs included in the list linked off the UPB entry are still needed, if not, they will be removed. This is easily done by checking the new **list_of_users** field in the LIB entries of every LSP included in the **list_of_lsps** included in the DiffServ database entry.

Example:

Let us assume the entry used in the previous section has been added and is now going to be removed. The Interface Module needs to be notified before removal is effective because it needs access to the information contained in the **list_of_lsps** field in the entry to know which LSPs are affected by the deletion of the entry.
After notice, the following actions follow:

- Every LSP contained in the **list_of_lsps** is checked, and the User ID corresponding to the entry to be removed is erased from the linked list of users linked off the LIB entry.

- Are there any users left in the list? If not, it means that the LSP will not be used, and it is therefore removed (i.e. LDP is instructed to remove it). Otherwise the LSP is left as it is.

4.3.2 Scenarios triggered by LDP

4.3.2.1 Tearing down of non-BE LSPs

Event:

A link failure causes the removal of all LSPs through the link, including some non-BE LSPs. LDP will notify the Interface Module before the LIB entry corresponding to the torn down non-BE LSP is erased.

Actions:

The interface module will remove the LSPs to be torn down from the LIST_OF_LSPs record in the DiffServ database entry of every user included in the LIST_OF_USERs field in the LIB. This is done because otherwise, when a DiffServ database entry is to be removed and its list of LSPs checked for deletion, non-existent LSPs would be checked, or worse, if a list of pointers is stored, unrelated LSPs could be removed.

4.3.2.2 Establishment of BE LSPs

Event:

LDP informs the DiffServ/LDP interface module of the establishment of new BE LSPs. There are two reasons why the interface module must be informed of this event:

1) There may exist DiffServ entries which specify traffic preferences for destination addresses which do not map to any IP prefix in the routing table except to **0.0.0.0** (default route). Packets with such destination will be labelled with the **default** label and IP routed.

If one of such destinations, say subnet **5.2**, shows up in the routing table of the egress edge routers, a BE path will immediately be set up to the newly appeared subnet (the setup procedure being triggered by the topology change), but no non-BE LSPs will be set up.

Assume a packet destined for subnet **5.2** arrives at the edge router and the DiffServ classifier marks the packet as **PS**. The LSP corresponding to **{5.2, PS}** does not exist. When the IP forwarding tries to retrieve the corresponding label it will find none. At best, the packet will be routed as default, but now the LSP necessary to comply with the SLA subscribed to by the sender could be established.

2) When a link goes down, LDP will remove all LSPs along the link. When the link comes back up, LDP will re-establish (upon modification of the routing tables by the routing protocol) all BE LSPs previously existing through that link, but it will not re-establish PS and AS LSPs because it has no means of knowing that they existed previously. This was not a problem in the topology driven system, since LDP would always establish all three LSPs upon addition of a new routing entry.

Actions:

The DiffServ/LDP Interface Module must determine whether some user requires an AS or PS LSP to be set up to the newly appeared destination. The following actions follow:

- Search for a user requiring PS or AS parallel to the new BE LSPs.
- If a user is found, instruct LDP to start the proceedings to set up the appropriate LSP. Include the user in the LIB **list_of_users** record in

the LSP's LIB entry (the LIB is created by LDP right after the message requesting a label binding is sent), and the LSP in the **list_of_lsps** record in the user's DiffServ database entry.

- Search for more users. If found, include them in the LSP's linked **list of users** and include the LSP in the user's linked **list of LSPs**.

4.3.2.3 Establishment of a non-BE LSP

Event:

A non-BE LSP has been successfully established and the edge router which originated the label binding request gets a label back.

Actions:

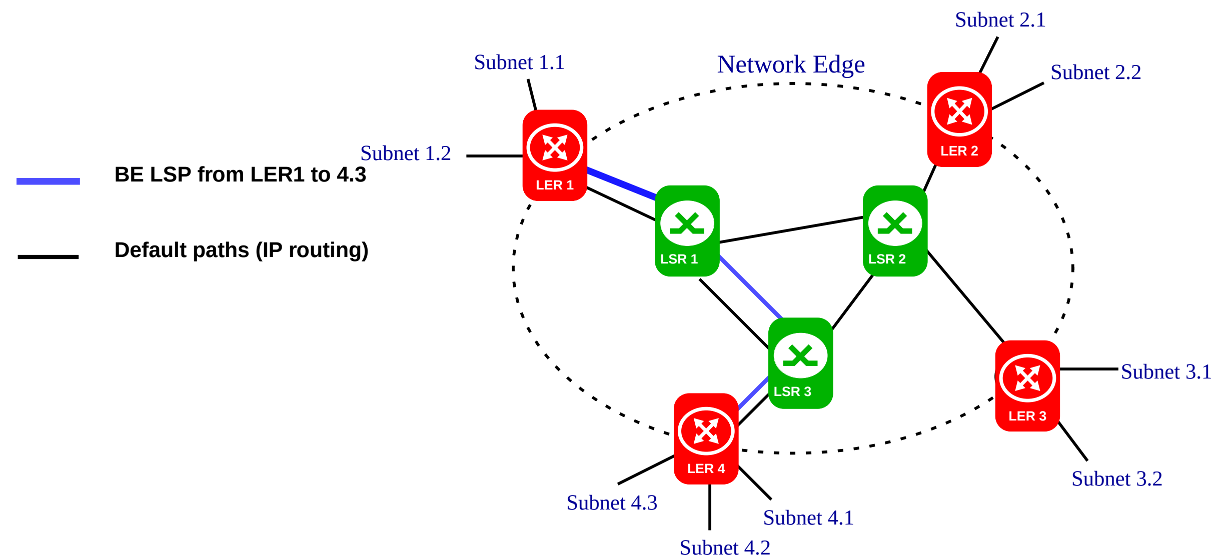
The following data structures are updated:

- The routing table with the corresponding label
- The LIB entry corresponding to the recently established LSP with the label, the new state, etc.
- The DiffServ database entry corresponding to the user/s who are authorized to use the LSP with a pointer to the LIB entry corresponding to the new LSP in the **list_of_lsps** field

5 TRAFFIC DRIVEN MANAGEMENT SOLUTION

5.1 GENERAL DESCRIPTION

When a packet is marked by the DiffServ classifier, it is passed down to the IP routing function, which performs a longest-match lookup in the routing table and returns the label to be attached to the packet. Figure 27 illustrates how the routing table can look like for LER1:



IP Prefix	Next hop	BE	AS	PS
4.3	LSR1	x	default	default

Figure 27 - Only a BE LSP has been established to 4.3

The triggering event for the setup of an LSP will be finding a **default** label when looking for a non-BE LSP label and checking that the LSP is not under establishment. For instance, if a packet headed for destination **4.3** is marked by the DiffServ traffic classifier as **PS**, when IP routing performs a lookup, it will find a default label according to Figure 27 above. After determining that the LSP is not under establishment by checking the corresponding LIB entry, LDP is instructed to establish it.

When LDP is instructed to set up an LSP, it will create a timer, which will be reset every time a packet is send along the LSP. If an LSP sits idle it will time out and LDP will be instructed to remove it.

5.2 MODIFICATIONS TO THE SYSTEM

5.2.1 Addition of a new time field to the LIB

A timer is associated to each LSP by enhancing the LIB with a **time** field. This field is read by a low priority background process and updated with a new time stamp by IP every time a labelled packet is forwarded along the LSP.

When LDP starts the proceedings necessary to set up an LSP, it allocates memory for the corresponding LIB entry, and fills in the corresponding values of the fields. The new **time** field will be initialized with a time stamp. The label field in the routing table is initialized to **default.**

5.2.2 New background process

A low-priority background *monitor* process takes care of checking whether the existing non-BE LSPs have timed out (i.e. have been idle without being used for longer than the established time limit). In this case it will prompt LDP to start the proceedings necessary to remove them.

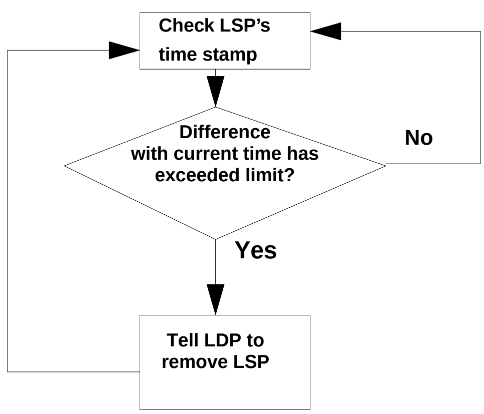


Figure 28 - A background process checks for timed-out LSPs

5.2.3 Addition to the IP forwarding code

When a packet comes in the router, IP retrieves the appropriate label to be attached to the packet from the routing label. If this label is default and the LIB entry associated to the FEC does not exist, IP must prompt LDP to start the set up of the corresponding LSP. If the label is found, IP must then record the value of the system clock in the TIME field of the LIB entry corresponding to the label.

