CHAPTER

MULTIPROTOCOL LABEL SWITCHING (MPLS) 22

If I have seen further than others, it is by standing upon the shoulders of giants.
Isaac Newton

READING GUIDELINE

Understanding the material presented in this chapter requires basic knowledge of packet switching, circuit switching, and routing protocols. In addition, the background presented in the chapter on quality-of-service routing is helpful. This chapter mainly focuses on the protocol aspects of MPLS/GMPLS; its traffic engineering aspects and implications are discussed in Chapter 23.

In this chapter, we present Multiprotocol Label Switching (MPLS). We discuss its benefits to routing and traffic engineering. MPLS is designed for packet switched arbitrary rate services. An additional framework, generalized multiprotocol label switching (GMPLS), is well suited for circuit-oriented periodic or on-demand services with bidirectional requirements, especially for the optical domain. We also discuss extensions required in routing protocols that can be useful in MPLS/GMPLS environments. MPLS is sometimes used in a "vanilla" mode with IP where it is used to replace IP table lookup through accelerated switching, while IP layer still uses IP routing. On the other hand, a more sophisticated use of MPLS is when it is used for traffic engineering of MPLS networks (such as for virtual private networks), which is covered in Chapter 23. In this chapter, we capture the basic features of MPLS and GMPLS.

22.1 BACKGROUND

We start by reviewing the IP routing process. At a router, the routing table is determined, from which the forwarding table is built; when a packet arrives, the IP address of the destination is looked up and mapped against the forwarding table to identify the appropriate routing interface. The underlying principle is to operate on a packet-by-packet basis, while tracking a microflow and forwarding packets on the same path is possible. However, there are situations when controlling the flow of packets for a class of traffic streams is desirable, not just for a single packet or a microflow. In an IP environment, IP traffic engineering that involves link weight determination in an OSPF/IS–IS environment can direct overall flow control; we have previously discussed IP traffic engineering in Chapter 7. However, in IP traffic engineering, all packets for a destination follow the same path due to a destination-oriented routing paradigm of IP routing; certainly, if equal cost paths are found, traffic can take two or more different paths to the destination. However, having a well-defined control mechanism that allows packets to take different paths to a particular destination (e.g., depending on the type of packet or the class of traffic) is desirable. This can be thought of as *traffic engineering with more knobs or controls*, to distinguish it from traditional IP traffic engineering based on a link-weight setting. Nowadays, some service level control can be accomplished if multitopology routing in OSPF/IS–IS (refer to Section 6.3) is invoked by assigning separate sets of link weights for each of the multiple topologies, where each topology is defined for a service class.

To attain the ability to define a path and force packets for a particular traffic stream, class, or affinity to take this path, it is imperative that a mechanism is needed that allows the path to be defined/identified *first*, independent of designated packets traversing this path. The question then is how to define such paths. It may be noted that the notion of defining a path is done in a voice telephone network where call set-up is done first using, for example, ISUP signaling. Here, instead, we are interested in having a mechanism that can work for packet traffic, but that is not necessarily meant to work on a per microflow or per-call basis. There is an important point to be noted here—if we were to look for a mechanism to work on a per-call basis that can also work in an IP environment, a session initiation protocol (SIP) can be considered; SIP is, however, at the end-to-end basis. Rather, we are interested in a mechanism that is on a router-to-router basis. Thus, it should be clear that to set up a path between two routers, a mechanism for signaling such a set-up is certainly required that is not necessarily on a per microflow basis. Second, if part of the goal is to do traffic engineering with more knobs/controls, the state of the network still needs to be communicated among the routers.

As we know, one way to communicate the state of the network is to communicate link state information through a link state routing protocol. But standard link state protocols defined for IP networks carry only a single metric value for each link, typically to represent the cost of the link. For traffic engineering with more knobs/controls, additional information, such as the bandwidth of a link, must be communicated. This means that to learn about the state of a network, a link state protocol paradigm can still be used but with additional information about each link. Finally, to traffic/network engineer a network, it should be possible to invoke different routing path computation frameworks, which is preferably decoupled from the link state update mechanism; such a path communication framework may also depend on the specific operational use and requirements of a network for the services it provides (for example, refer to Table 21.1).

In addition to packet-based traffic, the need for a traffic engineering framework is also felt for circuit-mode connections for services provided in a modular switching environment such as optical wavelength switching or time-division multiplexed switching.

With the above background, we now discuss enabling environments that allow signaling set-up and traffic engineering with more knobs/controls to be included.

22.2 TRAFFIC ENGINEERING EXTENSION TO ROUTING PROTOCOLS

We can see from earlier discussions that at a minimum, the bandwidth of a link must be communicated for the purpose of traffic engineering. In addition, a link may allow higher reservable bandwidth than the announced bandwidth due to statistical multiplexing gain for certain types of traffic, meaning that oversubscription may be tolerable. Also, a link may announce currently unreserved bandwidth which is useful for routing path computations, but which is not necessarily based on a shortest path computation. Since a network may provide more than one type of prioritized services, it would be useful to announce the unreserved bandwidth allowed for each priority class. Also, a network provider might want to use a different metric, other than the standard link metric; this link metric might have meaning that is internal only to the provider. To summarize, the following attributes of a link are desirable:

- Maximum link bandwidth that is usable
- Maximum reservation bandwidth in case oversubscription is allowed
- Unreserved bandwidth available at different priority levels
- Traffic engineering metric.

The question is: how is this information communicated? This is where two popularly deployed link state routing protocols are OSPF and IS–IS, presented in Chapter 6, come into the picture. These two protocols have been extended to allow for communication of the above attributes. The actual extensions are somewhat different in GMPLS compared to MPLS due to additional requirements in circuit-mode connections. We will cover them as we introduce MPLS and GMPLS in the following sections; specifically, refer to Section 22.3.4, and Section 22.4.4, respectively.

22.3 MULTIPROTOCOL LABEL SWITCHING (MPLS)

Multiprotocol Label Switching (MPLS) is a mechanism that addresses several issues discussed above; it is meant for packet-based services. Briefly, MPLS adds a *label* in front of a packet, i.e., as another header so that routers know how to act based on this label. Adding a label in MPLS is commonly referred to as an MPLS *push* function. Similarly, removing a label in MPLS is commonly referred to as an MPLS *pop* function. An MPLS *swap* function means that a label is exchanged or replaced. To be able to act based on a label, routers must be *label-switched routers* (LSRs) and each LSR must maintain a valid mapping from the label of an incoming packet to a label ("incoming label") to be attached to packet before being sent out ("output label"). This, in turn, means that LSRs maintain states in terms of input/output labels associated with a particular path, referred to as a *label-switched path* (LSP), which may be designated for a particular class of traffic flows. Note that an LSP must already be established between two routers so that packets can follow this path. To establish a path, a *label distribution protocol* is used. Certainly, the next question then is: how do we know this is the best path for the particular class of traffic flows? This will depend on the traffic engineering requirements of the network, and on the service requirements of the traffic flow to be carried by the LSP. In this section, we present the basic foundation for MPLS and the enabler for traffic engineering in MPLS; in Chapter 23, we present examples of routing and traffic engineering design using MPLS networks. This is also a good place to briefly comment on "basic" MPLS that is used with IP; in this case, MPLS is simply used to replace IP table lookup and to accelerate switching, but not for creating any MPLS path. On the other hand, when MPLS is used with traffic engineering (discussed in Chapter 23), more of the knobs and controls discussed in this chapter are used.

In Figure 22.1, a conceptual architecture of an MPLS label switched router with IP functionality is shown. As can be seen, there are two planes: a control plane and a data plane. At the control plane, IP routing protocols can exchange routing information, and another component manages label distribution and binding. It also maintains a *label information base (LIB)*, and creates a *label forwarding information base (LFIB)*. The packet arriving on the data plane consults the LFIB for proper forwarding in an outgoing direction.

It may be noted that the establishment of an LSP is a connection-oriented functionality—that is, a path must be set up before traffic can use this path. An established LSP may or may not have any packet traffic flow on it; furthermore, the packet traffic flow rate on an LSP can vary from one instant of time to another. Many traffic flows may be combined on a specific LSP; usually, such a flow aggregation is based on some affinity, such as a traffic class. The aggregated flow constructed on some affinity basis is referred to as a *traffic trunk*. Typically, a traffic trunk is defined on an ingress/egress LSR pair basis and is carried on an LSP. Note that all traffic between the same two ingress/egress LSRs may be split into multiple traffic trunks; each traffic trunk is then mapped into an LSP. Thus, a traffic trunk is a logical entity while an LSP is a transport manifestation of this logical entity.

22.3.1 LABELED PACKETS AND LSP

A label in MPLS is 20-bit long and is part of a 32-bit MPLS shim header. A packet with an MPLS shim header will be referred to as an *MPLS packet*. If an MPLS packet is to carry an IP packet, then we can think of MPLS as being placed in layer 2.5 (see Figure 22.2). Note that it still requires the help of layer 2 for packet delivery on a link-by-link basis between two adjacent LSRs; the main difference

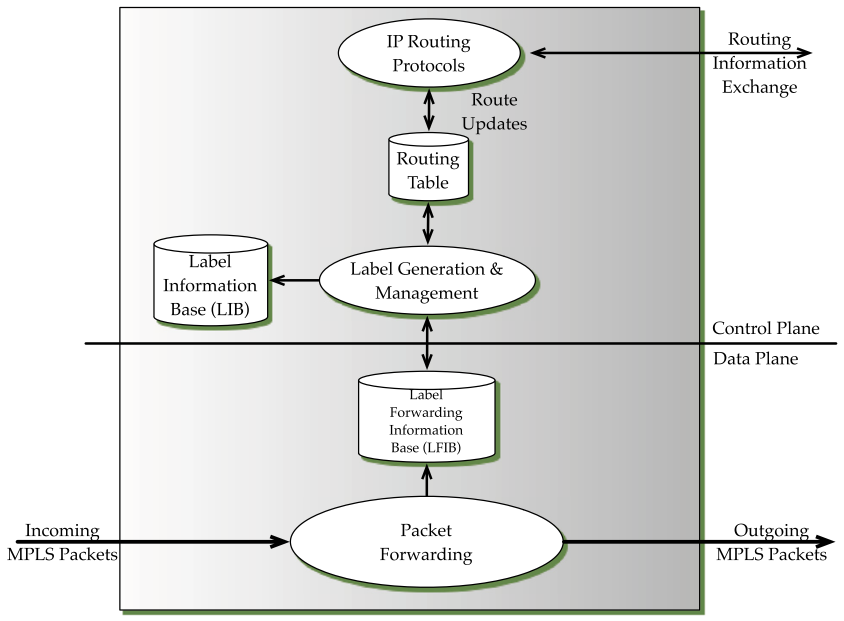


FIGURE 22.1

Conceptual architecture of an MPLS label switched router.

