

Chapter 2

Quality of Service Implementation

2.1 QoS Definition

In order to transfer the various applications and services that exist in the triple-play network, it is necessary to provide resources and different levels of performance so that the end-users would get a satisfactory level of service. Within a converged network, QoS is by far the most important implementation consideration. QoS is a networking term that specifies a guaranteed network data performance level. QoS guarantees are important if the network capacity is insufficient, especially for real-time streaming multimedia applications such as VoIP, online games, and IP-TV, since these often require a fixed bit rate and are delay sensitive. Accordingly, QoS could also be defined as a set of algorithms and forms which enable different priorities to be given to different applications, users, or data flows, or to guarantee a certain level of performance for data flow. In practical terms, QoS can be described as a mechanism that has to provide transmission of audio and video data with minimum delay and minimum packet loss rate. Without QoS configuration, transmission of IP voice or videoconferencing calls would be unreliable, inconsistent, and often unsatisfactory.

The performance of a packet-based network can be characterized by several parameters where the most important ones include the following [20]:

- Bandwidth—typically specified in kilo or mega bits per second (kbps or Mbps), is measured as the average number of bits per second that can travel successfully through the network;
- The average packet delay (latency)—End-to-end delay is the average time it takes for a network packet to traverse the network from one endpoint to the other;
- The delay variation (jitter)—Jitter is the variation in the end-to-end delay of sequential packets;
- The percentage of lost packets (packet loss ratio)—Packet loss is measured as the percent of transmitted packets that never reach the intended destination.

Table 2.1 Characteristics of different types of traffic without the implemented QoS mechanisms

Type of traffic	Performance without QoS mechanisms
Speech	Speech is difficult to understand
	Interruptions in conversation
	Delays conversation; the participants do not know when the other party is closed
	Calls are disconnected
Video	The image is blurry
	Speech and images are not synchronized
	Slow moving
Data	The data arrives at their destination with delay
	Users give up using the service due to the excessive response time
	The network is slow

For example, in case of IP voice and video communications systems, the bandwidth should be as large as possible while the end-to-end delay, jitter, and packet loss are minimized in order for systems to work properly. Lower end-to-end delay leads to a more satisfactory, natural feeling conferencing experience, while large delay values lead to unnatural conversations with long pauses between phrases or sentences. Moreover, large jitter values may cause packets to arrive in the wrong sequence causing jerky video or stuttering audio.

Table 2.1 shows the characteristics of different types of traffic without the implementation of QoS mechanisms. It is obvious that QoS support has become the essential parameter for the successful realization of multiservice networks.

2.2 QoS Parameters

QoS implementation should increase the performance of traffic that is transmitted in the network, but the fact that improving the characteristics of one type of traffic can cause the loss of performance transmission of another type of traffic must be taken into account as well. Successful implementation of QoS includes the basic setup of the parameters given above (bandwidth, average packet delay, jitter, and packet loss) in order to improve network performance and meet the demands of end-users. The parameters are defined as follows [20]:

A. *Bandwidth*

In the IP-based networks, bandwidth is defined as the capacity of the transmission medium. It is usually referred as ‘speed’ or network throughput and specified in kbps, Mbps, or Gbps. In case of QoS implementation, this is related to bandwidth allocation techniques because QoS techniques cannot affect or increase the actual bandwidth of a link. Specifically, QoS mechanisms cannot create additional

bandwidth, but they can contribute to a more efficient allocation of the existing bandwidth.

B. *Packet delay*

The delay of a network specifies how long it takes for a bit of data to travel across the network from one node or endpoint to another. Although users only care about the total delay of a network, designers and engineers need to conduct precise measurements. Thus, engineers usually report both the maximum and average delay, and the following components have been defined (Fig. 2.1):

- **Processing delay**—the time that elapses between the receipt of data on the input port and its forwarding to the destination on the output port. This delay depends on the CPU speed and CPU load in the system;
- **Serialization delay**—the time it takes for the packet or frame to be transmitted, i.e., time it takes to push the packet's bits onto the link. This delay depends on the speed of the communication link;
- **Queuing delay**—the time that the packet spends in routing queues, i.e., the delay between the point of entry of a packet in the transmit queue and the actual point of transmission of the message. Network interfaces transmit one frame at a time (typically one bit at a time). Hence, when two or more packets are forwarded to a network interface at the same time or close to the same time, one packet is transmitted while the others are put in a queue on the interface buffer to await their turn at the interface. Packets that are put into the queue must wait until they can be transmitted, adding milliseconds to the delay. Increases in queue delay can be measured and detected by monitoring traffic along a given network path. This delay depends on the load on the communication link;
- **Propagation delay**—This is the delay between the point of transmission of the last bit of the packet and the point of reception of the last bit of the packet at the other end. This delay depends on the physical characteristics of the communication link.