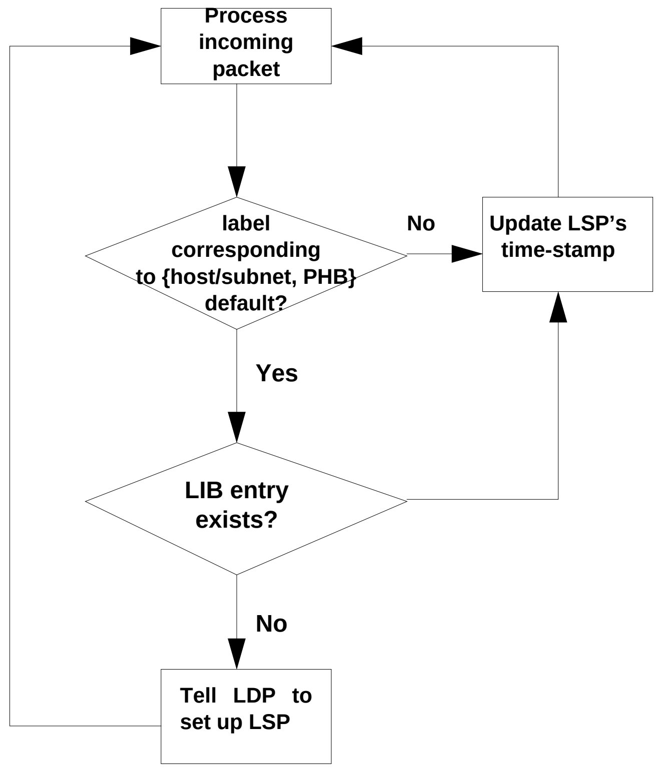


Figure 29 - Additions to the IP forwarding algorithm

5.3 SYSTEM BEHAVIOUR

5.3.1 Processing of incoming packets

Event:

A packet arrives at the edge router and is passed to the IP layer.

Actions:

The following actions take place:

- The DiffServ module will inspect the contents of the layer-3 and layer-4 headers, perform an exact lookup match on the DiffServ database and determine the PHB to be assigned to the packet
- The DiffServ byte in the network layer header of the packet will be marked with the appropriate encoding of the PHB
- A longest match lookup on the routing table returns three possible labels, IP chooses one based on the PHB.
- Does IP find a non-default label for the specified FEC?
- If it does, (i.e. the LSP has already been set up) IP sets the **time** field of the LSP corresponding to the retrieved label to the current system clock value. IP attaches the retrieved label to the packet.
- If it does not (i.e. the needed LSP does not exist yet), IP checks whether the LSP is under establishment by inspecting the **state** field of the corresponding LIB entry. If the LSP is not under establishment, it prompts LDP to create the LSP. IP labels the packet **default**.
- The packet is forwarded along the adequate LSP or using the default path.

5.3.2 Removal of an LSP

The background process described in section 5.2 checks the timestamps of every established LSP periodically. If the difference between the current time and the last time stamp (i.e. the last time an LSP was used) for a certain LSP is above a certain limit, LDP is told to remove the LSP. Otherwise, the process does nothing.

5.3.3 Network topology changes

The traffic driven approach is very stable upon topology changes. When a link goes down LDP will remove <u>all</u> LSPs along the link, regardless of their type of service. When the link comes back up, BE LSPs will be set up driven by the addition of a new routing table entry, and non-BE LSP setup will be triggered in a traffic driven way by the first packets mapping to the appropriate FEC.

While the link is down, the arrival of packets destined for one of the non-BE LSPs which have been removed will still trigger LDP to start the setup procedure. Since the setup will be unsuccessful, the LSP's **state** in the LIB entry will be set to **WAITING**, and packets will be labelled **default** and IP routed.

5.3.4 Time-out value

How long an LSP may remain idle before it is taken down is dictated by the *time-out value*. This value can be changed as needed using the router's command line interface. A short value implies that LSPs will be removed shortly after the last packet of a connection, which will increase the utilization of resources but bring forward two serious shortcomings:

- The processing overhead caused by the background process will be greater, since LSPs will have to be reviewed more often
- Short connections using the same LSP will cause a lot of signaling overhead, requiring the continuous establishment and tearing down of the same LSP.

A high time-out value will add stability to the network configuration, LSPs remaining up longer after the last time they are used. With a high time-out value, there exist several pitfalls as well:

- The overhead caused by the LSP monitoring process can be considered negligible, but resource utilization is lower. A very large time-out value would lead to the establishment of all LSPs necessary to service every possible packet sent by every subscriber, leading initially to a network configuration like the one generated by user profile driven LSP management. No event would however cause the tearing down of an LSP and the resulting final configuration would consume all network resources.

A time-out value must be found which matches the system where the solution will be implemented, taking into consideration factors like network stability, average length of a connection, number of short timed connections using non-BE LSPs, etc.

6 COMPARISON ANALYSIS

The two previous proposals address the problem using different approaches and offer different solutions. In this comparison, both solutions are analysed and presented with several scenarios, encountering some of the usual trade-off ever present in networking: signalling overhead vs. per packet processing overhead, utilization vs. SLA-compliance, etc.

From the edge router point of view, attention has been paid to the effect of the solutions on packet forwarding and to the overhead generated by LSP setup and tearing down procedures.

As far as the whole network is concerned, issues like resource utilization, signalling overhead, stability, and scenarios created by network congestion have been examined.

Since the network scenario considered is that of a service provider network interconnecting LANs and dial-in users, the compliance with the Service Level Agreement and the effect of both solutions on the operation of networked applications has also been looked into.

6.1 RESOURCE UTILIZATION

6.1.1 Assumptions

- The ingress router's DiffServ database contains user profiles corresponding to users which are currently connected to the network. When a user is no longer connected, his DiffServ entry is removed from the database residing in the edge router. As new users connect to the network, their DiffServ entries are added to the database. This means that a local repository of Service Agreements exists and communicates with the Edge Router. DiffServ entries are downloaded from a server on to the Edge routers when new users connect. For the purposes of this thesis, it could be considered that entries are added and removed manually[vi].

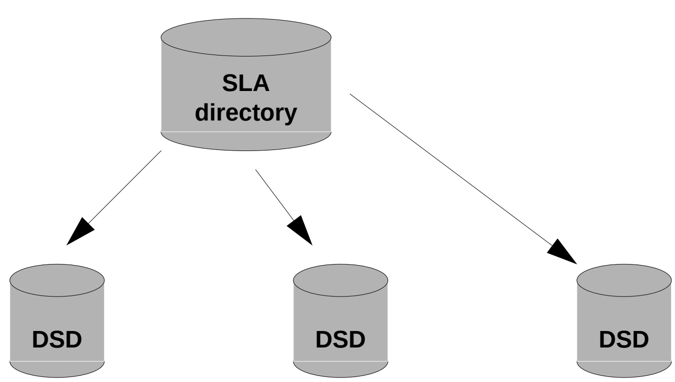


Figure 30 - The DiffServ databases contain information about users currently connected

vi. In the AXI 512 Edge Router, user entries can be inserted through the command line interface or through RADIUS for dial-in users.

- Three traffic classes supported: BE, AS, and PS. BE LSPs are established and torn down in a topology driven way. See Section 2.1.5 for details.
- No aggregation of destination targets is assumed.
- Since the number of BE-LSPs is, for a given topology, constant and the same for both proposals; it is the utilization of PS and AS LSPs which must be optimized.
- PS is assumed to be a limited and expensive service, to be used for bandwidth exacting or business critical applications.The number of PS LSPs will be lower than the number of AS LSPs and far below the number of BE LSPs.
- The MPLS/DiffServ system considered does not perform any reservation of resources along the path of an LSP. Signaling procedures bind a label to a destination and a forwarding treatment. Thus, a PS-LSP is treated exactly as a BE-LSP, the only difference being that PS packets are dispatched much faster by the intermediate nodes, because the PS queue is always emptied before non-preferential queues.

6.1.2 Traffic driven system

LSPs are established when they are first used and torn down after a certain period of time without being used. Thus, only those LSPs which are actually being used are established. The utilization of LSPs increases as the time out value decreases and so does the overhead inherent to monitoring the state of every LSP.

6.1.3 User profile driven system

The user profile driven system establishes as many non-BE LSPs as necessary to service all FECs specified in the DiffServ database. Resources are allocated to meet the traffic preferences of all users for all possible connections.

6.1.4 Conclusions

- At any given point in time, the traffic driven system will have set up at most as many non-BE LSPs as the user profile driven system, in the worst case scenario of all users using all the services they are subscribed to at the same time. This is very unlikely to happen, and therefore the number of LSPs setup at a given time by the traffic driven approach is clearly lower.