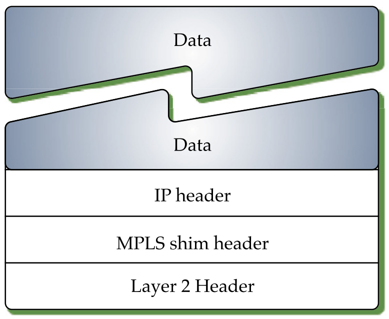


FIGURE 22.2

MPLS header as layer 2.5.

for MPLS in being in layer 2 is that it does provide a form of routing through labels and label-switched paths across multiple hops. Suppose that layer 2 is a Packet over SONET technology between two LSRs. Since PPP is used for Packet over SONET, we have PPP as the layer 2 protocol to deliver an MPLS packet from one LSR to the other.

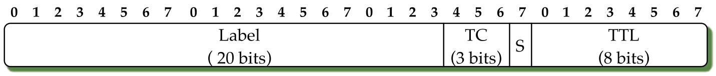


FIGURE 22.3

MPLS shim header.

The 32-bit shim header also includes 3 traffic class (TC) bits, a bit (“S” bit) to indicate this label is the last label (bottom of the stack, “BoS”) in the case of stacked labels, and 8 bits for the time-to-live (TTL) field (see Figure 22.3). The TC bits are used to describe traffic that require different priorities. The label values 0 to 15 are reserved. For example, label 0 (*explicit null label*) refers to a packet that is to be forwarded based on the IPv4 header and is allowed only at the bottom of the label stack. Similarly, label 2 serves as the explicit null label for IPv6. The explicit null label can be used by an egress router to signal its immediate upstream router (“penultimate hop router”). In turn, the egress router receives packets from the penultimate hop router with label value 0 and can also learn any priority information included in the experimental bits that it can use for IP forwarding.

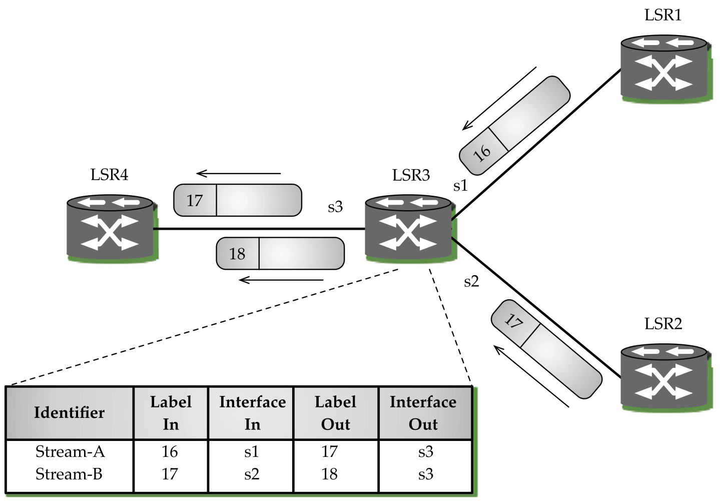
There is another terminology, *tunnel*, closely related to an LSP. A tunnel provides a transport service between two routers so that packets for a specific stream can flow without being label swapped at any intermediate switches or routers. A tunnel in MPLS can be realized by a well-defined label-switched path, often based on serving certain traffic engineering requirements. Thus, typically, such tunnels have longevity. Furthermore, note that tunnel is a generic name used in networking; it is not limited just to MPLS.

When an MPLS packet arrives at an LSR, the incoming label is swapped with an outgoing label; this assumes that a label-switched path is already defined and lookup tables at LSRs have appropriate entries. Before sending the MPLS packet out to the next hop, the TTL value is decreased by one. If the TTL value becomes zero, then the MPLS packet is to be dropped. Since the TTL field is 8 bits long, the likelihood of a path having more than 255 hops is zero. Consider Figure 22.4. Here an MPLS packet with label 16 arrives at LSR3 from LSR1 and is swapped with label 17 for transmittal to LSR4; similarly, the MPLS packet with label 17 that arrives from LSR2 at LSR3 is swapped with label 18 for transmittal to LSR4. In this case, the assumption is that two label switched paths, LSR1-LSR3-LSR4 and LSR2-LSR3-LSR4, are already defined.

It may be noted that MPLS also allows stacked labels. A stack tag means that an MPLS shim header may appear more than once; each one is related to a particular LSP between certain points. Thus, a stacked MPLS packet has the following form:

Layer-2 header | MPLS shim header | MPLS shim header | Data

Consider Figure 22.5. Here, there is already an LSP set up from LSR3 to LSR5; this is referred to as an LSP tunnel. Now consider two LSPs, one from LSR1 to LSR6 and the other from LSR2 to LSR7, that use the LSR3-LSR5 tunnel as a logical link on their paths. For illustration, consider the LSP from LSR1 to LSR6. The packet with label 16 arrives at LSR3 and is swapped with label 17; due to the LSP tunnel LSR3-LSR5, an additional label 21 will be added—the “S” bit for this label would not be set. When this MPLS packet arrives at LSR4, the top label 21 will be swapped to 42—this means that it is as if LSR4 is thinking about the LSP tunnel LSR3-LSR4 and does not care about any labels in the



Identifier	Label In	Interface In	Label Out	Interface Out
Stream-A	16	s1	17	s3
Stream-B	17	s2	18	s3

FIGURE 22.4

Label swapping and label switched paths.

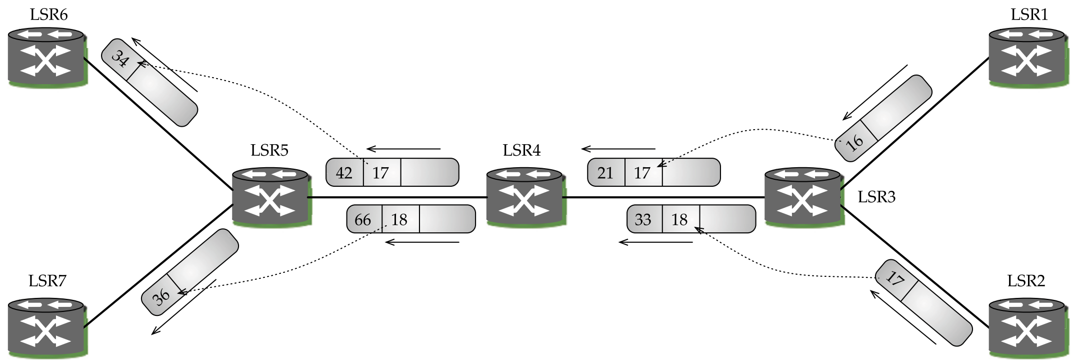


FIGURE 22.5

Label-switched paths using an LSP tunnel (“tunnel within a tunnel”).

MPLS packet. When this packet arrives at LSR5, the role of the label with value 42 will end here since this is the end of the tunnel; because of LSP path LSR1-LSR6, the second label 17 will be swapped to 34 and will be forwarded to LSR6. Similarly, the packet from LSR2 to LSR7 will be handled. There is an important point to note here: having an already established LSP tunnel between two routers does not imply that all LSPs that traverse this tunnel need to have the same label value. Now, if the LSP from LSR1 to LSR6 is established as long-lived, it will serve as a tunnel for other traffic; thus, we arrive at a scenario known as a *tunnel within a tunnel*. Note that we have described only the functionality—when

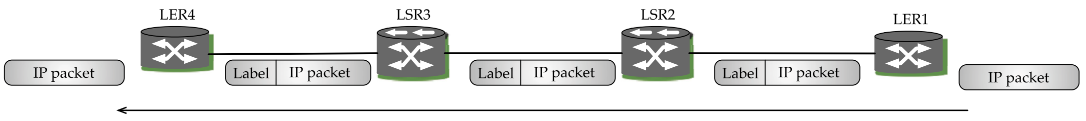


FIGURE 22.6

Label-edge routers and label-switched routers.

and where to establish LSP tunnels in a network to encapsulate other LSPs that can be driven by traffic engineering decisions.

Note that examples discussed so far do not show how MPLS receives an IP packet. An MPLS network must have *edge* routers, which are the points where a native IP packet is prepended with an MPLS label; these routers are known as *label edge routers* (LER). This is shown in Figure 22.6.

Another concept that is associated with an LSP is a *forwarding equivalence class* (FEC). In a network, an FEC streamlines packets based on, for example, traffic engineering requirements. Thus, an FEC must then have an association with at least one LSP in the MPLS network so that packets for this FEC have an actual path to follow along with any QoS considerations. That is, an FEC does not define a path; one or more LSPs are used for carrying packets for an FEC.

MPLS uses *Next Hop Label Forwarding Entry* (NHLFE) when forwarding a labeled packet. The following information is included in NHLFE: 1) the next hop of the packet, 2) which operation to perform such as replacing the label or popping the label stack.

Finally, we know that all packets are eventually sent as Ethernet (Layer-2) frames in a network. How does an incoming packet at an arriving router determine that it is indeed an MPLS packet? In other words, at a receiving router, the processing function must be able to distinguish an MPLS packet from, say, an IP packet. To be able to distinguish, for an MPLS unicast packet, the EtherType (a 16-bit field) in the IEEE 802 Ethernet frame is set to 0×8847 while for an MPLS multicast packet, the EtherType in the frame is set to 0×8848. You might want to note that the EtherType for an IPv4 packet is set to 0×0800 and for an IPv6 packet to $0 \times 86DD$ [391]. Thus, when the ethernet frame is constructed at the sending end, this information is populated in the EtherType field so that the receiving end can properly process it.

22.3.2 LABEL DISTRIBUTION

Between defining an FEC and establishing an LSP for this FEC, there is another very important phase known as label distribution. The basic idea of label distribution is analogous to route information exchange, somewhat similar to BGP. In BGP, two BGP speakers exchange IP prefix reachability information while in MPLS two LSRs exchange label-related information. A key difference is that while a BGP speaker computes and determines an outgoing routing decision, in MPLS, an LSR essentially trusts that the label information is based on a valid LSP and does only a sanity check with its appropriate neighbor about label mapping; note that label mapping to an LSP may be generated by an outside entity, such as a traffic engineering manager. So that labels distributed can be associated with LSRs, each LSR must have an identifier (LSR ID) that must be unique in the MPLS domain which it is part of. Typically, a router's IP address serves as the LSR ID.