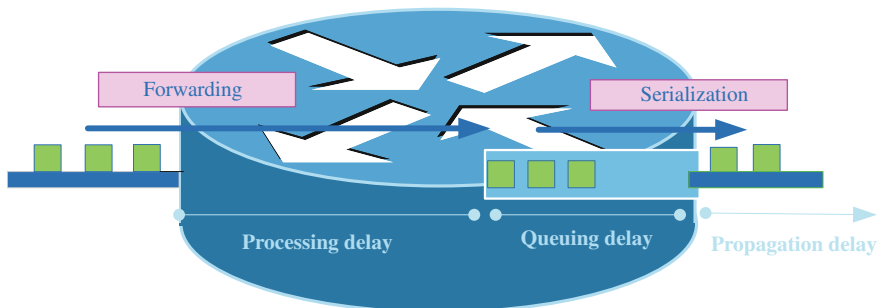


Fig. 2.1 Definition of delay in IP networks

The term end-to-end delay is used as a sum of all propagation, processing, serialization, and queuing delays in the path as shown in Fig. 2.2. Jitter defines the variation in the delay. In best-effort networks, propagation and serialization delays are fixed, while processing and queuing delays are unpredictable.

Most of the processing, queuing, and propagation delays are influenced by the following factors:

- Average length of the queue;
- Average length of packets in the queue;
- Link bandwidth.

There are several approaches to accelerate packet dispatching of delay-sensitive flows:

- Increase link capacity. Enough bandwidth causes queues to shrink, making sure that packets do not wait long before they can be transmitted. Additionally, more bandwidth reduces serialization time. On the other hand, this might be an unrealistic approach due to the costs associated with the upgrade;
- A more cost-effective approach is to enable a queuing mechanism that can give priority to delay-sensitive packets by forwarding them ahead of other packets.

Today, a variety of queuing mechanisms are used in IP networks, such as priority queuing, custom queuing, and weighted fair queuing. By minimizing delay, jitter is also reduced.

C. Packet delay variation (jitter)

In computer networking, packets from the source will reach the destination with different delays [21]. A packet's delay varies with its position in the queues of the routers along the path between the source and destination, and this position can vary unpredictably. This variation in delay is known as jitter and can seriously affect the

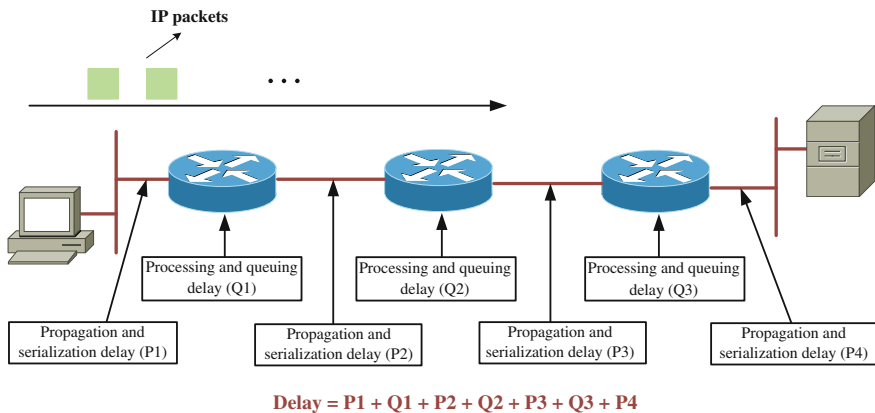


Fig. 2.2 Calculation of end-to-end delay

quality of streaming audio and/or video. It is important to emphasize that with delay minimization, jitter is also reduced (the delay is more predictable).

D. *Packet loss*

Packet loss in the network is defined as the percentage of lost packets on the path from source to destination due to the packet overload of the buffer, line errors, or even because of the implemented QoS mechanisms. Namely, the routers might fail to deliver some packets if their data is corrupted or if they arrive when their buffers are already full. Moreover, the receiving application may ask for this information to be retransmitted, possibly causing severe delays in the overall transmission. Packet loss is usually the result of congestion on an interface. Most applications that use TCP are slowed down due to TCP adjusting to the network’s resources. There are some other applications that do not use TCP and cannot handle congestion.

The following approaches can be taken to prevent drops of sensitive applications:

- Increased link capacity to ease or prevent congestion;
- Guarantee enough bandwidth and increase buffer space to accommodate bursts of fragile applications.