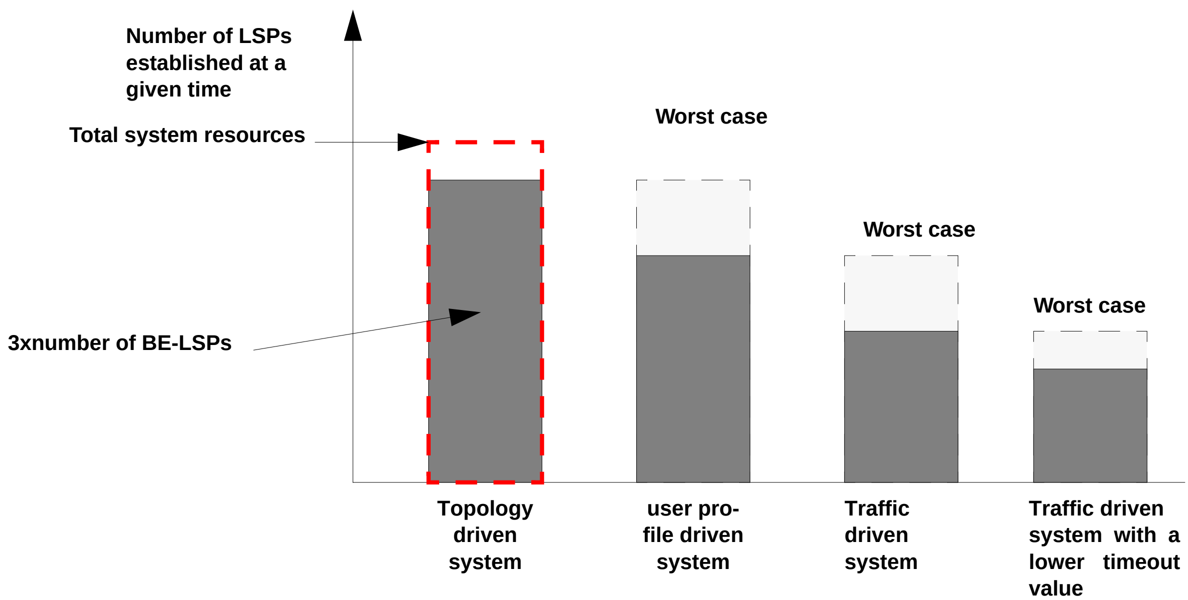


Figure 31 - Resource utilization

Figure 31 above shows a comparison of how network resources are utilized by four different systems: the topology driven system explained in Section , the user profile driven system, the traffic driven system with a medium time-out value, and the traffic driven system with a very low time-out value (called pure traffic driven system). The system achieving the best utilization is the traffic driven with a low time-out value, but the overhead caused by the continuous monitoring of LSPs makes a low time-out value not worth the gain. A traffic driven system with a higher time-out yields a better utilization of the network than the user profile driven approach, while keeping a low monitoring overhead.

6.2 SIGNALLING OVERHEAD

Both solutions generate different signalling frequencies in different network scenarios.

6.2.1 Assumptions

- Signalling is used in an MPLS network to bind labels with FECs along forwarding paths, thus building an LSP, or to release such bindings. It is assumed in this section that when an LSP is torn down, the labels assigned to it are disposed and can be reused. In MPLS terms, this means that *liberal label retention mode* (See [4] for further details) is used, labels being discarded when the corresponding LSP is torn down. This is how Ericsson's implementation works.

- In this section a completely static topology is assumed, the effects of topology changes upon each system are analyzed in Section 6.3.
- The period of time it takes to signal the set up of an LSP using downstream on-demand ordered control (see Section 2.1.9) can range from very low to abnormally high values in the case of MPLS networks with large congested switching cores [21]. An abnormal signalling delay is a sign of bad network design or insufficient resources[vii].
- The only events generating LDP signalling overhead are the establishment and removal of LSPs.

6.2.2 User profile driven System

In the user profile driven system, there are two events which could lead to a change in the non-BE LSPs configuration and therefore produce signalling: the addition of an entry to the DiffServ database and the deletion of an entry from the DiffServ database.

If we assume a stable topology, LSPs will be established as users connect to the network (i.e. as their service profiles are added to the DiffServ database) and torn down as users log off.

If a user subscribes to a service not covered by any LSP, the necessary LSPs will be set up. When users connect later and request the same service, no LSP

---

vii. In the case where the signalling delay is high, both solutions will be affected, but the performance of the Traffic driven system drops dramatically if the average LSP setup time is very high. The setup time can be considered as the sum of the round trip time (usually negligible, except in very large networks, e.g. 10 ms in a 600 km wide core) and the per-nod processing time including queuing delay. The fact that the considered system uses an ATM core adds relevance to this problem, since a small cell loss average will result in a much larger packet loss value because of the fact that an IP frame 1500 bytes long is fragmented into 32 ATM cells and the loss of one of them suffices to cause the packet to be dropped at the egress router. This is a platform dependent problem and has not been considered in this thesis, since a proper assessment of this problem was beyond the means of the project.

will be established because LSPs with the same characteristics are shared since there is no bandwidth reservation in Ericsson's implementation.

When a user disconnects from the network, his profile will be removed from the DiffServ database and the LSPs he could be using will be checked, removing those which have no other owner.

With this approach, the LSPs necessary to service the FECs specified by a user in the SLA, are established every time the user logs in, regardless of whether he will use them or not. This means that if users connect to the network for short periods of time several times a day, and they have specified non-BE FECs in their SLA, this will generate a lot of signalling overhead.

6.2.3 Traffic driven system

The events that will trigger signalling in the traffic driven system is the arrival of a packet mapping to an FEC which is not serviced by any already established LSP and the timing out of an LSP.

The traffic driven system reacts differently depending whether most connections which use non-BE LSPs are long or short-lived. A user might use a PS LSP to download a file using FTP and then not use the LSP during a period of time above the time-out value. Many different short lived connections making use of non-BE LSPs will generate a lot of signalling and computing overhead.

6.2.4 Conclusions

Two factors affect system behaviour when frequency of setup/tearing down of LSPs events is considered:

- Duration of data flows (a flow is a stream of packets with the same PHB and destination address and port)
- Frequency of log-in/log-off events

If most data transfers are short-lived flows, the performance of the Traffic Driven System will be seriously undermined, because it implies a high signalling frequency. This situation does not affect however the user profile driven approach.

A frequent occurrence of login/logoff events will cause the performance of the user profile driven system to sink, since it implies a frequent recalculation of the network configuration and the establishment and tearing down of the necessary LSPs. It will not influence the operation of the Traffic Driven System.

6.3 BEHAVIOUR WHEN TOPOLOGICAL CHANGES OCCUR

6.3.1 User profile driven system

The user profile driven system behaviour when topology changes occur is described in Section 4.3.2. Basically, the Interface module is notified each time a non-BE LSP is torn down due to a link failure or a route change and it updates state information kept in the LIB and DiffServ database. When new BE-LSPs are set up, topology-driven, the interface module is notified as well, and it must check the whole DiffServ database to see whether there are any users subscribing a preferential service parallel to the new BE-LSPs.

Checking the whole DiffServ database and re-establishing LSPs which could be using the link before it went down means many CPU-intensive database searches at the edge router, and implies as well re-establishing LSPs which might not be used at all.

6.3.2 Traffic driven system

The traffic driven system does not react upon topology changes. When non-BE LSPs are torn down due to a topology change, the system will try to re-establish them as packets mapping to the appropriate FEC arrive. While the LSP is down, packets will be labelled **default** and IP-routed. BE-LSPs will be re-established in a topology driven way, triggered by OSPF routing table updates.

6.3.3 Conclusions

- The behavior of the traffic driven system is much better with an unstable network topology. It brings about only the necessary signalling overhead and no processing overhead at all.

6.4 EFFECTS ON PACKET FORWARDING

6.4.1 Assumptions

- Processing and signalling overhead caused by the establishment and removal of LSPs and by the Interface module in the case of the user profile driven solution are not considered here, but do have influence in how fast the router is able to forward packets, since they require CPU time.

- The overhead caused by the monitoring process in the traffic driven solution is not considered either, but affects forwarding performance just the same. The low priority process is less exacting than all the overhead generated by DiffServ lookups, but its frequency is higher.

- The effect on per-packet performance is interpreted in terms of CPU time destined to other purposes but packet routing. This CPU time is spent mainly doing database lookups.

6.4.2 User profile driven system

Once an LSP has been setup, the user profile driven system does not affect at all the packet forwarding process.

6.4.3 Traffic driven system

The packet forwarding process must be slightly modified to accommodate the traffic driven solution. This modification is detailed in Section 5.3.1. When a packet comes in the edge router, it is classified and marked by the DiffServ Classifier, and the FEC the packet maps to is determined based on the results of such classification and a look-up in the routing table. Up to this point the forwarding process is identical to the one in the user profile driven system.Two cases can now be considered:

1) The IP forwarding function is unable to find the label corresponding to the FEC in the routing table. This means that the LSP matching the FEC has not been established. The following takes place:

- The IP module instructs LDP to set up the LSP for the given FEC.
- The system clock value is recorded in the **time** field of the LIB entry corresponding to the LSP to be established.
- The packet is labeled **default** and forwarded.

2) The LSP the packet must be forwarded along does exist, and therefore the forwarding function does retrieve a valid label. The following takes place:

- The packet is labeled and forwarded
- The system clock value is checked and recorded in the **time** field of the appropriate LIB entry.

So, all the per packet processing overhead added by the Traffic driven system is a logical check (to see whether a null label is returned) and recording the value of the system clock in a memory register. All checks are done via pointer references, therefore no database searches are necessary. Compared with the code and processing associated to doing per packet classification and routing table lookup, it is a very inexpensive operation.

6.4.4 Conclusions

Even though the traffic driven system introduces some processing overhead in the packet forwarding algorithm while the user profile driven system does not, the time delay generated can be considered as negligible.

6.5 OVERHEAD GENERATED BY LSP SET-UP AND TEARING DOWN

6.5.1 User profile driven system

Section 4.3.1 details how and when the user profile driven system instructs LDP to set up or tear down an LSP.

A new LSP is set up when a new entry is added to the DiffServ database which specifies a service which is not already covered by an existing LSP. The new entry must be parsed, and the one routing table lookup at least is necessary per destination address contained in the entry. Thus, the set up of a new LSP implies at the very least a routing table lookup and some additional processing.