In MPLS, *label binding* refers to directly associating a specific label to a specific FEC. Consider two LSRs, LSR-u and LSR-d, that have agreed to bind a specific label, for packets that will be sent from LSR-u to LSR-d; in regard to this label binding, LSR-u is referred to as the *upstream LSR* and LSR-d as the *downstream LSR*. The decision to bind a label is made by the LSR, which is downstream with respect to this binding. The downstream LSR then informs the upstream LSR of the binding. We can then say that labels are *assigned* from upstream to downstream while label bindings are *communicated* from downstream to upstream. In MPLS, the distribution of labels for label binding can be accomplished through a *label distribution protocol* (LDP) [33]. Two approaches for label distribution are possible in the LDP paradigm in MPLS: in the *downstream on demand* approach, a downstream LSR can distribute an FEC label binding when it receives a request explicitly from an upstream LSR on demand; in the *downstream unsolicited* approach, an LSR can distribute label bindings to LSRs without receiving an explicit request.

In the terminology of LDP, two LSRs that can exchange label/FEC mapping information are referred to as *LDP peers* using a bidirectional *LDP session*. TCP is used for setting an LDP session. The question is how. Do they need to set up an LSP first to exchange this information? That is, is there a chicken and egg problem? Fortunately, no. Recall that an MPLS router actually serves in dual-mode, IP for control plane and MPLS for data plane. Thus, two adjacent LSRs can use the IP only-mode to set up this TCP session, bypassing the MPLS plane.

22.3.3 RSVP-TE FOR MPLS

The MPLS framework originally defined the basic specification for a label distribution protocol [33]. Recently, the *Resource ReSerVation Protocol with Traffic Engineering extension* (RSVP-TE, in short) has become the *de facto* label distribution protocol for the purpose of traffic engineering. Thus, we will focus on RSVP-TE. We first start with a brief overview of RSVP.

Resource ReSerVation Protocol (RSVP): Overview

RSVP is a connection set-up protocol in a packet network. Originally, it was defined in the context of an *integrated services* (int-serv) framework. RSVP is considered a *soft-state* approach; this means that if a reservation refreshing message is not sent periodically for a session that has been set up, the session is torn down after a given interval. Such a soft-state measure is appealing in a networking environment where both best-effort and guaranteed bandwidth services are offered in a connection-less packet mode. RSVP is not scalable when there are many on-going end-to-end conversations on a network due to the number of messages that would be generated. However, for RSVP-TE, the set up messages are generated only for LSPs—this number is much smaller than if used for signaling of end-to-end sessions; furthermore, RSVP refresh reduction is possible in RSVP-TE. Here, we first highlight a few key elements about RSVP that are applicable to RSVP-TE.

All RSVP messages have a common header (see Figure 22.7). A key field in the header is the message type. Originally, seven main message types have been defined: Path, Resv, PathErr, ResvErr, PathTear, ResvTear, and ResvConf; they are used in regard to connection set-up and connection tear-down. Additional types have been added for a variety of purposes, see [398] for an updated list. However, discussion of all of them is outside the scope of this book.

An RSVP message includes one or more RSVP objects, with each object having a certain significance in regard to a specific message; it may be noted that some objects can be optional. An RSVP

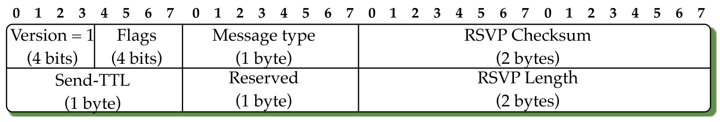


FIGURE 22.7

RSVP common header.

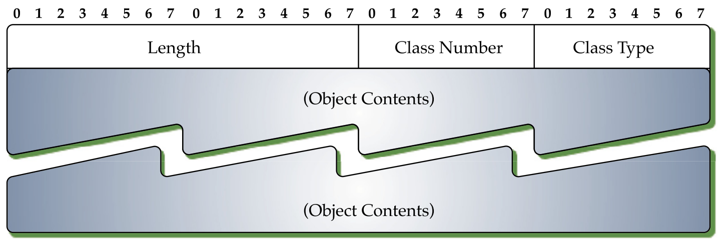


FIGURE 22.8

RSVP object format.

```
⟨Path Message⟩ ::=          ⟨Common Header⟩ [ ⟨INTEGRITY⟩ ]
                            ⟨SESSION⟩ ⟨RSVP-HOP⟩
                            ⟨TIME-VALUES⟩
                            [ ⟨EXPLICIT-ROUTE⟩ ]
                            ⟨LABEL-REQUEST⟩
                            [ ⟨SESSION-ATTRIBUTE⟩ ]
                            [ ⟨POLICY-DATA⟩ ... ]
                            ⟨sender descriptor⟩

⟨sender descriptor⟩ ::=     ⟨SENDER-TEMPLATE⟩ ⟨SENDER-TSPEC⟩
                            [ ⟨ADSPEC⟩ ]
                            [ ⟨RECORD-ROUTE⟩ ]
```

FIGURE 22.9

RSVP path message in Backus–Naur form.

object consists of four fields (see Figure 22.8): Length (16 bits), Class Number (8 bits), Class Type (8-bits), and Object Contents (variable). Length is the total object length in bytes, in multiples of 4 bytes. Class Number (Class-Num) identifies an object class, while Class Type (C-Type) identifies unique information within a class.

In Figure 22.9, an RSVP Path message is shown using a notation known as a *Backus–Naur Form* (BNF). Objects are specified using angle brackets, such as "<SESSION>"; an optional object is enclosed in square brackets, such as "[<INTEGRITY>]." A molecular object can be formed from atomic objects, which in turn can be part of a more complex object. The notation "::=" is used to separate the name of an object on the left side that is defined by the set of objects on the right side. It may be noted that the Path message example shown in Figure 22.9 is actually for RSVP-TE, not for the original version of RSVP; this is included for ease of our discussion in regard to traffic engineering.

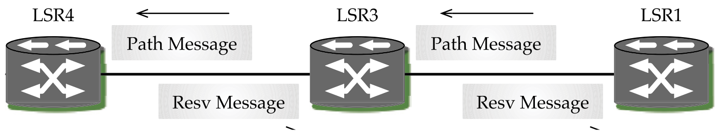


FIGURE 22.10

Example: label distribution using RSVP-TE.

RSVP-TE

RSVP-TE, is the extension to RSVP, which has been developed for use in establishing LSPs, particularly geared to traffic engineering. The label distribution approach in RSVP-TE is based on the downstream on demand approach. There are certain key differences between RSVP-TE and the original RSVP. For example, RSVP-TE is used for signaling between routers to set up LSP flows, unlike the original RSVP that was used between hosts to set up microflows. Because of this, RSVP-TE does not have the scalability problem that RSVP faces in regard to management of microflows. RSVP-TE is used to set up directional unicast LSPs, while RSVP allows multicast flow set-ups.

Traffic trunks, discussed earlier, can be carried in RSVP-TE-defined LSPs. A traffic trunk may be split on two LSPs established from an ingress LER to an egress LER, or multiple traffic trunks may be combined to be carried on a single LSP. Thus, such LSPs serve as traffic engineering tunnels. Note that RSVP-TE does not dictate how to decide or when to create different traffic trunks or when to split a traffic trunk into multiple LSPs; rather, the role of RSVP-TE is that of an *enabler* from a functional point of view while the actual decision is left to operational network providers.

RSVP allows three service-type specifications that can be used by RSVP-TE when setting up LSPs. The service types described within the purview of int-serv are Guaranteed Quality-of-Service [751], Controlled-load Service [872], and Null Service [100]. For example, if the tunnel requires a bandwidth guarantee, the Peak Data Rate parameter is specified and that it is a guaranteed service request. In case of Null Service, it need not specify resource requirements. However, controlled-load service can provide a full guarantee if the network is under little congestion; but in case of congestion, some delay would be experienced by packets.

Consider Figure 22.10 where we want to set up the traffic engineering tunnel LSR1-LSR3-LSR4; this example is a snapshot of the LSP shown earlier in Figure 22.4. Here the Path message generated at LSR1 is renewed at LSR3 for destination LSR4. LSR4, being the destination, generates the Resv message in the reverse direction to LSR3, which in turn forwards it to LSR1. Thus, in RSVP-TE, the Resv message then helps accomplish the label binding function.

What does the Path message include when it is initially generated by a router such as LSR1? Also, what specifics are key when it is used for RSVP-TE? In general, the Path message generated at LSR1 will include objects as shown in Figure 22.9; it can contain certain key objects:

- SESSION: It identifies the session type through a C-Type field set to LSP-TUNNEL-IPv4 and includes the IP address of the destination.
- LABEL-REQUEST: This indicates that a label binding is requested.
- RSVP-HOP: This indicates the IP address of the sending RSVP-capable router, and the outgoing interface.

- EXPLICIT-ROUTE: This is included to specify that this particular path LSR1-LSR3-LSR4 is to be followed; However, RSVP-TE allows an intermediate node to modify this EXPLICIT-ROUTE object to allow for any local rerouting; in this case, both the original and the modified EXPLICIT-ROUTE objects are stored. Certainly, any such local rerouting assumes that the intermediate router is somehow aware of this reroute. Note that such a local reroute is essentially a crankback feature.
- RECORD-ROUTE: This serves as a form of acknowledgment so that the sending node knows if the path specified was the actual route taken. The visited routers are added as a subobject to the RECORD-ROUTE as the Path message is being forwarded downstream.
- SESSION-ATTRIBUTE: This is included for the purpose of session identification as well as troubleshooting. Set-up priorities and holding priorities are included here.
- SENDER-TEMPLATE: This is used primarily to announce the IP address of the sender along with an LSP identifier,
- ADSPEC: This is advertising information that may be carried in a Path. This information is passed to the local traffic control at a router that returns an updated ADSPEC, which, in turn, is forwarded in Path messages that are sent downstream.
- SENDER-TSPEC: The traffic characteristics of the tunnel are defined through this object that uses int-serv as C-Type. This field contains Token bucket rate (r in bytes/sec), Token bucket size (b in bytes), Peak data rate (p in bytes/sec), Minimum policed unit (m in bytes), and Maximum packet size (M in bytes). The service option can be specified as either Guaranteed QoS, Controlled-Load Service, or Null Service.

Similarly, the Resv message includes SESSION, RSVP-HOP, TIME VALUES, STYLE, FLOWSPEC, FILTERSPEC, and LABEL. There is an important connection between quality of service, SENDER-TSPEC, and FLOWSPEC. Note that RSVP-TE is specifically designed to setup QoS-enabled LSPs, where QoS parameters may include diffserv parameters. QoS is specified using the SENDER-TSPEC object; The egress node, in return, creates and sends the FLOWSPEC object.