There are several mechanisms that can guarantee bandwidth and/or provide prioritized forwarding of drop sensitive applications such as priority and custom queuing. In order to decrease packet loss rate, it is necessary to prevent congestion (by dropping other packets before congestion occurs).

Table 2.2 summarizes the QoS requirements, i.e., expected values of the parameters of different multimedia applications.

In addition to the previous definitions and parameters, QoS implementation in the IP network also includes the following definitions:

Table 2.2 QoS requirements of different multimedia applications

Application	Bandwidth	Delay	Jitter	Packet loss rate
Voice transmission	Low-Medium	Low	Low	Low
Interactive voice transmission	Medium	Low	Low	Low
Streaming video traffic	Medium-High	High	High	Low
Video signalization	Low	Low	Medium	Medium
Speech signalization	Low	Low	Medium	Medium
Interactive and criminal record	Low-Medium	Low-Medium	Low-Medium	Medium-High
Non-interactive critical data	Variable, usually High	High	High	Medium
Interactive, non-critical	Variable, usually Medium	High	High	Medium

- Flow (or microflow) is a sequence of packets identified by source and destination IP addresses, protocol identifier (e.g., TCP and UDP), and source and destination port numbers;
- Traffic stream is a collection of flows with a common set of parameters (e.g., the same port number and the same source and destination network);
- Traffic profile specifies typical properties of a traffic stream (average rate and burstiness). Stream provisioning should be performed based on traffic profiles and the importance of traffic streams.

2.3 QoS Implementation

The key point for providing QoS in the NGN networks is the ability of these networks to differentiate traffic and to provide differentiated service levels based on the types of traffic. Namely, for the successful transmission of real-time applications, such as VoIP, the amount of available bandwidth and low end-to-end delay is necessary. At the same time, applications such as file transfer and/or e-mail transmissions are quite insensitive to bandwidth and delay issues.

The standard IP architecture is based on a ‘best-effort’ model where all network traffic is equally important and everyone receives service based on availability, without any guarantees [22]. Best-effort delivery describes a network service in which the network does not provide any guarantees that data will be delivered or that a user will be given a guaranteed QoS level or a certain priority. The implementation of QoS mechanisms is not supported, and users obtain best-effort service depending on the current traffic load. However, an important issue regarding the Internet, and consequently every network connected to it, is the support for multimedia applications. In networks with capacity problems that have to transmit real-time multimedia applications such as IP telephony and IP-TV, the best-effort mechanism is not able to provide enough resources. In this situation, mechanisms that are able to provide differentiated or guaranteed QoS to certain data flows must be employed.

The IETF (Internet Engineering Task Force) is responsible for standardization of the Internet and most of the protocols are used in the Internet. When faced with a challenge, vendors introduce their own solutions. However, the IETF is there to create standards that allow different vendor’s equipment to interoperate. One of the challenges in the past was to introduce QoS into the best-effort-driven Internet. Accordingly, two QoS architectures have emerged in the IETF:

- Integrated services architecture (IntServ), which provides end-to-end QoS on a per-flow basis and incorporates end-to-end signaling;
- Differentiated services architecture (DiffServ), which supports QoS for traffic aggregates through the implementation of different QoS for different traffic classes.

2.3.1 IntServ Model

IntServ model, or integrated services model defined in IETF RFC 1633 [23], represents an architecture that specifies the elements which will ensure QoS on networks and can be described as a flow-based system. In this model, every router in the system implements IntServ, and every application that requires some kind of guarantees has to make an individual reservation, Fig. 2.3. The integrated services model expects applications to signal their QoS requirements to the network. In order to accommodate such requirements, network devices will use this information to manage network resources. Network devices control network resources and provide QoS services by configuring the traffic handling mechanisms. Resource reservation can be applied to individual flows or aggregated flows.

Resource reservation mechanisms include the following functions:

- Provision of resource reservation signaling that notifies all devices along the communication path on the multimedia applications' QoS requirements;
- Delivery of QoS requirements to the admission control mechanism that decides whether there are available resources to meet the new QoS requirements;
- Notification of the application regarding the admission result.