The decision to tear down an LSP is taken when the only user requesting it disconnects from the network. The overhead implied is not much, only the list of LSPs contained in the user's DiffServ database entry must be checked.

This procedures do not impose a heavy burden in the router, unless they are performed very often.

6.5.2 Traffic driven system

The triggering of LSP setup is built into the IP forwarding mechanism and does not add any additional overhead apart from the per packet delay introduced. The decision to tear down an LSP is taken by the monitoring process, which is implemented as a low priority daemon process.

Exactly how requiring the monitor process is depends on the time-out value. With a reasonable value, the overhead generated by the process checking the time field of all the LIB entries and comparing it with the current time should not be high.

6.5.3 Conclusions

Both implementations do add some load to the router's CPU and place memory requirements. none, however, is exacting enough as to make this a determining factor.

6.6 COMPLIANCE WITH SERVICE LEVEL AGREEMENT

6.6.1 Assumptions

- A SLA is signed between the Service Provider (who owns the MPLS network) and the user. In this SLA the user specifies his traffic preferences and the SP agrees to providing them up to a certain guarantee.
- The exact nature of the SLA depends on the SP policy and on how the user is going to be charged.

6.6.2 User profile driven system

The user profile driven System sets up the LSPs necessary to provide the services specified by the user in the SLA when the user connects to the network. Once the LSPs requested by the user are established, the services subscribed to are readily available during the whole duration of the connection. A data transfer using one of the established PS-LSPs will be treated as Premium traffic entirely, from the very first packet.

6.6.3 Traffic driven system

In the traffic driven system, a user can send packets which, after being properly classified into a certain FEC, are sent along an LSP which already existed, thus being immediately delivered a preferential treatment. This cannot be assumed however. A connection whose packets map to an FEC which is not serviced by any LSP will be treated unevenly. The first packet belonging to this connection will trigger LDP to start the setup of the needed LSP as shown in Figure 32 below.

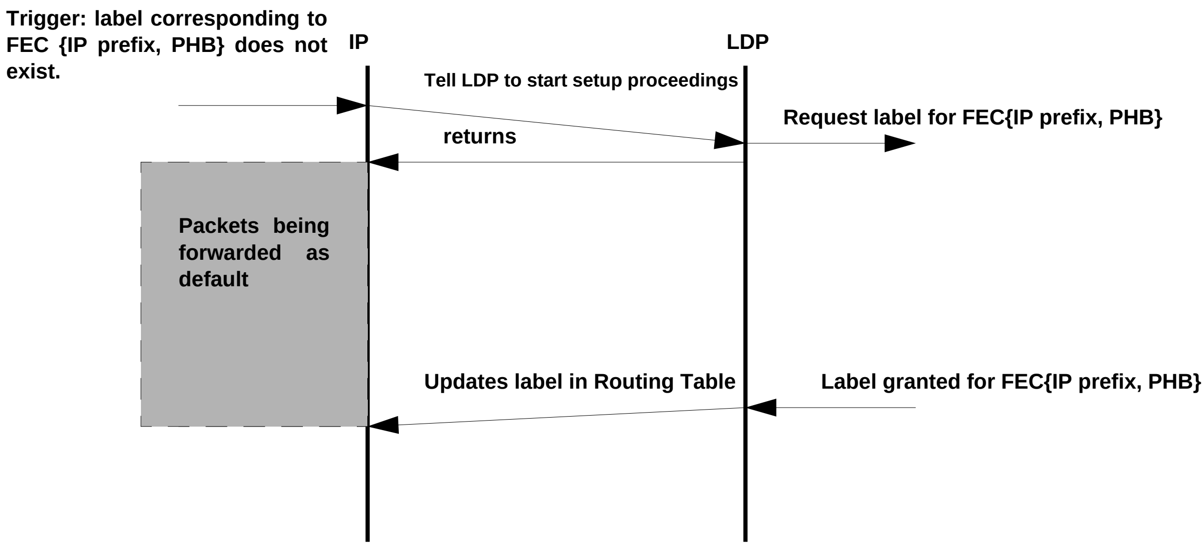


Figure 32 - SLA compliance

Until the LSP has been established and the corresponding label recorded in the LER's routing table, all packets belonging to that connection will be routed as **default**.

6.6.4 Conclusions

Figure 33 below shows how the traffic driven system behaves. Every time the first packet of a new flow arrives and the LSP necessary to properly forward the packets of the flow is not established, this will trigger the establishment of an LSP. Until the edge router gets a response back with the appropriate label binding and it updates the routing table, all packets from the flow will be forwarded as default (using IP over ATM). This means that the user must be billed accordingly since a significant amount of his packets will not received the requested treatment.

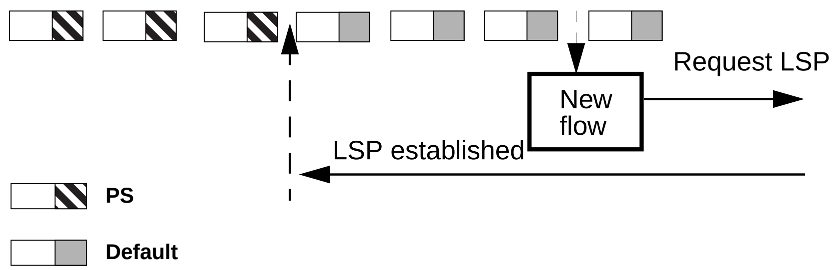


Figure 33 - Traffic driven system

Figure 34 shows what happens in the user profile driven System. When a user logs in and until all the LSPs necessary to comply with his user profile are established, packets will be forwarded as default. However, this only happens once every time a user logs in. The user must launch an application and start sending packets along LSPs which have not been established yet to cause this situation, and even then it is only temporary. When all the appropriate LSPs have been established, all subsequent packets will be properly treated according to the information contained in the DiffServ database.

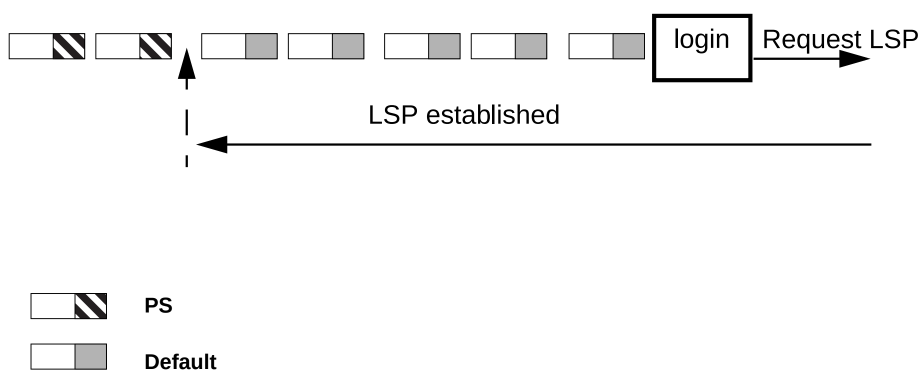


Figure 34 - user profile driven system

6.7 EFFECT ON APPLICATION PERFORMANCE

Bandwidth-consuming applications like the transfer of video or voice usually assess the state of the network before transferring any data, providing better or worse quality according to network parameters like available bandwidth, delay, and jitter.

Since packets belonging to the same connection might be treated in different ways by the traffic driven approach, being the head of the connection treated in a best effort manner and the tail with a QoS treatment, the application could assess the state of the network too early and underutilize the services offered.

This depends on how the applications are implemented. If continuous network monitoring is present, the application will be able to fully use the service the user is subscribed to.

6.8 CONCLUSION: CHOICE OF A SOLUTION

Both systems are feasible to implement and provide an important optimization when compared to topology driven allocation of LSPs. The decision whether to choose one or the other is closely related to the actual system where the solution will be installed and to the characteristics of the subscribers to the network services. none of the factors taken into consideration lead to discarding one of the two solutions.

6.9 SUMMARY OF COMPARISON

The biggest pitfall of the traffic driven solution is that it does not provide a whole traffic flow with the same treatment, this can lead to the following problems:

- non compliance with the SLA
- Confusing state of the network. Applications are unable to determine the exact state of the connection in terms of parameters like delay or bandwidth.

In a network scenario with a large number of short-lived connections making use of preferential LSPs, the traffic driven system will not perform well and will not scale as the number of users increases.

The user profile driven solution does treat connections equally and services subscribed to by the user are made available almost from the beginning. It does however not scale well in network scenarios with frequent topology changes or with many users connecting to the network for short periods of time.

6.10 SYSTEM SUITABLE FOR THE USER PROFILE DRIVEN SOLUTION

The ideal system for implementing the user profile driven solution presents the following features:

- Users log in once and remain logged in during a few hours at least. Small LANs and home-working users adapt to this profile. They connect to the MPLS network once a day and stay connected at least eight hours.

- The core network is reasonably stable and does not cause frequent changes in the routing tables.

6.11 SYSTEM SUITABLE FOR THE TRAFFIC DRIVEN SOLUTION

The ideal system for implementing the traffic driven solution presents the following features:

- Connections using preferential forwarding are long-lived. Users do not send data along non-BE LSPs for short file transfers, web browsing or e-mail.
- The SLA is flexible enough to allow BE treatment of the head part of some connections.