In addition to Path and Resv messages, there are other message types as well; for example, the PathErr message is generated if an LSP tunnel cannot be established at any of the intermediate routers; the PathTear message is used to tear down an LSP session. Furthermore, a new optional message type, Hello, has been introduced to determine if an adjacent LSR is reachable.

Each RSVP object, shown in Figure 22.8, must contain a valid and unique class number and a class-type. Relevant objects that are included in an RSVP Path message are shown in Figure 22.9. To address any future requirements, the ability to add new objects is possible by defining new class numbers. In Table 22.1, a representative set of objects with the class number and class type is listed; an up-to-date list is maintained at [398].

You may note that RSVP-TE Path and Resv messages can contain many parameters, and sub-parameters. In fact, if we were to print a Path message with each field listed separately, it can run to several pages. Instead, you might want to consult [869], in which samples of RSVP-TE messages are available—they are very helpful in understanding RSVP-TE messages. Below, we briefly illustrate the important contents of the SENDER-TSPEC object.

Example 22.1. Illustration of traffic characteristics of SENDER-TSPEC.

As mentioned earlier, traffic characteristics have three key parameters: Token bucket rate (r in bytes/sec), Token bucket size (b in bytes), and Peak data rate (p in bytes/sec). For the concept of token bucket, refer to Chapter 18.

Table 22.1 RSVP object examples for MPLS (an up-to-date list is maintained at [398]).

Object Name	Used in	Class Number	Examples: Class-type with value in parentheses (source RFC listed as [] from bibliography)
INTEGRITY	Path, Resv	4	RSVP Integrity (1) [68]
SESSION	Path, Resv	1	LSP Tunnel for IPv4 (7) [61]
RSVP-HOP	Path, Resv	3	IPv4 (1), IPv6 (2) [121]
TIME-VALUES	Path, Resv	5	Time Value (1) [121]
FILTER-SPEC	Resv	10	LSP Tunnel for IPv4 (7) [61]
SENDER-TEMPLATE	Path	11	LSP Tunnel for IP4 (7) [61]
*RSVP-LABEL	Path	16	Type 1 Label (1) [61]; Generalized Label (2) [99]
*LABEL-REQUEST	Path	19	No label range (1) [61]; generalized label request (4) [99]
*EXPLICIT-ROUTE	Path	20	Type 1 Explicit Route (1) [61]
POLICY-DATA	Path	14	Type 1 (1) [121]
SENDER-TSPEC	Path	12	Integrated Services (1) [121]
*RECORD-ROUTE	Path	21	IPv4 (1) [61]
*SESSION-ATTRIBUTE	Path	207	LSP Tunnel (7) [61]
DETOUR	Path	63	IPv4 (7) [639]
FAST-REROUTE	Path	205	Type 1 (1) [639]
UPSTREAM-LABEL	Path	35	Same as in RSVP-LABEL

If the data is generated at a steady rate of 625,000 bps to provide guaranteed service, then $r =$ 625,000 bps, and $b = 1$ byte. Note that $b = 1$ means that the token is spent immediately. In this case, the peak rate does not play a role. If however, b is given to be 1000 bytes, then credits can be accumulated to use in a future time slot as long as it is allowed by the peak rate. For example, p is also set at 625,000 bps, then the bucket size value is not meaningful. If however, p is set at 630,000 bps, then even if the token is received at 625,000 bps, not all need to be used up; i.e., it can accumulate 1000 bytes credit for up to five seconds, so that it can transmit at 630,000 bps at the end of the five seconds. ●

An important issue during the label distribution phase is loop detection. This might give the impression that an EXPLICIT-ROUTE, when created at the originating router, will check and provide a loopless path that would be sufficient; the difficulty is that local rerouting along the path may not be ruled out. Thus, RSVP-TE relies on the RECORD-ROUTE object; more specifically, when an intermediate router processes the RECORD-ROUTE object, it checks all subobjects inside this object to see if it is already listed as one of the nodes visited to detect looping.

Once a traffic engineering tunnel is set up, traffic trunks can use them. However, a tunnel might get broken, for example, if one of the links between the intermediate routers goes down. In this case, the MPLS network faces the issue of generating a new tunnel in place of the original tunnel so that traffic trunks have a path to the destination. However, there is a lag time from when a link fails to when a new path is established. This lag time, however, may not be acceptable to customers and services that rely on the network for reliable services. Consider, for example, TCP-based services that are using this link—in fact, in most actual networks, the bulk of the services are TCP-based. From the transport layer protocol perspective, a new TCP connection can be established on the new path after a failure (when the old connection times out). However, this also introduces delay; furthermore, if n TCP sessions are using such a link, $3n$ messages will be generated due to a TCP connection set phase. Such a delay and overhead also impact user perception on service reliability. To nullify such a delay and overhead, MPLS has introduced the concept of *fast reroute*. This means that when a TE tunnel (LSP) using Path message is established, a backup path is also established that is along routers and links that do not belong to the first path; this set up can be on a local basis or on an end-to-end basis between two LERs. To provide this functionality, RSVP-TE has also added two new objects, FAST-REROUTE and DETOUR. To control the backup for a protected LSP, the FAST-REROUTE object is used. The bandwidth to be guaranteed is the value in the bandwidth field of the FAST-REROUTE object. An LSP that is used to re-route traffic around a failure in the one-to-one backup is referred to as a *detour* LPS; then, the DETOUR object is used in the one-to-one backup method to identify detour LSPs.

A question remains on how an RSVP-TE message is sent between two MPLS routers. In many instances, MPLS routers are integrated IP/MPLS routers; thus, an RSVP-TE message is sent over the IP control plane, on a hop-by-hop basis.

To summarize, RSVP-TE supports the following key functionalities: downstream-on-demand label distribution, explicitly routed LSPs, and allocation of network resources. It also allows tracking of the actual route traversed by an LSP and loop detection. Rerouting is possible through fast-reroute.

22.3.4 TRAFFIC ENGINEERING EXTENSIONS TO OSPF AND IS–IS

Earlier in Section 22.2, we highlighted the additional attributes that need to be communicated about a link by the routing protocol. This information is then used by a routing computation module, whether centralized or decentralized, to determine LSPs for traffic trunks.

Here, we summarize the extensions to protocols OSPFv2 and IS–IS for MPLS traffic engineering.

Extensions to OSPFv2

In OSPFv2, several link state advertisement (LSA) types have been already defined. One of them is known as the *opaque LSA* [191], [98] briefly described earlier in Section 6.2.8. The intended use of opaque LSA is to allow a general LSA feature so that it might be useful for any future extensions. With regard to the opaque LSA, three links-state types have been presented for the scope of flooding; they are known as type-9, type-10, and type-11 for local subnet flooding, intra-area flooding, and flooding in the entire autonomous systems, respectively.

For MPLS traffic engineering, opaque LSA type 10 is only used; this limits flooding to an intra-area of an OSPF domain; it is known as a traffic engineering LSA (TE LSA). The TE LSA [439] contains a standard header that includes information such as link state age, advertising router, and link state sequence number; in addition, it uses nested TLV to contain information needed for TE LSA. At the top level, there are two TLVs: (1) a router address TLV and (2) a Link TLV. The Router Address TLV contains the IP address of the advertising router; this is preferably a stable and reachable address such as loopback addressing (refer to Section 9.4).

The link TLV contains several sub-TLVs. The key ones have already been mentioned in Section 22.2: the maximum link bandwidth, maximum reservation bandwidth, unreserved bandwidth available at different priority levels, and traffic engineering metric. Note that IP differentiated services requirements can be mapped to the different priority levels. In addition, the link is identified as to whether it is a point-to-point or a multi-access link.

Extensions to IS–IS

Recall that in IS–IS, each intermediate system, i.e., the router, advertises link state protocol data units (LSPs) that are analogous to LSAs in OSPF. The basic format of an LSP is a fixed header followed by a number of TLV-formated information units. The important point to note is that LSPs already use TLV encoding. Thus, IS–IS extensions for traffic engineering define new TLVs. Most importantly, it replaces the extended IS reachability TLV that was originally defined in the IS–IS protocol. The new TLV types are 22, 134, and 135 (see Table 6.2 in Chapter 6).

The proposed extended IS reachability TLV specifies the following: system ID, default metric, and then through sub-TLVs, the fields needed for traffic engineering such as maximum link and reservable link bandwidths are specified, the same ones as in an OSPF extension.

22.3.5 POINT-TO-MULTIPOINT LSP AND MULTIPOINT-TO-MULTIPOINT LSP

A point-to-multipoint (P2MP) LSP has one ingress LSR (root node) and one or more Egress LSRs (leaf nodes). The LSP setup and teardown are initiated by the leaf nodes. If a P2MP LSP has any transit nodes, they take the role of propagating the setup and tear-down message to the root node. For a P2MP LSP, all nodes that are part of this LSP must maintain their MPLS forwarding state. At the time of traffic delivery, the packets move from the source node to the leaf nodes, for which MPLS labels were already set up using the label distribution protocol. When an incoming MPLS-labeled packet arrives at a transit node, it would need to be replicated for each of the outgoing interfaces that help to reach the destination leaf nodes. While the labels for the outgoing packets can be different from the incoming packets, it may be operationally preferable to have the same label for a particular multicast tree so that it is easy to troubleshoot; this certainly requires a label assignment manager to avoid cross-labeling wrong tunnels for different purposes. P2MP uses the notion of Bud LSRs—they are egress routers that have one more downstream LSR. A new P2MP FEC element has also been defined. This FEC element contains the address of the root and an Opaque value. The Opaque value serves as a tree identifier. The tuple ⟨RootLSRAddress, Opaque Value⟩ uniquely identifies a P2MP LSP.

Before a P2MP LSP can be set up, a reservation would need to be initiated using the RSVP protocol. To allow P2MP functionality, RSVP has been extended to include P2MP SESSION objects: P2MP-LSP-TUNNEL-IPv4 and P2MP-LSP-TUNNEL-IPv6. We now describe this below using a simple illustration that was simplified from an example described in [15].

In Figure 22.11, a P2MP LSP is shown. Here, LSR1 is the ingress LSR and there are three egress LSRs: LSR4, LSR6, LSR7. Because of this, we have three source-to-leaf (S2L) sub-LSPs: sub-LSP-4, sub-LSP-6, sub-LSP-7. They can be specified through EXPLICIT-ROUTE Object (ERO) and P2MP-SECONDARY-EXPLICIT-ROUTE Object (SERO). The ERO to LSR4 will be {LSR2, LSR3, LSR4} if this is the first S2L sub-LSP. Then SERO to LSR6 will be {LSR3, LSR5, LSR6} and SERO to LSR7 will be {LSR5, LSR7}. By specifying in this manner, the Path message size can be compressed. In this illustration, LSR3 and LSR5 are Bud LSRs.