Applications use the transport layer RSVP (Resource Reservation Protocol) to request and reserve resources through a network. RSVP does not transport application data but is similar to a control protocol. Moreover, this protocol assumes that resources are reserved for every flow requiring QoS at every router hop in the path between the receiver and transmitter using end-to-end signaling. RSVP requests resources for simplex flows (a traffic stream in only one direction from sender to

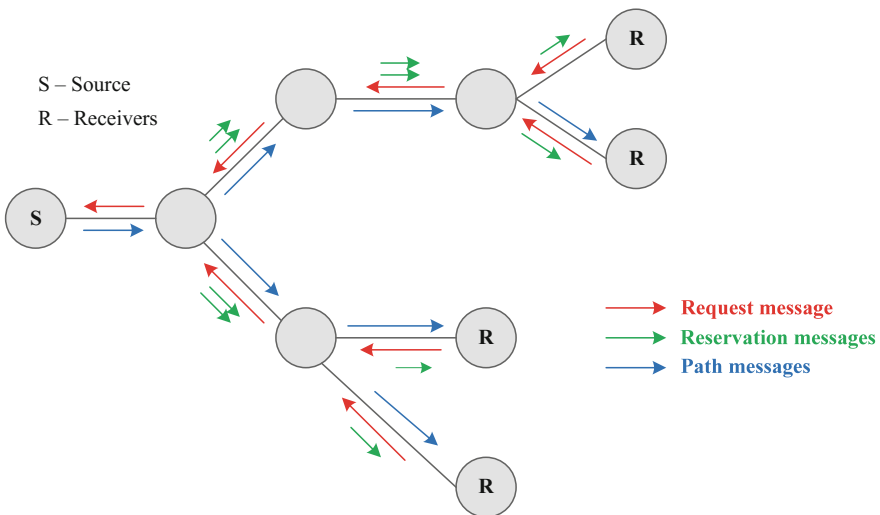


Fig. 2.3 RSVP functions

one or more receivers) and is receiver-oriented (the receiver of a data flow initiates and maintains the resource reservation for that flow).

RSVP operates over an IPv4 and provides receiver-initiated setup of resource reservations for data flows. RSVP’s main functionality is to exchange QoS requirement information between the source host, the destination host, and intermediate devices as shown Fig. 2.4.

The IntServ model itself describes the application of QoS in IP networks. In addition to RSVP, additional standards were developed to cover the exact protocols used for QoS implementation (Fig. 2.5):

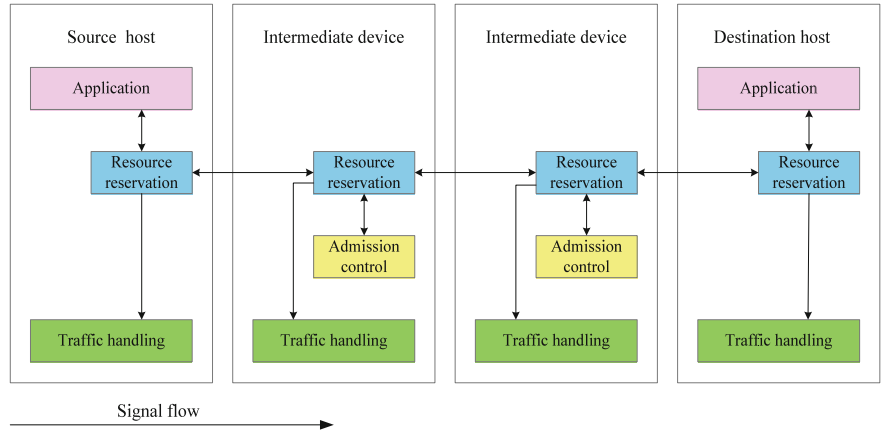


Fig. 2.4 Implementation of RSVP mechanism

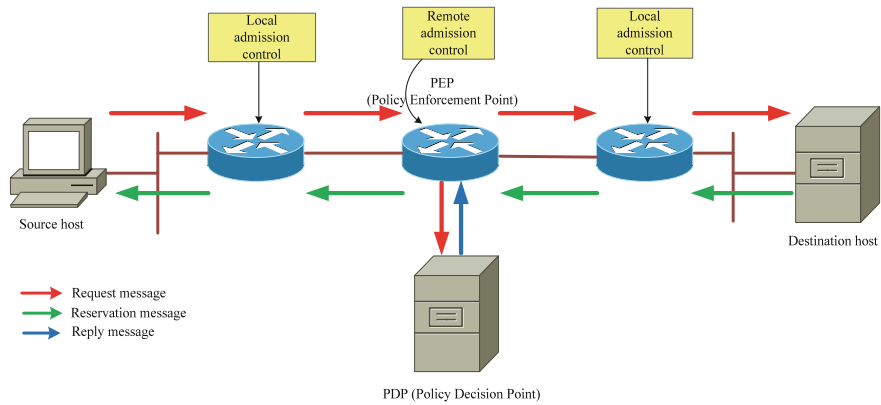


Fig. 2.5 Additional QoS mechanism

- Admission control is either implemented locally on the routers or offloaded to central servers;
- COPS (Common Open Policy Service) is another IETF standard that defines a protocol that can be used for policy exchange between network devices, PEPs (Policy Enforcement Points), and policy servers, PDPs (Policy Decision Points). An additional standard was added to integrate RSVP with COPS. The COPS (Common Open Policy Service) protocol is defined in RFC 2748 [24]. COPS usage for RSVP is defined in RFC 2749 [25].