6.12 CONSIDERED SYSTEM

The system considered by this thesis to implement the MPLS system with DiffServ support presents the following features:

- The edge router used is a small access router which acts as an access point for ADSL and dial-in users.
- Two user groups can be distinguished: the home user, using only BE LSPs to do web browsing and e-mail, and the SOHO user (Small Office/ Home Office) who connects to the network and stays connected most of the day.

6.13 CHOICE AND JUSTIFICATION

The solution chosen for implementation on the AXI 51x Edge Router was the user profile driven solution. The reasons for this choice were:

- The platform is an access router designed for ADSL Broadband access, the user group will be mainly composed of LANs and SoHo users or home-workers.
- Users actually needing the provision of QoS (i.e. prone to use non-BE LSPs) are within the SoHo user group and therefore are likely to log in once a day and stay connected[viii].
- No assumptions about the core behavior could be made.

- It provides on-demand allocation of resources without the latency problem present in the traffic driven solution. A short time after logging in, a user has access to all services he subscribes to.

7 DISCUSSION AND FUTURE WORK

Even though the solution chosen for implementation was the user profile driven one, the Traffic Driven System provides a simple and dynamic solution, less CPU intensive and which adapts faster to topological changes. It is definitely worth a more thorough evaluation.

Prototyping and testing both solutions in different environments would be an excellent way to make a decision on which solution to implement on a certain platform. An in-depth study is needed to determine what performance improvements are achieved by each solution, and how much conditions like signalling overhead or delay to set up an LSP affect each system. It seems likely that each solution will adapt well to a system with the proper characteristics and the conclusion of this thesis is that none of them can be discarded without further studies.

8 ACKNOWLEDGEMENTS

The author of this Master's Thesis would like to thank Gunnar Karlsson and Viktoria Elek from the Royal Institute of Technology in Stockholm, Eric Lin and Kerry Lowe from Ericsson Datacom & IP Services, Manuel Sanchez from Ericsson Core Unit of Technology and Research, and the whole unit of PDU Access Stockholm for all the help and support (specially Daliang Wen for all his time and Andreas Johansson for his “excellent” taste for Swedish music from the 50s).

9 APPENDIX A : PROOF OF CONCEPT

9.1 SUMMARY

This appendix details issues related to the implementation of the user profile driven Solution on the AXI 512 Access Router.

In order to implement this solution to the non-BE LSPs management problem, existing data structures and modules must be modified, and a new module must be built to act as an interface between MPLS and DiffServ.

viii. US ISPs are offering fixed rate subscriptions. An ADSL user logs in once and stays connected, paying a fixed monthly fee

9.2 DATA STRUCTURES AFFECTED

9.2.1 Routing table

The routing table in the edge router is implemented as an AVL tree [31]. An AVL tree is a binary tree nearly balanced at all times thus limiting both the worst and the average search time. The nodes of the tree are data structures called RT_NODE, Figure 35 illustrates the relevant fields.

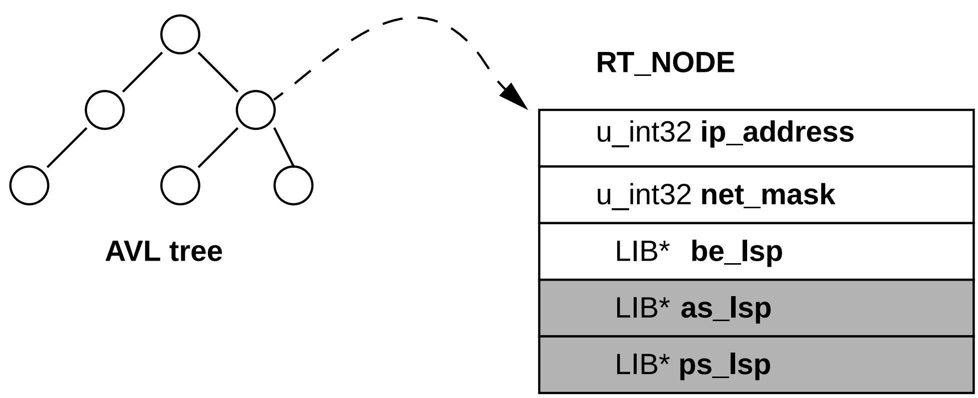


Figure 35 - Routing table relevant fields

The shaded fields in Figure 35 have been added to each RT_NODE structure to enable the forwarding of packet along non-BE LSPs (i.e. the AS and PS LSPs). They contain pointers to the LIB (see 9.2.3) entries of the corresponding LSPs or to the default LIB entry (if those LSPs have not been set up yet). Via the LIB entry, the correct label can be referenced when performing packet forwarding.

9.2.2 Differentiated Services database

As users log in, the edge router contacts a RADIUS [11] server, which performs authentication of users and accounting. If the authentication procedure proves positive, the RADIUS server will send a message accepting the connection and will piggyback the DiffServ profile of the user into this message[ix].

---

ix. User profiles can be inserted manually by using the command line interface (CLI). However, at the time of this implementation, the CLI was not completed and therefore could not be tested.

The DiffServ database is structured as an AVL tree as well, allowing for efficient searching.

This user DiffServ profile specifies which services the user is subscribed to and therefore how his packets should be marked by the DiffServ classifier. User profiles are stored in the Differentiated Services database in the edge router, in data structures as the one shown in Figure 36.

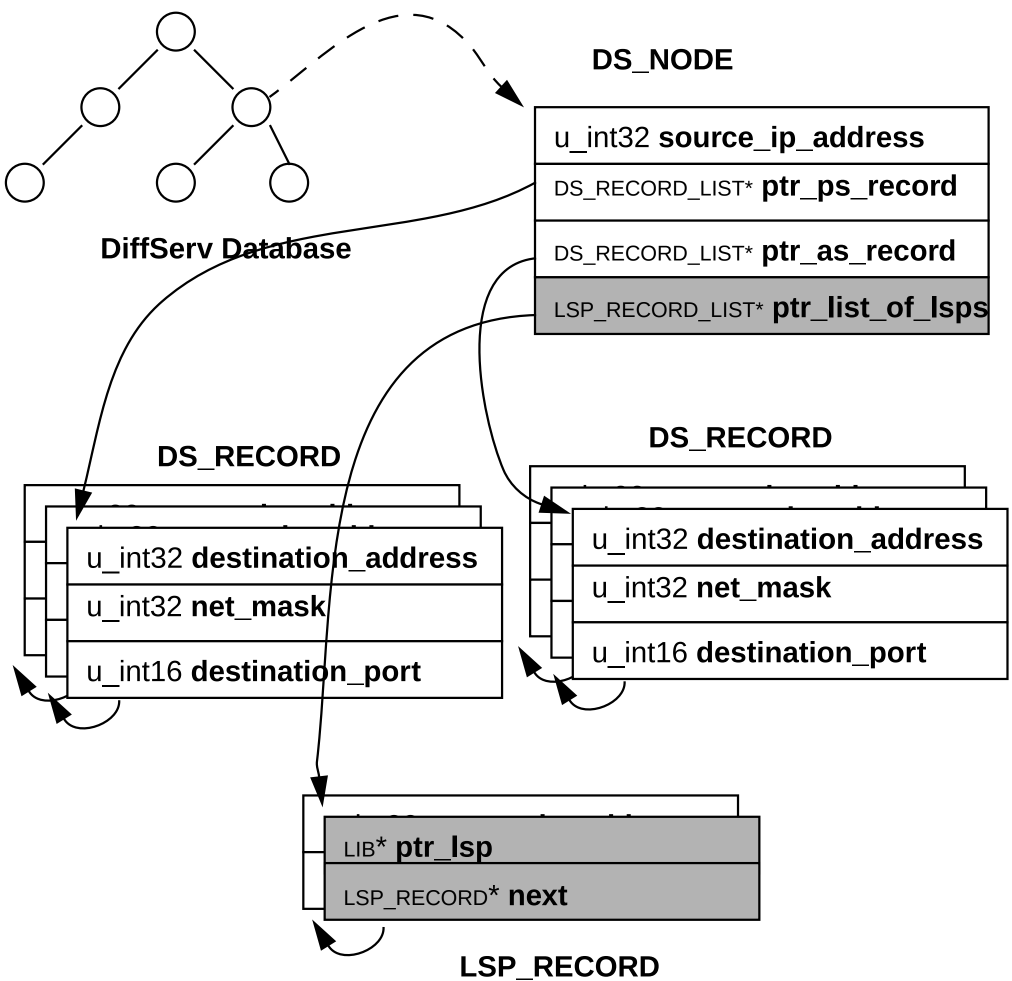


Figure 36 - DS_NODE

A linked list of LSP_RECORDS containing pointers to LIB structures has been added to the existing DS_NODE. New fields are shown shaded in the figure above.

9.2.3 Label information base (LIB)

The LIB holds information on each LSP that is established on under establishment procedures. Figure 37 below shows relevant fields in an LIB entry.