The multipoint-to-multipoint (MP2MP) functionality is needed when in a group of entities, all want to send information to the rest in the group. Thus, when we break it down what is needed of MP2MP,

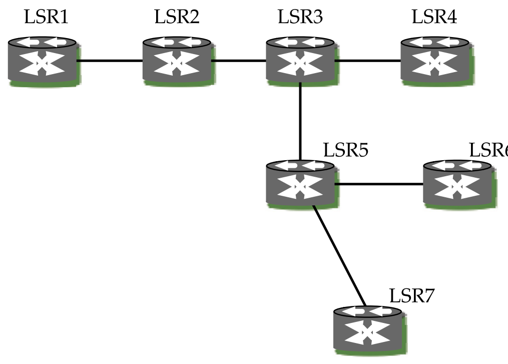


FIGURE 22.11

P2MP Illustration.

it can be accomplished like P2MP. That is, an MP2MP LSP is very much like a P2MP LSP: there is a root node along with leaf nodes and the leaf nodes are both ingress or egress nodes. Thus, MP2MP is made up of a set of P2MP trees, with each leaf node being the head of a P2MP tree. That is, each leaf node plays two roles: one as the originator for its P2MP tree and the other as the receiver for the other leaf nodes' P2MP tree.

22.4 GENERALIZED MPLS (GMPLS)

Generalized MPLS (GMPLS), as the name suggests, is an extension of MPLS. Certainly, the question then is why is such an extension needed and what is it meant for? Recall that MPLS has been designed to *switch* packets using a labeling mechanism. Yet, there is the need for an MPLS control type functionality for controls that is beyond just switching packets, such as wavelength switching, time division multiplexing, and fiber (port) switching; this mode of switching is traditionally referred to as circuit switching or circuit routing since a dedicated path and physical resources must be allocated for a service from one end to another. Note that the term circuit switching is commonly used with telephony voice circuit switching, and is not always the best terminology to use in this generalized context.

It is worth noting that originally GMPLS was intended for wavelength (lambda) switching where a dedicated path is required to be set up for a wavelength path end to end; however, it was soon realized that there is a need for a similar framework that can be used for other switching as well. GMPLS is thus intended for the following four switching capabilities:

- *Packet-Switch Capable (PSC)* for IP packets or ATM cell level switching
- *Time-Division Multiplexing Capable (TDMC)*: for timeslot-based circuit switching
- *Lambda-Switch Capable (LSC):* for wavelength switching at optical cross-connects
- *Fiber-Switch Capable (FSC):* for fiber-level switching at optical cross-connects.

Collectively, for brevity, we will refer to them as *generalized modular switching* (GMS) instead of calling them circuit switching. Note that this is our terminology that is used here for ease of discussion, and to save us, each time, from discussing/mentioning switching for all of the above four technologies. GMS is thus distinguished from the umbrella suite GMPLS; *modular* is used here since in GMPLS, switching can only be on well-defined modular values that are tied to specific technology, for example, OC-3 rate in SONET or T1 for TDM. This certainly makes sense for TDM, LSC, and FSC. What about PSC? This requires some elaboration. Consider Packet over SONET. Here the data stream is coming as IP packets at an *arbitrary* data rate that is then mapped to a specific SONET frame rate through *asynchronous* mapping in an envelope mode; similarly, if IP traffic is to be carried over TDM switching, then PSC capability means that the IP packets coming at an arbitrary data rate are mapped to say a T3 rate for TDM switching. Thus, for PSC, an incoming stream is mapped to a modular value of a data rate, which may not be completely filled.

In essence, GMPLS is an umbrella suite that encompasses signaling and traffic engineering for generalized modular switching services. Note that GMPLS encompasses MPLS due to PSC, however, with a few twists. While in MPLS, a bandwidth request can be any quantity; in GMPLS the bandwidth request corresponds to one of the well-defined modular values such as asynchronous mapping of T1 or Packet over SONET at an OC-3 rate. That is, if PSC is used in a GMPLS environment, then the requirement must be mapped to one of the allowable packet switching types along with the associated bandwidth. In addition, due to the bidirectional nature of the services offered by GMPLS, LSPs must be set up in each direction for a path to be operational.

It is important to note that GMPLS is not necessarily only for IP traffic, since it is meant for generalized modular switching. Thus, a GMPLS tunnel does not need to start or end on an IP router; rather, it starts and ends on *similar* GMS nodes. To allow for different GMS types, a generalized label has been defined for GMPLS; this is discussed later.

GMPLS allows LSP set-up for protection for any link-related failure. Thus, a primary and secondary (backup) LSP can be set up if and when needed. This set-up indicates whether it is primary or secondary. Furthermore, protection types such as dedicated $1 + 1$, dedicated 1:1, and shared 1:N can be announced. Furthermore, GMPLS allows control and data channels to be separate.

22.4.1 GMPLS LABELS

GMPLS defines several types of labels. The most common one is known as a *generalized label request*. A generalized label request includes three pieces of information: LSP encoding type, switching type, and generalized payload identifer (G-PID) (see Figure 22.12). LSP encoding types are values such as Packet, Ethernet, SDH, and Lambda (see Table 22.2). Switching type refers to the generalized modular switching discussed earlier with additional details. For example, within PSC, variations in implementation such as asynchronous mapping, bit synchronous mapping, and byte synchronous mapping are possible; thus, they are identified separately while G-PID specifies the payload identifier (see Table 22.3 and Table 22.4). Note that since GMPLS is for generalized modular switching, bandwidth encoding for well-defined data rates such as DS0, DS1, and Ethernet have also been defined.

In addition to the standardized label request discussed so far, GMPLS allows a generalized label, port, and wavelength label by using a 32-bit field without specifying any details. For wavelength switching, there are three fields defined, each of 32 bits; they are identified wavelength ID, start of a label, and end of a label (Figure 22.13).

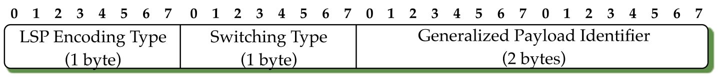


FIGURE 22.12

Generalized label request.

Table 22.2 GMPLS LSP encoding type.

Value	Type
1	Packet
2	Ethernet
3	ANSI PDH
5	SONET ANSI T1.105/ SDH ITU-T G.707
7	Digital Wrapper
8	Lambda (photonic)
9	Fiber
11	FiberChannel
12	ITU-T G.709 Optical Data Unit (ODU*k*)
13	ITU-T G.709 Optical Channel

Table 22.3 GMPLS switching types.

Value	Type
1	Packet-Switch Capable-1 (PSC-1)
2	Packet-Switch Capable-2 (PSC-2)
3	Packet-Switch Capable-3 (PSC-3)
4	Packet-Switch Capable-4 (PSC-4)
51	Layer-2 Switch Capable (L2SC)
100	Time-Division-Multiplex Capable (TDM)
150	Lambda-Switch Capable (LSC)
200	Fiber-Switch Capable (FSC)

22.4.2 LABEL STACKING AND HIERARCHICAL LSPS: MPLS/GMPLS

It is possible to coordinate between MPLS and GMPLS through label stacking to create hierarchical LSPs. This can be useful when operating over multiple technologies such as in multilayer networking (refer to Section 24.6). In this section, we present a simple example to illustrate this nested label concept over multiple technologies.

In Figure 22.14 we consider a label from an MPLS router marked as node-1 to an MPLS router marked as node-6. This LSP is an MPLS packet level LSP. This LSP, however, is connected on the GMPLS LSP between node-2 and node-5, which are TDM switches. Not only that, the TDM level GMPLS-based LSP is carried further over two optical switches, marked as node-3 and node-4. Thus, the original LSP between node-1 and node-6 is nested in GMPLS LSP between node-2 and node-5, which is turn is nested in another GMPLS LSP between node-3 and node-4.

Table 22.4 Examples of GMPLS Generalized Protocol identifier (G-PID).

Value	Type	Technology
0	Unknown	All
6	Asynchronous mapping of DS3/T3	SDH
7	Asynchronous mapping of E3	SDH
8	Bit synchronous mapping of E3	SDH
9	Byte synchronous mapping of E3	SDH
13	Asynchronous mapping of E1	SDH
14	Byte synchronous mapping of E1	SDH
15	Byte synchronous mapping of 31 * DS0	SDH
16	Asynchronous mapping of DS1/T1	SDH
17	Bit synchronous mapping of DS1/T1	SDH
18	Byte synchronous mapping of DS1/T1	SDH
19	VC-11 in VC-12	SDH
22	DS1 SF Asynchronous	SONET
23	DS1 ESF Asynchronous	SONET
24	DS3 M23 Asynchronous	SONET
25	DS3 C-Bit Parity Asynchronous	SONET
28	POS – No Scrambling, 16 bit CRC	SDH
29	POS – No Scrambling, 32 bit CRC	SDH
30	POS – Scrambling, 16 bit CRC	SDH
31	POS – Scrambling, 32 bit CRC	SDH
32	ATM mapping	SDH
33	Ethernet	SDH, Lambda, Fiber
34	SONET/SDH	Lambda, Fiber
36	Digital Wrapper	Lambda, Fiber
37	Lambda	Fiber
38	ANSI PDH	SDH
43	FiberChannel-3 (Services)	FiberChannel
44	HDLC	SDH
45	Ethernet V2/DIX (only)	SDH, Lambda, Fiber
46	Ethernet 802.3 (only)	SDH, Lambda, Fiber
47	G.709 ODUj	G.709 ODUk (with $k > j$)
48	G.709 OTUk(v)	G.709 OCh (ODUk mapped into OTUk(v))
53	IP/PPP (GFP)	G.709 ODUk (and SDH)
54	Ethernet MAC (framed GFP)	G.709 ODUk (and SDH)
55	Ethernet PHY (transparent GFP)	G.709 ODUk (and SDH)
58	Fiber Channel	G.709 ODUk, Lambda, Fiber

22.4.3 RSVP-TE FOR GMPLS

For GMPLS LSP set-up, several set-up and confirmation message types are used in RSVP-TE (see Table 22.6). Note that RSVP uses unreliable delivery since it is embedded in a UDP packet. However,

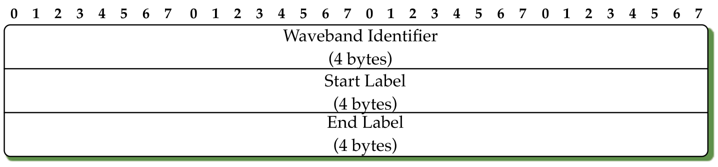


FIGURE 22.13

GMPLS label for wavelength switching.