To enhance the RSVP functionality, admission control mechanisms have to be implemented in order to determine whether the application (flow) can obtain the requested resources. Admission control mechanisms allow a network to reject (or downgrade) new RSVP sessions if one of the interfaces in the path has reached the limit (all reserved bandwidth is booked). These mechanisms will ensure that QoS parameters of the existing sessions will not be degraded and the new session will be provided with adequate QoS support.

In case the network does not have the resources necessary for the successful transfer of new flow, the mechanism may reject a request to establish a new session or to accept it and it will send a message to the end-user that the session is accepted but that the network may not be able to provide sufficient resources. Admission control mechanisms and resource reservation signaling mechanisms closely cooperate with each other and are implemented in the same device.

In today's networks, there are two approaches for the implementation of the admission control:

- Explicit admission control.
This approach is based on explicit resource reservation. Applications send the request that contains QoS parameters to join the network through the resource reservation signaling mechanism. The request is forwarded to the admission control mechanism that decides to accept or reject the application based on the application's QoS requirements, available resources, performance criteria, and network policy;
- Implicit admission control.
There is no explicit resource reservation signaling. The admission control mechanism relies on bandwidth overprovisioning and traffic control (i.e., traffic policing).

In addition to the admission control mechanisms, the COPS protocol is implemented in order to enhance the functionality of RSVP. The COPS protocol has been specified for RSVP by RAP (Resource Allocation Protocol) working group of IETF. The protocol is part of the Internet Protocol suite as defined by the IETF's RFC 2748. It is a simple query and response protocol that can be used to exchange policy information between a policy server (PDP) and its clients (PEPs). One example of a policy client is an RSVP router that must exercise policy-based admission control over RSVP usage.

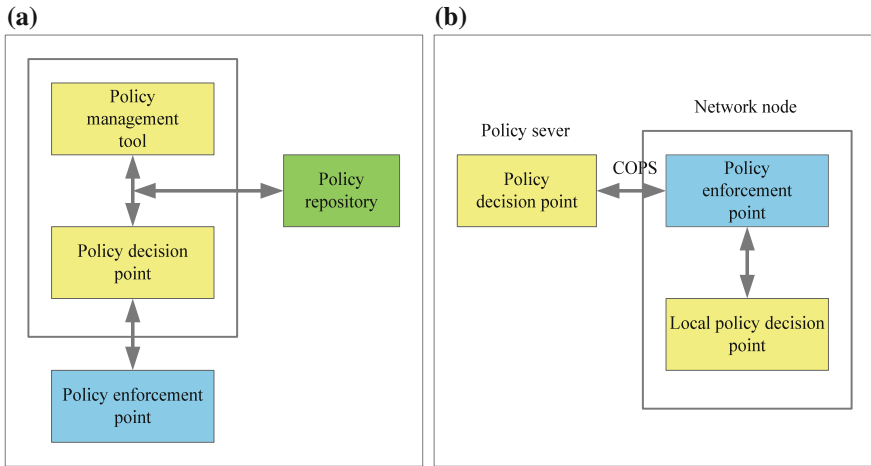


Fig. 2.6 **a** Intra-domain policy management; **b** Common open policy service

The COPS protocol specifies a simple client/server model for supporting policy control over QoS signaling protocols (e.g., RSVP). Policies are stored on servers, and acted upon by PDP, and are enforced on clients (PEP) as shown Fig. 2.6. At least one policy server exists in each controlled administrative domain. COPS protocol has a simple but extensible design.

The main characteristics of COPS include the following [26]:

- The protocol employs a client/server model where the PEP sends requests and updates to the remote PDP and deletes and the PDP returns decisions back to the PEP;
- The protocol uses TCP as its transport protocol for the reliable exchange of messages between policy clients and a server. Therefore, no additional mechanisms are necessary for the reliable communication between a server and its clients;
- The protocol is extensible in that it is designed to leverage off self-identifying objects and can support diverse client-specific information without requiring modifications to the COPS protocol itself. The protocol was created for the general administration, configuration, and enforcement of policies;
- COPS protocol provides message level security for authentication, replay protection, and message integrity. The protocol is stateful in two main aspects:
 - Request/Decision state is shared between client and server—Requests from the client PEP are installed or remembered by the remote PDP until they are explicitly deleted by the PEP. At the same time, decisions from the remote PDP can be generated asynchronously at any time for a currently installed request state;

- State from various events (Request/Decision pairs) may be inter-associated—The server may respond to new queries differently because of the previously installed Request/Decision state(s) that are related.