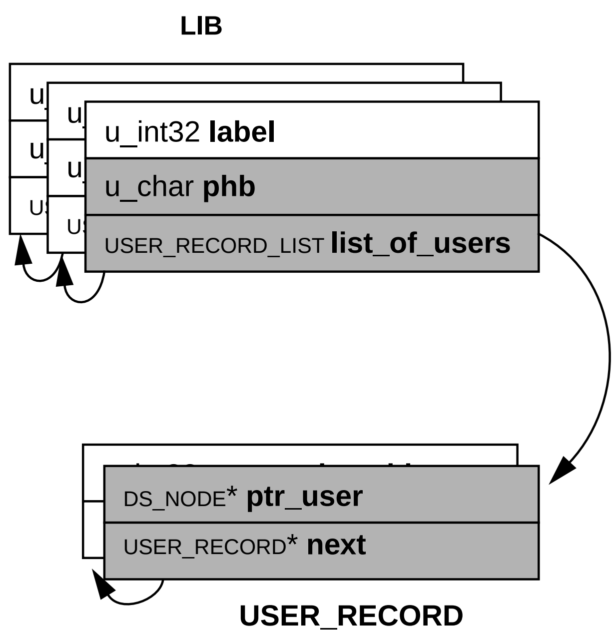


Figure 37 - LIB entry

A linked list of records has been added to each LIB entry for LSP management purposes. A new field (**phb**) has been added to specify which forwarding treatment is being provided by the corresponding LSP.

Figure 38 below provides a general view of all three databases and how they relate to one another.

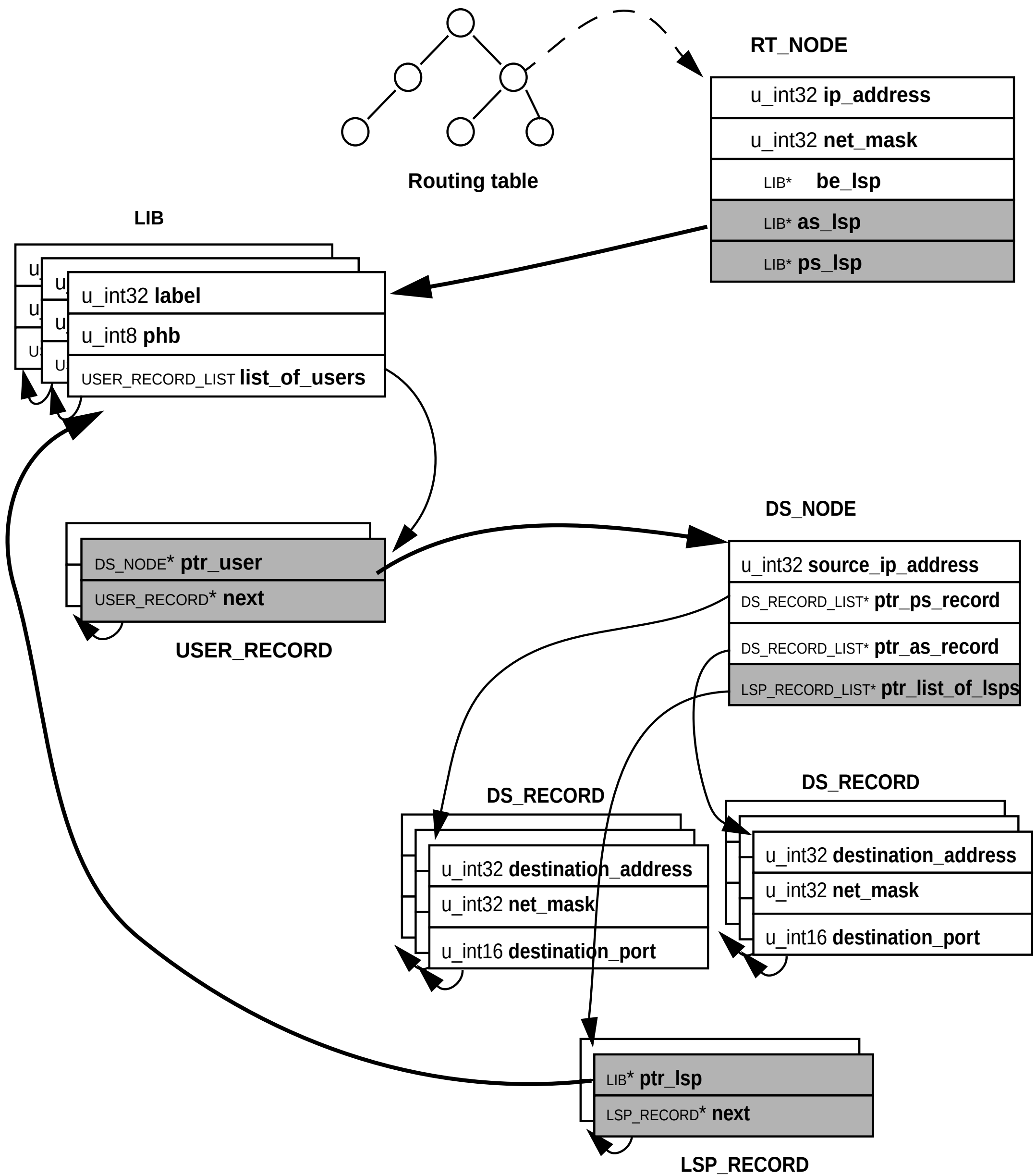


Figure 38 - Relationship among all three databases

9.3 FUNCTIONS

Functions necessary for the operation of the user profile driven System are listed and explained in this section. Functions belonging to the IP, DS, or LDP modules required only slight modifications to adapt to the new requirements.

The four functions making up the Interface Module are completely new and have been designed specifically for this system. A more detailed description of these functions is provided in form of pseudo-code. Auxiliary database management functions are enumerated as well.

Functions described in this section can be broken into existing functions, functions that required small modifications, and completely new functions.

9.3.1 IP module

Only one function from the IP module is used and it was not necessary to modify it.

9.3.1.1 ip_fwd_best_match (): Existing function

Arguments:

IP address

Returns:

Pointer to the RT_HEAD data structure which is the best match for the packet's destination address

Called from:

- **ip_forward ()** to determine the next hop for a packet to be forwarded.
- **ldp_ds_check_setup()** to determine routing table entry corresponding to a longest match look-up on an IP address

Description:

This function performs a lookup in the routing table trying to determine the longest match for the destination IP address of the packet. The function starts searching at the highest resolution nodes (those with a longer netmask) and it proceeds as follows:
- The search key (destination address) is ANDed bitwise with the subnet mask of the RT_NODE entry and it is compared with the subnet/host address of the entry.
- If they don't match the search continues. If they do, it stops and returns the given RT_NODE, since the routing table is organized so that subsequent searches will always yield shortest matches.

Modifications:

No modifications were necessary

9.3.2 DiffServ module

The DiffServ module was modified to include system calls to functions in the Interface Module and new functions were added.

9.3.2.1 LSP RECORD LIST management: New functions

Several functions are necessary to manage the list of lsps linked off the DiffServ database's DS_NODES, among others:

9.3.2.1.1 ds_insert_lsp()

Arguments:

Pointer to LIB entry to be inserted

Pointer to the LSP_RECORD_LIST where the entry is to be added

Description:

Inserts a pointer to an LIB entry in the LSP_RECORD_LIST corresponding to a user needing the LSP

9.3.2.1.2 ds_delete_lsp()

Arguments:

Pointer to LIB entry (list element) to be deleted

Pointer to the LSP_RECORD_LIST (list) where the entry is to be added

Description:

Removes list element (LIB entry) from list.

9.3.2.1.3 ds_lsp_list_empty()

Arguments:

Pointer to the LSP_RECORD_LIST

Returns:

TRUE if list is empty, FALSE otherwise

Description:

Checks if list is empty

9.3.2.2 ds_insertDSnode (): Modified function

Arguments:

Pointer to the User Profile downloaded when a user logs in

Returns:

Pointer to the DS_NODE data structure which contains the information on the user's traffic preferences

Description:

- It allocates memory to hold the User Profile
- It builds the DS_NODE structure and links it to the DiffServ database

Modification:

- Before returning a pointer to the new DS_NODE structure it makes a function call to **ldp_ds_check_setup()** with a pointer to the new DS_NODE as argument.

9.3.2.3 ds_exact_match (): New function

Arguments:

IP address (32-bit long integer)
Subnet mask (32-bit long integer)

Returns:

Pointer to a RT_NODE structure or NULL if no match was found

Description:

- A search key is obtained ANDing bitwise both 32-bit integers supplied as arguments
- This function parses the whole routing table comparing the search key and the subnet/host IP prefix of each routing table entry **ANDed with the subnet mask supplied as argument**
- If they match, a static variable holding the current position of the search is updated with that pointer and the function returns the value of the RT_NODE memory address
- Every time the function is called, it searches from the position it returned the time before

- If no match is found, a NULL pointer is returned

9.3.2.4 ds_deleteDSnode (): Modified Function

Arguments:

The IP address of the user who logged off and whose entry is to be removed from the DiffServ database

Description:

- Finds DS_NODE corresponding to source IP address
- Calls **ldp_ds_check_teardown ()** with pointer to DS_NODE
- Removes the entry from the DiffServ database

Modification:

Call to ldp_ds_check_teardown()

9.3.3 LDP

Several functions in the LDP module are modified to include function calls to the Interface Module and some new functions have been added.