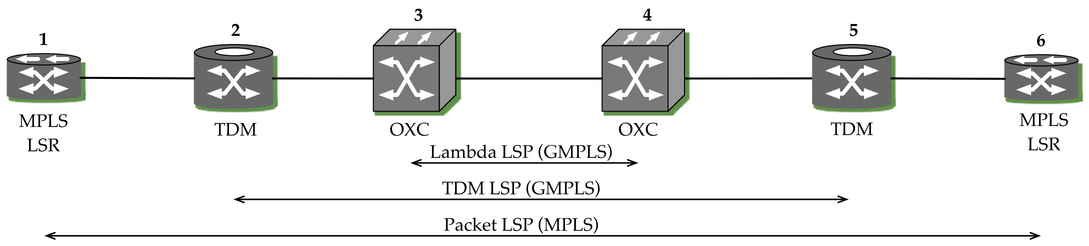


FIGURE 22.14

Label stacking and hierarchical LSPs: MPLS/GMPLS.

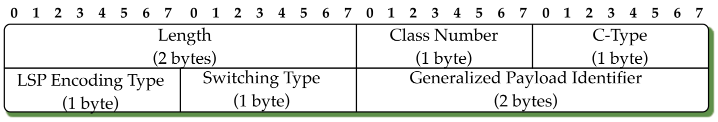


FIGURE 22.15

RSVP-TE generalized label request object for GMPLS.

GMPLS requires reliable delivery of set-up messages; to accomplish this, a simple reliable delivery mechanism is used where the sender retransmits the set-up message until it receives an acknowledgment message from the neighbor. Message retransmission can be frequent, such as every 10 millisec; however, an exponential decay mechanism can be used to reduce frequency if successive tries do not result in a response, along with a maximum timeout value to indicate failure to establish a tunnel.

To carry a GMPLS label, RSVP-TE creates an object by including its own 32-bit header to identify the length of the message, class number, and C-Type. For the generalized label request shown in Figure 22.12, the corresponding RSVP-TE object is shown in Figure 22.15. Similarly, the RSVP-TE header is included for other GMPLS labels. Note that class number and class type depend on the RSVP object (see Table 22.5).

Recall that label set-up is bidirectional in GMPLS. To accomplish this in RSVP-TE, an Upstream-Label is included in the Path message. The class number for the Upstream-Label is 35 and the C-Type used is as in the RSVP-LABEL (see Table 22.5). The data rate of particular GMPLS connections is also communicated through an RSVP-TE Path message using SENDER-TSPEC. Since the data rates

Table 22.5 RSVP object examples for GMPLS, in addition to Table 22.1.

Object Name	Used in	Class Number	Examples: Class-Type with value in () (source RFC listed as [] from bibliography)
RSVP-LABEL	Path	16	Generalized Label (2) [99]; Wavelength switching (3) [99]
LABEL-REQUEST	Path	19	Generalized label request (4) [99]
SENDER-TSPEC	Path	12	Integrated Services (1) [121]; G.709 (5) [642]
Upstream-Label	Path	35	Same as in RSVP-LABEL
RECOVERY-LABEL	Path	34	Same as in RSVP-LABEL
PROTECTION	Path	37	Type 1 Protection (1) [99]
NOTIFY-REQUEST	Path, Resv	195	IPv4 request (1) [99]

Table 22.6 GMPLS message type and RSVP-TE protocol messages.

GMPLS Message type for setup	RSVP-TE Protocol message
LSP Setup	Path
LSP Accept	Resv
LSP Confirm	ResvConfirm
LSP Upstream Error	PathErr
LSP Downstream Error	ResvErr
LSP Downstream release	PathTear
LSP Upstream release	PathErr
LSP Notify	Notify

for GMPLS generalized modular switching are well-defined, the actual rate does not need to be coded; instead, a mapping value is provided for well-known data rates; this is listed in Table 22.7.

22.4.4 ROUTING PROTOCOLS IN GMPLS

The role and use of a routing protocol in GMPLS are primarily to enable traffic engineering of GMPLS-based connection-oriented networks. Two important points to note are as follows:

- GMPLS-based networks require a link state-based framework for communicating the status of links.
- The actual path computation algorithm need not be within the scope of these routing protocols.

You may recall that we have made the above points about link state routing protocols earlier in Chapter 3. Since GMPLS can be used for different technologies, the routing protocols need to be somewhat generic so that it can be used in any of these technologies. At the same time, there are existing routing protocol frameworks that can be applicable, for example, OSPF and IS–IS. To satisfy the traffic engineering requirements, extensions to OSPF/IS–IS have been developed. However, first, we need to note that a "link" in a GMPLS network may not necessarily be a physical link. Depending on the technology, it may ride over another physical technology; this will be discussed later in the context of multilayer networking. Second, an LSP that has been set up can serve as a point-to-point link for other nodes. Third, a protection notion about a link may be helpful to communicate what can

Table 22.7 Data rate for GMPLS.

Signal Type	Bit-rate (in Mbps)	32-bit Encoding value (in hex)
DS0	0.064	0x45FA0000
DS1	1.544	0x483C7A00
E1	2.048	0x487A0000
DS2	6.312	0x4940A080
E2	8.448	0x4980E800
Ethernet	10.000	0x49989680
E3	34.368	0x4A831A80
DS3	44.736	0x4AAAA780
STS-1	51.840	0x4AC5C100
Fast Ethernet	100.000	0x4B3EBC20
E4	139.264	0x4B84D000
FC-0 133M		0x4B7DAD68
OC-3/STM-1	155.520	0x4B9450C0
FC-0 266M		0x4BFDAD68
FC-0 531M		0x4C7D3356
OC-12/STM-4	622.080	0x4C9450C0
GigE	1,000.000	0x4CEE6B28
FC-0 1062M		0x4CFD3356
OC-48/STM-16	2, 488.320	0x4D9450C0
OC-192/STM-64	9, 953.280	0x4E9450C0
10GigE-LAN	10,000.000	0x4E9502F9
OC-768/STM-256	39,813.120	0x4F9450C0

be used by the routing path computation module. Thus, in general, it is safer to refer to a link in the context of GMPLS routing as a *TE link*. Since in GMPLS, control and data planes are completely separate, appropriate identifiers must be used so that end of a link can be identified, which is discussed later in Section 22.4.5.

GMPLS extension of traffic engineering of a routing protocol relies on MPLS traffic engineering extensions. For example, sub-TLVs discussed earlier for MPLS such as maximum link bandwidth are still applicable. In addition, the following new sub-TLV types have been defined as follows:

- Link Local/Remote Identifiers to provide support for unnumbered links
- Support for link protection to announce the protection capability available for a link; this may be useful for path computation. Typical values are unprotected, shared, dedicated 1 + 1, and dedicated 1:1
- Interface Switching Capability Descriptor to identify generalized modular switching capabilities (see Table 22.3)
- Shared Risk Link Group (SRLG) identification: This issue arises from multilayer networking since multiple TE links at a particular layer may be using the same "link" at a lower layer, such as at a fiber level. Alternately, a TE link may belong to multiple SRLGs. This information may be used

for path computation as well. We will describe SRLGs in detail later in the context of multilayer networking.

We now briefly highlight some of the extensions to OSPF and IS–IS.

OSPF and IS–IS extension

To allow for TE link information exchange, OSPF again uses the *opaque* LSA—this is the same as done for MPLS. That is, for TE usage, an Opaque-type value of 10 is specifically assigned; this LSA is known as a TE LSA. From an information encoding point of view, a TE LSA uses one top level TLV along with nested sub-TLVs, when needed, for an unnumbered link identifier, link protection type, SRLG, and so on. The sub-TLVs then include the extension information described above. Note that this is a continually evolving standard; for an up-to-date list of sub-TLVs for TE, refer to [396].

In IS–IS, all routing information encoding is always done using TLVs. Thus, for GMPLS, new TE TLVs have been defined to capture information such as protection information, SRLGs, and so on, which we have listed earlier in this section (also, see Table 6.2 in Chapter 6). An up-to-date list of defined TLVs can be found at [395].

22.4.5 CONTROL AND DATA PATH SEPARATION, AND LINK MANAGEMENT PROTOCOL

Control and data path separation in GMPLS is quite different from IP networks. Recall that, in IP networks, there is no separation of control and data traffic carried in terms of a physical channel or partitioned bandwidth. An IP link carries both control and data traffic—the separation of control traffic is identified either at the IP protocol type field level (e.g., OSPF packet) or port level (e.g., RSVP packet). In an IP/MPLS environment, although there is separation of control traffic and MPLS packet forwarding, they both use the same logical link between two routers.

Now let us consider a different example: PSTN with SS7 for signaling. In PSTN, data traffic means the voice calls, which are carried on TDM trunks, while control traffic, for example ISUP messages for call set up, is sent over the SS7 network—that is on a completely separate network. Thus, ISUP call set up messages on the SS7 network traverse on a completely different path or channel than the voice circuits (for example, refer to Figure 25.11).

In GMPLS, control and data path separation is similar to PSTN/SS7 setting. In GMPLS, they are separated through use of separate channels; the difference between the GMPLS approach and that of PSTN is that it does not define a completely separate network architecture like SS7 for PSTN. Instead, a separate channel may be dedicated for the delivery of control traffic from data traffic (Figure 22.16). More importantly, the channel that carries control traffic need not be on the same physical path as the data traffic. Recall that in PSTN, the voice circuit to be used is identified in an SS7 message through trunk ID (that is, TCIC code, refer to Section 25.6); similarly, in GMPLS the data link must be explicitly identified so that this can be communicated through exchange of control traffic.

Separation of control and data paths, however, necessitates the need for a link management control so that the two ends of a data link can communicate link-level information, which is completely decoupled from control information for traffic engineering. To accomplish this, the link management protocol (LMP) has been defined for use in a GMPLS networking environment [482].

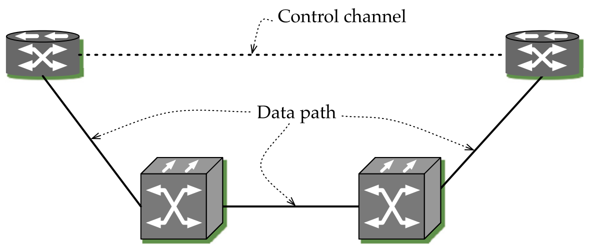


FIGURE 22.16

Control and data path separation in GMPLS.