Additionally, the protocol is stateful in that it allows the server to push configuration information to the client, and then allows the server to remove such state from the client when it is no longer applicable.

From the point of view of the client or PEP, there are two models of COPS: the outsourcing model and the provisioning model.

- The outsourcing model is the simplest COPS implementation. In this model, all policies are stored at the PDP. Whenever the PEP needs to make a decision, it sends all relevant information to the PDP. The PDP analyzes the information, makes the decision, and relays it to the PEP. The PEP then simply enforces the decision;
- In the provisioning model, COPS is used for policy provisioning (COPS-PR), and the PEP reports its decision-making capabilities to the PDP. The PDP then downloads relevant policies to the PEP. The PEP can then make its own decisions based on these policies. The provisioning model uses the policy information base as a repository for the policies.

The main benefits of RSVP are as follows:

- It signals QoS requests per individual flow. The network can then provide guarantees to these individual flows. However, in the same time, the problem of RSVP is that it does not scale to large networks because of the large numbers of concurrent RSVP flows;
- It informs network devices of flow parameters (IP addresses and port numbers). Some applications use dynamic port numbers, which can be difficult for network devices to recognize. Hence, the additional mechanism must be used to supplement RSVP for applications that use dynamic port numbers but do not use RSVP.

The main drawbacks of RSVP are as follows:

- Continuous signaling due to the stateless operation of RSVP;
- RSVP is not scalable to large networks where per-flow guarantees would have to be made to thousands of flows.

The major advantage of IntServ is that it provides service classes which closely match different application types. However, although IntServ is a straightforward QoS model, end-to-end service guarantees cannot be supported unless all nodes along the route support IntServ. Therefore, scalability is a key architectural concern for IntServ, since it requires end-to-end signaling and must maintain a per-flow state at every router on the path.

One way of solving this problem is by using a multilevel approach, where per-microflow resource reservation (i.e., resource reservation for individual users) is done in the edge network, while in the core network, resources are reserved for

aggregate flows only. The routers that lie between these different levels must adjust the amount of aggregate bandwidth reserved from the core network so that the reservation requests for individual flows from the edge network can be better satisfied. This approach is known as a hybrid model RSVP-DS (Resource Reservation Protocol with DiffServ) and is implemented as a backbone service architecture concept.

2.3.2 *DiffServ Model*

Differentiated services or DiffServ is a computer networking architecture that specifies a simple, scalable, and coarse-grained mechanism for classifying, managing network traffic, and providing QoS guarantees on modern IP networks [20].

The architecture describes services and allows for more user-defined services to be used in a DiffServ-enabled network. DiffServ operates on the principle of traffic classification, where each data packet is placed into a limited number of traffic classes, rather than differentiating network traffic based on the requirements of an individual flow. A class can be identified as a single application or it can be identified based on source or destination IP address (the most common case). Each traffic class can be managed differently, ensuring privileged treatment for higher priority traffic on the network. DiffServ recommends a standardized set of traffic classes for ensuring simpler interoperability between different networks and different vendors' equipment.

The idea behind this framework is for the network to recognize a class without having to receive any requests from applications as it is the case within the Intserv model. This allows the QoS mechanisms to be applied to other applications that do not have the RSVP functionality, which is the case with the majority of applications that use IP protocol. DiffServ can, for example, be used to provide low latency to critical network traffic such as voice or video, while providing simple best-effort traffic guarantees to non-critical services such as Web traffic or file transfers.

In DiffServ model, ToS (Type of Service) field is now called the DS (Differentiated Services) field (RFC 2474 [27]). The DS field consists of a differentiated services code point (DSCP) field (6 bits) and explicit congestion notification (ECN) field, which occupies the least significant two bits, Fig. 2.7. The six bits replace the three IP precedence bits, and is called the DSCP but maintains interoperability with non-DS compliant devices (those that still use IP precedence). Because of this backward compatibility, DiffServ can be gradually deployed in large networks.