9.3.3.1 USER RECORD LIST management: New functions

Several functions are necessary to manage the list of users linked off the LIB entries, among others:

9.3.3.1.1 lib_insert_user()

Arguments:

Pointer to DS_NODE entry to be inserted

Pointer to the USER _RECORD_LIST where the entry is to be added

Description:

Inserts a pointer to an DS_NODE entry in the USER _RECORD_LIST of an LSP's LIB record

9.3.3.1.2 lib_delete_user()

Arguments:

Pointer to LIB entry (list element) to be deleted

Pointer to the LSP_RECORD_LIST (list) where the entry is to be added

Description:

Removes list element (LIB entry) from list.

9.3.3.1.3 ds_lsp_list_empty()

Arguments:

Pointer to the USER_RECORD_LIST

Returns:

TRUE if list is empty, FALSE otherwise

Description:

Checks if list is empty

9.3.3.2 ldp_bind_receive(): Modified function

Arguments:

Pointer to LIB entry for which a binding message has been received

Called from:

LDP module upon the receiving of a label binding message

Description:

- Updates routing table with new binding
- Performs modifications

Modifications:

- If new LSP is non-BE, it calls **ldp_ds_update_dsd ()** with a pointer to the LIB and ADDED_LSP as second argument
- If new LSP is BE, it calls **ldp_ds_check_changes()** with a pointer to the LIB entry

9.3.3.3 ldp_lsp_add(): Existing function

Arguments:

- pointer to the RT_NODE for which the LSP is being established
- Traffic class (PS or AS)

Returns:

Pointer to the LIB entry corresponding to the LSP under establishment

Called from:

The DiffServ module when the LSP to be established is non-BE. If the LSP to be established is BE, this function is called from the Routing Table Manager module.

Description:

- It allocates memory for and builds the LIB entry corresponding to the LSP to be established
- It updates the RT_NODE structure with the pointer to the new LIB entry
- It generates a message requesting a label mapping for the given LSP, sets the state of the LSP to WAITING and returns.

9.3.3.4 ldp_lsp_remove (): Existing function

Arguments:

Pointer to the LIB entry corresponding to the LSP to be removed

Called from:

The DS module or the Routing Table Manager

Description:

- Initiates the removal of an established LSP
- Updates RT_NODE structures affected by the removal of the LSP

9.3.3.5 ldp_lib_delete (): Modified function

Arguments:

Pointer to the LIB entry to be removed

Called from:

LDP upon receival of a label release message to erase the LIB entry corresponding to the LSP which has been torn down.

Description:

- Calls **ldp_ds_update_dsd()** with a pointer to the LIB entry to be removed and REMOVED_LSP as second argument (mode)
- Deletes entry

Modification:

Call to ldp_ds_update_dsd()

9.3.4 DS/LDP Interface Module

The Interface module is completely new and it comprises four functions.

9.3.4.1 ldp_ds_check_setup (): New function

Arguments:

Pointer to DS_NODE

Returns:

TRUE if successful completion

Called from:

DiffServ module: ds_insertDSnode ()

Description: Pseudo-code looks like:

```
for
        every entry in the Premium Service record linked
        list in the DS_NODE
do
        Call ds_exact_match () with destination IP ad-
        dress and mask taken from DS_RECORD
        while
            ds_exact_match () returns a non-null pointer
            to an RT_HEAD structure
        do
            if
                pointer in PS field in RT_HEAD entry
                points to default
            then
                call ldp_lsp_add() with RT_HEAD and PS
                codepoint as arguments
```

```
                Insert a pointer to the DS_NODE in the
                USER_LIST in the LIB (pointer to LIB entry re-
                turned by ldp_lsp_add() )

            if ds_exact_match() did not return any non_null
            pointer to an RT_HEAD
            then
                Call ip_fwd_best_match() with IP destination
                address ANDed with the mask. A pointer to a
                RT_HEAD structure is returned
                if
                    pointer in PS field in RT_HEAD entry
                    points to default
                then
                    call ldp_lsp_add() with RT_HEAD and PS
                    codepoint as arguments

                Insert a pointer to the DS_NODE in the
                USER_LIST in the LIB (pointer to LIB entry re-
                turned by ldp_lsp_add() )

        for
            every entry in the Assured Service record linked
            list in the DS_NODE
        do
            Call ds_exact_match () with destination IP address
            and mask taken from DS_RECORD
            while
                ds_exact_match () returns a non-null pointer
                to an RT_HEAD structure
            do
                if
                    pointer in AS field in RT_HEAD entry
                    points to default
                then
                    call ldp_lsp_add() with RT_HEAD and AS
                    codepoint as arguments

                Insert a pointer to the DS_NODE in the
                USER_LIST in the LIB (pointer to LIB entry re-
                turned by ldp_lsp_add() )
            if ds_exact_match() did not return any non_null
            pointer to an RT_HEAD
            then
                Call ip_fwd_best_match() with IP destination
                address ANDed with the mask. A pointer to a
                RT_HEAD structure is returned
                if
```

```
        pointer in AS field in RT_HEAD entry
        points to default
    then
        call ldp_lsp_add() with RT_HEAD and AS
        codepoint as arguments
    Insert a pointer to the DS_NODE in the
    USER_LIST in the LIB (pointer to LIB entry re-
    turned by ldp_lsp_add() )
```

9.3.5 ldp_ds_check_teardown(): New function

Arguments:

Pointer to a DS_NODE about to be deleted

Returns:

TRUE if successful completion

Called from:
The DiffServ module: ds_insertDSnode ()

Description: Pseudo-code looks like:

```
for
        every LIB entry pointer in the LSP_LIST linked off
        the DS_NODE:
do
        delete_user () using the user's IP address and a
        pointer to the list as arguments
        if
            lsp_user_list_empty ()
        then
            ldp_lsp_delete () with pointer to LIB as argu-
            ment
```

9.3.6 ldp_ds_update_dsd () : New function

Arguments:

- Pointer to LIB entry

- Mode: REMOVED_LSP (0) or ADDED_LSP (1)

Returns:

TRUE upon successful completion

Called from:

The LDP module: ldp_bind_receive() or ldp_lib_delete ()

Description: Pseudo-code looks like:

```
if
        mode is ADDED_LSP
then
        for
            every DS_NODE pointer present in the USER_LIST
            of the LIB entry
        do
            insert_lsp() with a pointer to the list and a
            pointer to the LIB entry as arguments.
if
        mode is REMOVED_LSP
then
        for
            every DS_NODE pointer present in the USER_LIST
            of the LIB entry
        do
            delete_lsp() with a pointer to the list and a
            pointer to the LIB entry as arguments.
```

9.3.6.1 ldp_ds_check_changes(): New function

Arguments:

Pointer to the LIB entry corresponding to a newly established BE LSP

Returns:

TRUE upon successful completion

Description: Pseudo-code looks like

```
for
        every DS_NODE present in the DiffServ database
do
        for
            every PS DS_RECORD within DS_NODE
        do
            while
                ds_exact_match() with the destination IP
```

```
                address and subnet mask specified in each
                DS_RECORD as arguments is not NULL
            do
                if
                    pointer in RT_HEAD entry returned by
                    ds_exact_match() corresponding to LIB
                    points to default entry
                then
                    call ldp_lsp_add () with pointer to RT
                    and PS as traffic class argument.
        for
            every AS DS_RECORD within DS_NODE
        do
            while
                ds_exact_match() with the destination IP
                address and subnet mask specified in each
                DS_RECORD as arguments is not NULL
            do
                if
                    pointer in RT_HEAD entry returned by
                    ds_exact_match() corresponding to LIB
                    points to default entry
                then
                    call ldp_lsp_add () with pointer to RT
                    and AS as traffic class argument.
```

9.4 INTER-MODULE COMMUNICATION

Communication among the three modules: LDP, DiffServ, and the Interface Module is implemented using function calls. Other implementation alternatives are feasible, e.g. using a separate process for each module and some kind of inter-process communication using messages or events. The purpose of this implementation is only providing a proof of concept and no performance issues have been considered in detail. The design choice made was considered preferable because it is simpler and it requires no additional processes to be maintained.

9.5 SYSTEM BEHAVIOUR

In this section, it is shown how the functions described previously are used and how they interact with one another.

9.5.1 A user logs in

When a user tries to log in the following events take place:

1. The edge router using the RADIUS client (which received user login name and password) contacts the RADIUS server (repository of authorized user information), which performs authentication of the user and returns an Access-Accept packet (it the authentication procedure is successful) with the user's DS profile piggybacked into it.

2. RADIUS client decodes the user profile strings from RADIUS Access-Accept packet, builds linked lists, and passes an event: DS_NODE_ADDED to the DS classifier, with the following arguments:

- source IP address
- pointer to Premium Service linked list
- pointer to Assured Service linked list

3. DS module allocates memory to hold the structures and builds a DS_NODE (see figure 36) structure

4. DS module calls **ds_insertDSnode ()** (see 9.3.2.2 for a description of the function) with a pointer to the newly built data structure ds_node.

5. ds_insertDSnode () inserts the new node as a RB sub-tree in the DiffServ database and it calls **ldp_ds_check_setup ()** with a pointer to the RB tree node

6. ldp_ds_check-setup () checks whether it is necessary to setup new LSPs to provide the services subscribed to by the new user (see 9.3.4.1 for a detailed description of how it works)

9.5.2 User logs off

When a user disconnects from the network, the following events occur:

1. His user profile is removed from the DiffServ database. A DS_NODE_REMOVED event is passed to the DS classifier.

2. The DS classifier calls the function: **ds_deleteDSnode ()** with the user's IP address as an argument. This function removes entry. It has been modified to call **ldp_ds_check_teardown ()** before removal.