LMP provides two core functions: (1) *control channel management*, and (2) *link property correlation*. The role of control channel management is to establish and maintain bidirectional control channels between adjacent nodes. The establishment step is accomplished using a configuration message; after establishment, a keep-alive message is generated frequently to ensure that the control channel is up. The link property correlation function of LMP allows aggregation of multiple, parallel data links into a TE link along with synchronization of the properties of the TE link. A multiple link situation arises, for example, when there are multiple data links between ports of two adjacent nodes—they can be combined into a TE link through the link property correlation function.

From Figure 22.16, we can see that it is quite possible in GMPLS to have control channels on a completely different physical facility from the data channels. To check that both channels/paths are operating normally, additional functions have been defined. There are two additional LMP functions for this purpose: link connectivity verification and fault management. Link connectivity verification is accomplished by sending a test message over the data channel; in turn, a test status message is sent back over the control channel to complete such verification. That is, the verification is performed only at the end nodes of a TE link.

When the control channel uses a different path from the data channel, additional issues come up. Specifically, two scenarios are possible: 1) the channel that carries the control messages has a failure, while the data path is intact, 2) the control path is intact; however, a part on the data path is lost due to a failure. In the first case, the data path will continue to carry data traffic—an issue is that if an established data path is to be teared down, this cannot be communicated due to the control channel being down. The second scenario is more problematic. Here, the control ends think that the data path is working fine; however, it is not any more. To address this problem, an LMP fault management procedure has been additionally defined. Briefly, the LMP fault management procedure is based on a ChannelStatus message exchange that uses the following messages: ChannelStatus, ChannelStatusAck, ChannelStatusRequest, and ChannelStatusResponse. The ChannelStatus message is sent unsolicited and is used to notify an LMP neighbor about the status of one or more data channels of a TE link. The ChannelStatusAck message is used to acknowledge receipt of the ChannelStatus message. The ChannelStatusRequest message is used to query an LMP neighbor about the status of one or more data channels of a TE Link. The ChannelStatusResponse message is used to acknowledge receipt of the ChannelStatusRequest message and to indicate the states of the queried data links.

Finally, it is worth noting that, to use LMP, the end nodes are first established with a control channel; then, the address of the ends from the control channel is used to establish an LMP communication.

22.5 MPLS VIRTUAL PRIVATE NETWORKS

While MPLS was originally intended for controlled IP traffic engineering, it has been found to be useful in virtual private networking (VPN) as well. Suppose that a corporate customer has offices in different physical locations distributed geographically; such customers would like to lease a seamless "network" service through a provider that has a geographic presence in these areas and would require this provider to carry their corporate intranet IP-based traffic. Since MPLS is a label-based concept, a VPN backbone provider can assign (provision) a unique label along with an LSP for traffic between any two sites, including any bandwidth guarantee using RSVP-TE label set-up. To do that, the VPN backbone provider would need to have label-edge routers, commonly referred to as *provider edge (PE) routers*, at each site to which customers can be connected through their *customer edge (CE) routers*. To the customer, it appears to be a point-to-point link with a guaranteed data rate. For the VPN backbone provider, the MPLS-based approach is appealing since it can combine different customers' traffic on the *same* network by doing software-based provisioning of LSPs for different customers. Such VPN service is also known as provider-provisioned VPN (PPVPN) service; generic requirements for PPVPN can be found in [610] and terminologies are described in [34].

While conceptually this approach is appealing, there is a key issue: address conflict. What does address conflict mean? Private IP addressing as defined in RFC1918 [702] has become a popular mechanism for addressing hosts within almost any organization or corporate environment. Because of the prevalence of private IP addressing, two organizations may both choose to number hosts in the address block, say 10.0.0.0/8; certainly, within an organization the address allocation from this address block is unique for different hosts and subnets. If both these organizations are to be supported by a VPN provider, then address conflicts will exist since both are using the same address block. Thus, an identifier is needed so that they can be distinguished and the traffic is properly routed through the VPN provider's network.

A possible approach for identifier tracking is that every router in the provider's network maintains routing information for all customers' networks. While this is a possible approach, it raises a second issue: scalability; this is because there would be limitations based on how many sites a router can support in terms of amount of routing information.

22.5.1 BGP/MPLS IP VPN

A major advantage of BGP (see Chapter 9) is the idea of federation among different autonomous systems. When BGP was initially designed, it was meant for use with IPv4 addressing with federation achieved through autonomous systems that are given unique autonomous system numbers. With large scale deployment of VPN services, many customers use the same private IPv4 address block (such as 10.x.y.z). Thus, if two different customers with the same private IPv4 address block want an IP VPN service from a network VPN provider, it becomes difficult to distinguish. To circumvent this situation, a mechanism was necessary to distinguish different VPN customers and to announce reachability information of their address blocks. Furthermore, federation is also helpful for distinguishing VPN customers as well as if two network providers want to cooperate on IP VPN services.

Instead of designing an entirely new protocol, it has been possible to accomplish what is needed for IP VPN, by using BGP through its multiprotocol extension MP-BGP, for MPLS; this is known as BGP/MPLS. Refer to Section 9.10.5 for a detailed discussion on BGP/MPLS. For here, it suffices to understand the following two basic features for the examples on how BGP/MPLS is used for IP:

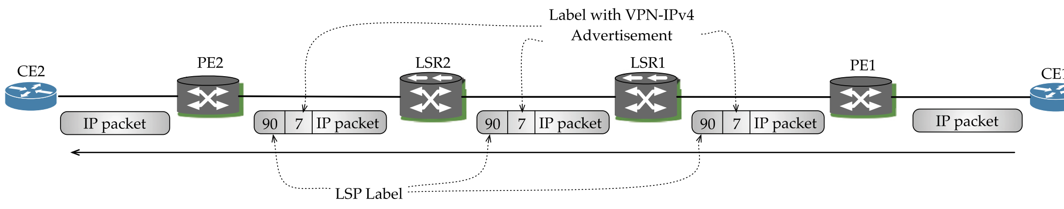


FIGURE 22.17

Prepending of two labels in BGP/MPLS packet forwarding.

- The network layer reachability information (NLRI) field in BGP/MPLS constitutes of two parts: a 64-bit route distinguisher (to differentiate different customers) and the traditional 32-bit IPv4 prefix (usually from the private IPv4 address space).
- The notion of a virtual router is introduced in MPLS VPN. Specifically, for different customers, Virtual Routing and Forwarding tables (VRFs) are used where each customer's routes are in a different table.

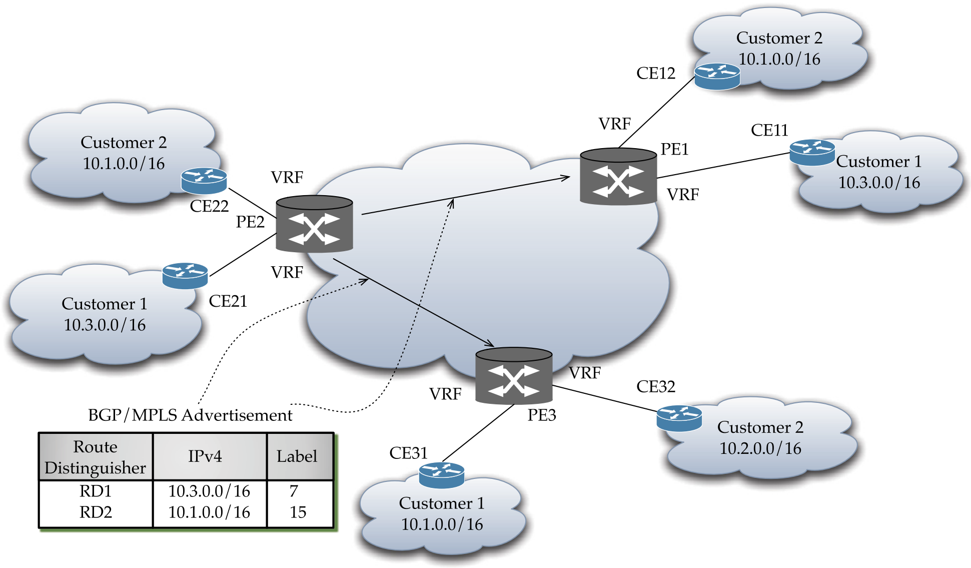
Example 22.2. Illustration of labeling in BGP/MPLS IP VPN between two customer edge (CE) routers of a customer.

In Figure 22.17, we show an IP packet being sent from CE router CE1 to CE router CE2 at another location; this packet is intended for a network/subnet served behind CE2 (not shown in the figure). Note that the CE1 is connected to the provider edge router, PE1; similarly, CE2 is connected to PE2. In between PE1 and PE2, the provider has LSRs; we show two in Figure 22.17—we assume that between PE1 and PE2, a label switched path (LSP) is already established for carrying this customer's traffic.

First, CE1 forwards the packet to its default gateway, which happens to be PE1. On receiving this packet from CE1, PE1 consults its VRF for route lookup. For the VRF to know that there is a route to the networks supported behind CE2, PE1 would need to have already received an advertisement from PE2 about the networks supported by CE2. Suppose that the VPN-IPv4 address for the networks supported behind CE2 is advertised as MPLS label 7. The VRF at PE1 would then have an entry for MPLS label 7. Thus, the IP packet will first be prepended with the MPLS label 7. Second, PE1 would lookup the LSP set-up between PE1 and PE2 that happens to have label 90; thus, the packet will now be prepended by label 90 for using this LSP for packet forwarding. ●

We can see from the above example that every IP packet will have *two* labels prepended: the inner label is for the route distinguisher, and the outer label is for packet forwarding through the MPLS VPN provider's network. It is important to note that the route distinguisher is advertised through a BGP advertisement; it is not included in the actual packet forwarding; instead an MPLS label is used. This MPLS label (inner label) is the information advertised with the route distinguisher.

While injection of two labels on an IP packet certainly incurs additional overhead, it provides the ability to clearly separate the route distinguishing part and the LSP part; furthermore, because of this advantage, a provider has the flexibility to use a single LSP between two sites to carry traffic between different customers. Alternately, for the same customer and for traffic between two sites, the VPN provider's network may choose to set up multiple LSPs through its network for its traffic engineering



Route Distinguisher	IPv4	Label
RD1	10.3.0.0/16	7
RD2	10.1.0.0/16	15

FIGURE 22.18

BGP/MPLS example with multiple customers.

requirements while satisfying customer's goals. Note that use of any multiple LSPs to split traffic between two sites is transparent to the customers. VPN traffic engineering is discussed in Section 23.2. Moreover, there is a scalability advantage since only one label for each pair of ingress/egress points through the backbone can be done, instead of having different labels for different customers for the same pair of ingress/egress points.