Namely, DiffServ uses the 6-bit DSCP field in the header of IP packets for packet classification purposes as shown in Fig. 2.7. DSCP replaces the previously used IP precedence, a 3-bit field in the ToS byte of the IP header originally used for classifying and prioritizing types of traffic. In contrast to IntServ, which is defined as a flow-based mechanism, DiffServ is a class-based mechanism for traffic

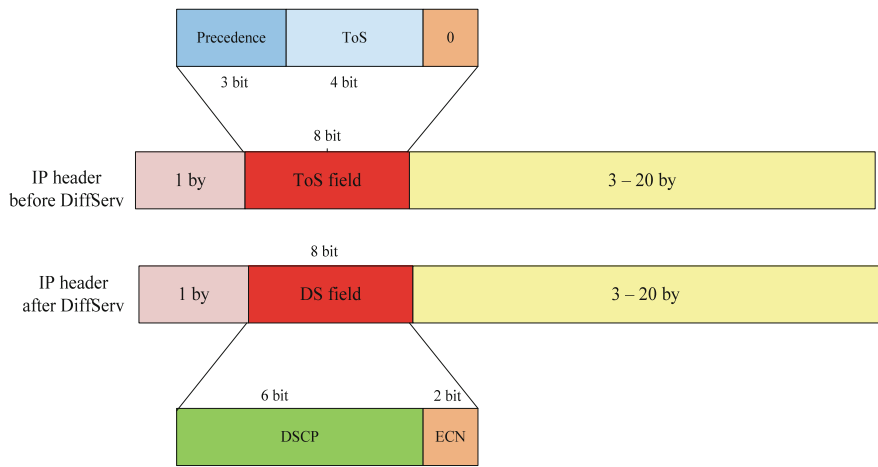


Fig. 2.7 ToS field

management which provides a framework for classification and differentiated treatment. Unlike the IP precedence solution, the ToS byte is completely redefined.

With DSCP, in any given node, up to 64 different aggregates/classes can be supported (2^6), which allows finer traffic segregation compared to IP precedence and eight supported traffic classes (2^3). Every classification and QoS revolves around the DSCP in the DiffServ model. The 64 possible values of DSCP field are divided into three groups (DSCP pools). The two groups that comprise 32 possible DSCP values are reserved for experimental use. The third group includes the other 32 values, where 21 values are defined in the DiffServ specification and are to be used when forwarding traffic as shown in Fig. 2.8.

Furthermore, the DSCP field is divided into two sub-fields of the three-bit field, called the class selector (CS) and the drop probability (DP) field that defines the probability of rejection of a given packet within a given class in the event of congestion in the network as shown in Fig. 2.9. DS0 bit of DP field is set to zero in

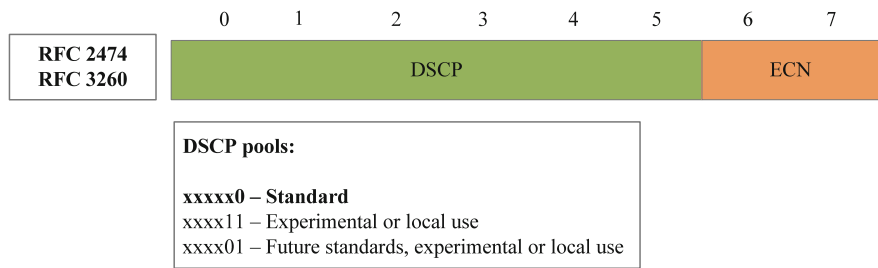


Fig. 2.8 DiffServ pool

ToS byte	P2	P1	P0	T3	T2	T1	T0	ZERO
DS byte	DS5	DS4	DS3	DS2	DS1	DS0	ECN1	ECN0
	Class selector (CS)			Drop probability (DP)				

Fig. 2.9 DSCP definition

accordance with the previous explanation and the definition of the standard DSCP pool which is currently used in IP networks. The definition of CS field provides compatibility with earlier models (IP Precedence definition) as shown in Table 2.4. However, the use of the DS field can define multiple classes of traffic and a finer classification of traffic in the network but this model does not incorporate definitions of which type of traffic should get priority treatment, and this is consequently left to be determined by network operators.

Each router on the network is configured to differentiate traffic based on its class. The collection of packets that have the same DSCP value (also called a code point) in them, and which are crossing in a particular direction, is called a behavior aggregate (BA). Packets from multiple applications/sources could belong to the same BA.

Furthermore, DiffServ-aware routers implement per-hop behaviors (PHBs), which define the packet forwarding properties associated with a certain class of traffic. Generally, traffic may be classified by many different parameters, such as source address, destination address, or traffic type and assigned to a specific traffic class. According to the standard (RFC 2474), the PHB is determined by the differentiated services (DS) field of the IPv4 header. Now, BA identifies packets marked with the same DSCP where PHB is applied to each BA according to the QoS policy.