3. ldp_ds_check_teardown is called by ds_deleteDSnode() before removing an entry. It checks whether network configuration must be changed.

9.5.3 LSP established

Upon successful establishment of a non-BE LSP, the following events take place:

1. LDP will receive the correct label binding and call the function **ldp_bind_receive()** which among other things will update the routing table with the binding information.

2. This function has been modified to call: **ldp_ds_update_dsd ()** with the LIB entry address as an argument and ADDED_LSP as second argument, if the LSP established is non-BE and **ldp_ds_check_changes()** if the LSP established is BE.

9.5.4 PS/AS torn down

When a non-BE LSP is torn down:

1. LDP receives a label_release message or decides to release a binding.

2. To erase the LIB entry, the function **ldp_lib_delete**() is called with a pointer to the LIB entry.

3. This function has been modified to call **ldp_ds_update_dsd** () with a pointer to the LIB entry and REMOVED_LSP (1) as second argument

Uppgjord (även faktaansvarig om annan) - *Prepared (also subject responsible if other)*		Nr - *No.*		
ETX/D/A Pablo Cebrian				
Dokansv/Godkänd - *Doc respons/Approved*	Kontr - *Checked*	Datum - *Date*	Rev	File
ETX/D/A Eric Lin		1999-10-25	PA1	report.fm

9.6 IMPLEMENTATION TEST

Figure 39 below shows the scenario used to test the implementation:

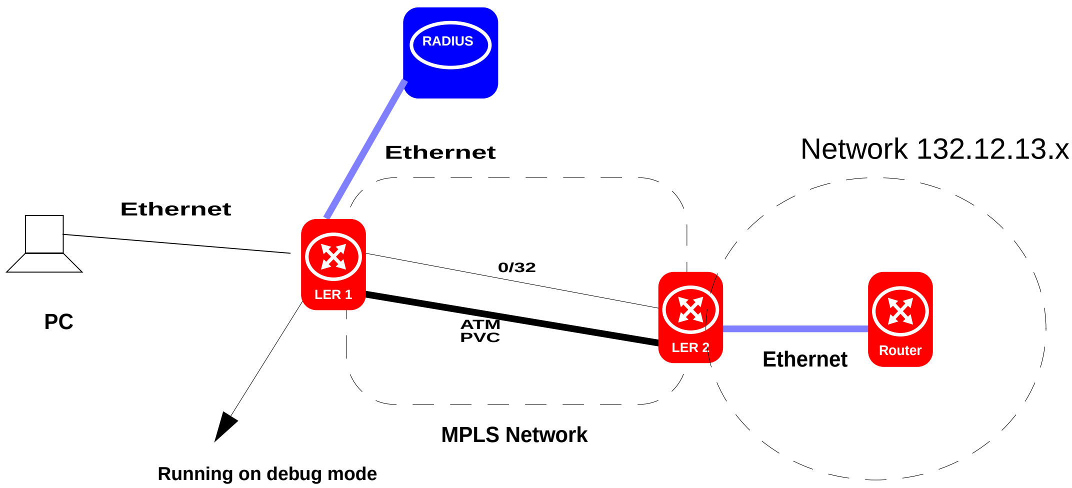


Figure 39 - Test scenario

- Using debug mode, dumps of the LIB were used to verify that an LSP had actually been set up or torn down.
- Two edge routers connected through the ATM interfaces and running LDP simulated the MPLS network.
- The RADIUS server installed on a PC running Windows NT and a PC was connected to the Ethernet port of the LER1 (running a RADIUS client) to simulate a user logging in and off.
- An entry for a fictitious user (PC) was inserted in the RADIUS server's database, specifying password and traffic preferences. PS and AS for all packets going to the subnet linked off the egress router (132.12.13) were requested.
- Another edge router was connected to the Ethernet interface of the egress router to simulate a destination network
- All three routers were running OSPF

Three events were simulated to test the functionality of the user profile driven Solution:

- PC-user logs in

- PC-user logs off
- Subnet goes down and comes back up

9.6.1 Event: User logs in

Simulation:

The web interface implemented in the Edge router was used to log in

Effect:

LIB entries corresponding to PS and AS LSPs and routing table entry 132.12.13 appear in the LIB

9.6.2 Event: User logs off

Simulation:

The user disconnects the PC, terminating the session.

Effect:

LIB entries corresponding to PS and AS LSPs are not present in the LIB any longer

9.6.3 Event: a network goes down

Simulation:

The router connected to the egress edge router is unplugged from the Ethernet. This stops the flow of OSPF Hello packets and causes the routing table at LER1 to be modified.

Effect:

All LIB entries except the default 0/32 are removed

9.6.4 Event: network comes back up

Simulation:

The router is connected to the Ethernet again. OSPF rebuilds routing table

Effect:

Entries corresponding to BE, PS, and AS LSPs to subnet 132.12.13 are again present in the LIB

10 REFERENCES

[1] Ross Callon, George Swallow, N.Feldman, A. Viswanathan, P. Doolan, A.Fredette, "A Framework for Multiprotocol Label Switching", Internet Draft, November 1997.

[2] Ross Callon, A. Viswanathan, E. Rosen, "Multiprotocol Label Switching Architecture", Internet Draft, April 1999.

[3] Michael O'Dell, Joseph Malcolm, Johnson Agogbua, Daniel Awduche, Jim McManus, "Requirements for Traffic Engineering over MPLS", Internet Draft, August 1998.

[4] Bob Thomas, N. Feldman, P. Doolan, Loa Anderson, A. Fredette, "LDP Specification", Internet Draft, May 1999.

[5] Keith McCloghrie, Bruce Davie, George Swallow, Y Rekhter, E. Rosen, P. Doolan, Jeremy Lawrence, "MPLS using ATM VC Switching", Internet Draft, November 1998.

[6] Loa Anderson, Tove Madsen, Andre Fredette, "MPLS solutions for high capacity IP networks", Ericsson IP White Papers No.2, March 1998.

[7] Loa Anderson, Tove Madsen, Göran Hågård, Bengt Nordberg, "Overview of IP over ATM technologies", Ericsson IP White Papers No.3, April 1998.

[8] Yoram Bernet et al. " A Framework for Differentiated Services", Internet Draft, October 1998.

[9] K. Nichols, S. Blake, F. Baker, D. Black, "Definition of the Differentiated Services Field (DiffServ Field) in the IPv4 and IPv6 Headers", Request for Comments 2474, December 1998.

[10] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, W. Weiss, "An Architecture for Differentiated Services", Request for Comments 2475, December 1998.

[11] C.Rigney et al., "Remote Authentication Dial In Service", RFC 2138, April 1997.

[12] B. Braden, et. al., "Recommendations on Queue Management and Congestion Avoidance in the Internet", Request for Comments 2309, April 1998.

[13] Xipeng Xiao, Lionel M. Ni, "Internet QoS Big Picture",

[14] Vijay P. Kumar, T. V. Lakshman, and Dimitrios Stiliadis, "Beyond Best Effort: Router Architectures for the Differentiated Services of Tomorrow's

Internet", IEEE Communications Magazine, May 1998.

[15] Andrew Campbell, Cristiona Aurrecoechea, Linda Hauw, "A Review of QoS Architectures", Center of Telecommunication Research, Colombia University, NY.

[16] S. Keshav, Rosen Sharma, "Issues and Trends in Router Design", IEEE Communications Magazine, May 1998.

[17] K. Nichols, V. Jacobson, L. Zhang, "A Two-bit Differentiated Services Architecture for the Internet", Internet Draft, November 1997

[18] Ashley Stephenson, "DiffServ and MPLS: A Quality Choice", Data Communications, November 1998.

[19] Göran Hågård and Mikael Wolf, "Multiprotocol Label Switching in ATM networks", Ericsson Review No.1, 1998.

[20] A. Vishwanathan et. al., "Evolution of Multiprotocol Label Switching",IEEE Communications Magazine, May 1998.

[21] Siamack Ayandeh, Yanhe Fan, " MPLS Routing Dynamics", Internet Draft, March 1998

[22] Juha Heinanen, "Differentiated Services in MPLS Networks", Internet Draft, June 1999

[23] Liwen Wu et al., "MPLS Support of Differentiated Services by ATM LSRs and Frame Relay LSRs", Internet Draft, March 1999.

[24] Shahram Davari et al., " MPLS Support of Differentiated Services over PPP links", Internet Draft, April 1999.

[25] Bruce Davie et al., " MPLS using LDP and ATM VC Switching", Internet Draft, April 1999.

[26] Ulyess Black, "ATM Volume 1: Foundation for Broadband Networks", Prentice Hall

[27] Anthony Alles, "ATM Internetworking", Cisco Systems, May 1995.

[28] Chuck Semeria, "Multiprotocol Label Switching: Enhancing Routing in the New Public Network", March 1999

[29] B. Braden et al. , "Resource Reservation Protocol (RSVP) -- Version 1 Functional Specification", Internet RFC 2205, September 1997.

[30] Y. Bernet et al., "Requirements of Diff-Serv Boundary Routers", Internet

Draft, November 1998.

[31] Advanced Computer Communications Internal Documents