Now consider again Figure 22.17. If the customer edge routers are also MPLS LERs, then the LSP can possibly be set up from CE1 to CE2 where the receiving customer edge router is responsible for the outer label processing. However, to reduce the load on the customer edge routers, a function known as *Penultimate Hop Popping* is performed at the provider edge router ("penultimate hop router") in which the outermost label of an MPLS packet is removed before the packet is passed to the customer edge router. To activate this function, the edge router must indicate to the penultimate hop router to use implicit null label 3.

Example 22.3. Illustration of multiple customers in a BGP/MPLS IP VPN provider's network.

Consider the topology shown in Figure 22.18 where two customers are identified as Customer-1 and Customer-2. Each uses private address space 10.0.0.0/8 that is assigned to each site as indicated in Figure 22.18; their CE routers in a particular location connect through the same PE routers in the MPLS VPN provider's network.

At each PE router, two VRFs would be maintained, one for each customer. When initialized, the network will use an M-BGP protocol to exchange route information, rather than VPN-IPv4 route information, as shown in Figure 22.18 from PE2 to other PE routers—note that when advertising this route, the BGP announcement will include the next hop as that of this PE. Similarly, the other PE routers will advertise route information (not shown in the figure). ●

It may be noted that the BGP extended community attribute can also be used with distribution of the VPN routing information. For example, this can be helpful to customers in controlling the distribution of routes.

Because of the policy capabilities of BGP, two customers can choose to exchange traffic through MPLS VPN if they have any business relations. In the above example, Customer-1 might want to exchange data with Customer-2; once the policy is finalized that determines which routes are allowed to which customer, the policies can be set up at the PE routers. It should be clear by now that the PE routers are acting as BGP speakers.

In general, it may be noted that BGP/MPLS is not designed for all types and sizes of customers or VPN services. For example, a customer that has just two sites in two different locations might choose to tunnel any internal traffic through the public Internet by encrypting the data such as IPSec, especially if it does not require any guaranteed quality-of-service; such a customer does not necessarily need to use the BGP/MPLS approach. This means that the BGP/MPLS approach is meant more for moderate to large customers that have multiple sites and a required guaranteed quality-of-service and an interest in sharing business traffic with other similar customers; for them, a BGP/MPLS provides a good alternative in an IP framework while bypassing the public Internet. It is worth pointing out that BGP/MPLS is meant for a single VPN provider to operate its network in order to carry traffic for different customers; it does not solve the need of a customer that wants a VPN tunnel between two locations that are not served by a single VPN provider's network—in such cases, often a VPN provider becomes the contract ("official") provider who, in turn, makes a contractual arrangement with another provider to carry the traffic for the customer to locations where the official provider has no connectivity.

What about traffic engineering from the point of view of a VPN provider? This is discussed in Section 23.2.

22.5.2 LAYER 2 VPN

In the previous section, we discussed the layer 3 VPN approach using the BGP/MPLS IP VPN approach. Many customers with offices located in geographically disparate sites are also interested in layer 2 VPN services; this means that they have a layer 2, e.g., Ethernet, subnet that they would like to tunnel across a wide-area network so that all sites for a particular customer look as if they are on the same LAN, e.g., by using standard Ethernet frames. Such a service is known as a *Virtual Private LAN Service (VPLS)*, *transparent LAN services (TLS)*, or "emulated" Ethernet service [537].

A key advantage of MPLS is its label-based forwarding mechanism to carry any type of packets. Thus, input to an MPLS router need not be only IP packets; it can be any other protocol packets as long as the ingress MPLS router has the functionality to accept them. When such a packet arrives, MPLS routers add the MPLS label header and forward it to a destination at another site. For example, such a received packet at an MPLS ingress router can be an Ethernet frame [537]. Thus, this frame will be carried as an MPLS packet through the wide-area network and is delivered to an egress MPLS router that will strip off the MPLS header and deliver the Ethernet frame on the LAN at the other site.

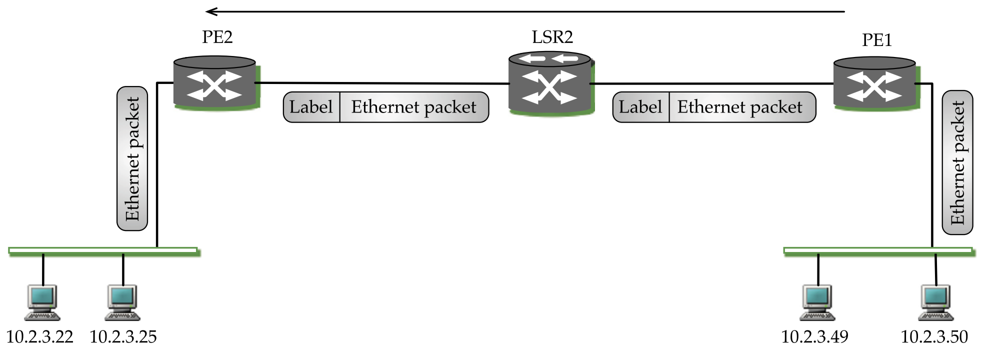


FIGURE 22.19

Layer 2 VPN over MPLS.

Example 22.4. A layer 2 over MPLS example.

Consider two geographically disparate sites that a customer needs to be on subnet 10.2.3.0/24. Now consider an IP packet generated at host 10.2.3.49 destined for 10.2.4.22 where the second host is at a different site.

As shown in Figure 22.19, the IP packet generated at host 10.2.3.49 will be encapsulated as an Ethernet frame that will arrive at the layer 2 CE device that forwards it to a provider ingress PE router. The Ethernet frame is then encapsulated in an MPLS packet using an LSP to reach the egress PE router where the MPLS label is stripped, and the Ethernet frame is sent to the customer edge device for forwarding to the LAN for delivery to 10.2.3.22. As far as the two hosts are concerned, the Ethernet frame stayed on the same LAN. ●

Note that for such services, for each Ethernet frame that arrives at the ingress MPLS router, two labels are added. The first is added at MPLS level to identify the input source, and the second is a label that serves as the LSP tunnel from the ingress to the egress router. Once the packet arrives at the egress router, it first removes the label associated with the LSP tunnel, and then for the second label, a lookup table is checked to find the destination interface to the customer edge device.

22.6 MULTICAST VPN WITH MPLS

Multicast VPN services can be provided either at layer 2 or layer 3. To provide multicast IP VPN services at layer 3, BGP/MPLS supports multicasting; see RFC 6513 [711]. The actual reservation and LSPs associated with multicasting are set up using P2MP LSPs.

The layer 2 VPN also creates an interesting scenario in that more than two sites for the same customer might want to be connected in the same emulated LAN. Consider again Figure 22.19 and imagine a third site for the same customer that is also on 10.2.0.0/16 subnet. In this case, copies of the packet would need to be sent to each destination requiring setup of multiple LSPs. While this approach is doable, it requires many more LSPs to be set up as well as extra packet transmissions, resulting in a higher bandwidth use than necessary. This is also where P2MP LSP and MP2MP LSP can be used.

22.7 SUMMARY

In this chapter, we have presented two enabling environments for traffic engineering: MPLS and GMPLS. While MPLS is exclusively used for packet-based networks, GMPLS can be used for both packet as well as circuit-based networks on modular well-defined data rate levels. Furthermore, GMPLS addresses bidirectional requirements in circuit-mode connections whereas MPLS is unidirectional. For traffic engineering information exchange in an MPLS/GMPLS environment, OSPF and IS–IS protocols have been extended to contain traffic engineering information. It is important to note that the TCP/IP stack is used for exchanging control traffic information in GMPLS. This requires GMPLS nodes to have IP addresses; this aspect is also significant when you consider nested label stacking between MPLS and GMPLS. This is shown earlier in Figure 18.13. Such use of the TCP/IP stack allows label exchanges to be performed seamlessly.

In subsequent chapters, we will discuss how these enabling environments help traffic engineering in a variety of networks.

FURTHER LOOKUP

MPLS architecture is described in RFC 3031 [715] while label stack encoding is discussed in RFC 3032 [714]. LDP specification is described in RFC 3036 [33]. For extensive coverage of MPLS, see books such as [211], [259], [321], [330]. Originally, MPLS was defined for use within a provider's network ("intra-AS"). There have been works on extending MPLS for inter-AS; see RFC 4216 [890], RFC 4726 [262].

There have been two primary candidates for traffic engineering specifications in MPLS, RSVP-TE and *Constraint-Routing Label Distribution Protocol* (CR-LDP, in short). The MPLS working group within IETF recommended discontinuation of any new effort on CR-LDP in 2003 [35]. Extensions of RSVP to RSVP-TE for MPLS have been described in RFC3209 [61] and RFC4090 [639], and updates in RFC5420 [261]; applicability of RSVP-TE to LSP is described in RFC3210 [62]. We refer readers interested in CR-LDP to RFC3212 [418], RFC3213 [53], and RFC3214 [55].

Point-to-Multipoint and Multipoint-to-Multipoint Label Switched Paths are discussed in RFC 6388 [863] and the related extension to RSVP in [15].

RFC 2764 [313] presents a framework for IP-based VPN. The concept of BGP/MPLS IP VPN was presented in RFC 2547 [712], and has been updated in RFC 4364 [713], while multicast for IP BGP/MPLS IP VPN is discussed in [711]; for additional work, see RFC 4684 [536], RFC 4797 [698]. Example 22.3 is adapted from [745]. Layer 2 VPN encapsulation of Ethernet over MPLS has recently been described in RFC 4448 [537]; see also, RFC 4761 [457], RFC 4762 [485].

GMPLS was originally described in RFC 3471 [97] and has been updated in RFC 4201 [453], and RFC 4328 [642]. For routing extensions to GMPLS, see RFC 4202 [456], RFC 4203 [455], and RFC 4205 [454]. GMPLS extension to G.709 is described in RFC 4328 [642]. LMP is described in RFC 4204 [482]. Details on GMPLS can be found in [260].

EXERCISES

22.1 Review questions:

- **(a)** What is MPLS?
- **(b)** What is OSPF-TE?
- **(c)** Which layer does MPLS operate?
- **(d)** What is penultimate hop popping?

22.2 What are the key differences between MPLS and GMPLS?

22.3 What is the FAST-REROUTE object used for?

22.4 Sample messages for RSVP-TE are available at [869]. Investigate how different values are specified in RSVP-TE messages.

22.5 Explore the current activities of the MPLS-related working groups at IETF.

22.6 Discuss advantages and disadvantages of MPLS-based layer 2 VPN against MPLS-based layer 3 VPN.

22.7 Two routers have 4 equal cost links. How many LDP sessions will be established?