However, the RFC standards do not dictate the manner of implementing PHBs, and it is the responsibility of the vendor. In more concrete terms, a PHB refers to the packet scheduling, queuing, policing, or shaping behavior of a node on any given packet belonging to a BA, as configured by a SLA (service-level agreement) or policy. For example, a vendor can implement queuing techniques that can base their PHB on the IP precedence or DSCP value in the IP header of a packet. Based on DSCP or IP precedence, traffic can be put into a particular service class where packets within this service class are treated in the same way. As we have explained earlier, in theory, the network could have up to 64 (i.e., 2^6) different traffic classes using different markings in the DSCP. The DiffServ RFC standards recommend, but do not require, certain encodings which give a network operator great flexibility in defining traffic classes. To date, four standard PHBs have been available for constructing a DiffServ-enabled network and achieving coarse-grained, end-to-end CoS (class of service) and QoS (Fig. 2.10):

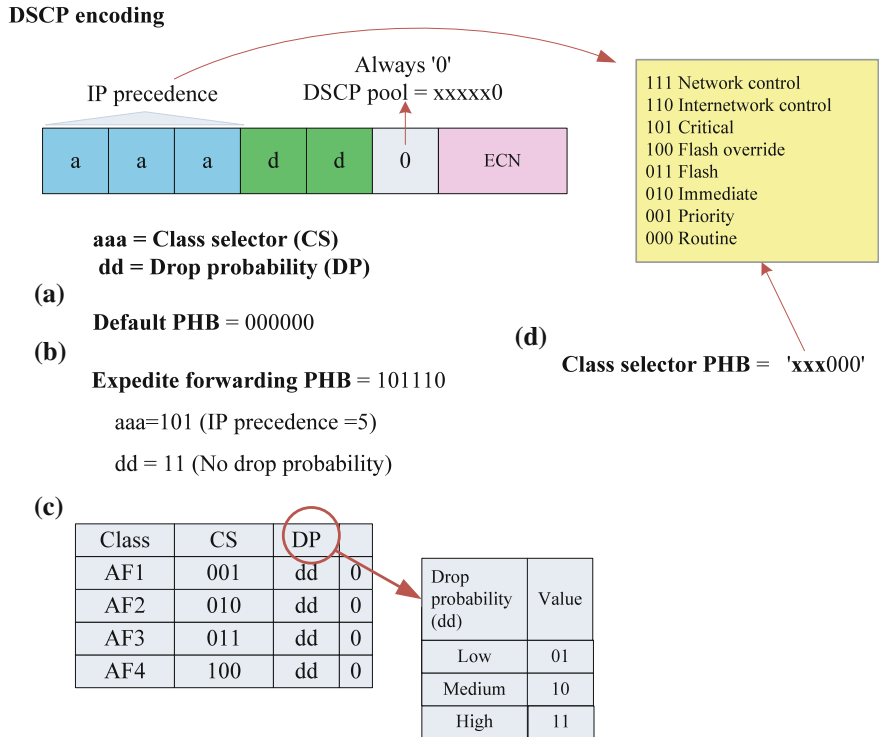


Fig. 2.10 PHB definition

- Default PHB—which is typically best-effort traffic;
- Expedited forwarding (EF) PHB—dedicated to low-loss, low-latency traffic;
- Assured forwarding (AF) PHB—gives assurance of delivery under prescribed conditions;
- Class selector PHBs—which maintain backward compatibility with the IP precedence field.

A. Default PHB

A default PHB is typically best-effort traffic. Any traffic that does not meet the requirements of any of the other defined classes is classified as the default PHB. In most cases, the default PHB has best-effort forwarding characteristics. The recommended DSCP for the default PHB is '000000' (in binary) as shown in Fig. 2.10a.

B. Expedited forwarding (EF) PHB

EF PHB is dedicated to low latency, low packet loss traffic, as defined in RFC 3246 [28]. The expedited forwarding PHB is identified based on the following parameters:

- Ensures a minimum departure rate to provide the lowest possible delay to delay-sensitive applications;
- Guarantees bandwidth to prevent starvation of the application if there are multiple applications using expedited forwarding PHB;
- Polices bandwidth to prevent starvation of other applications or classes that are not using this PHB.

The EF PHB is characterized with low delay, low packet loss and low jitter and is suitable for voice and other real-time services. Since an overload of EF traffic will cause queuing delays and affect the jitter and delay tolerances, a strict priority queuing is in most cases implemented for this class. The recommended DSCP binary value for packets requiring expedited forwarding is 101110 (46 or $0 \times 2E$ in decimal and hexadecimal system, respectively). Non-DS compliant devices will regard EF DSCP value as IP precedence 5 (101), which is the highest user-definable IP precedence, and is typically used for delay-sensitive traffic such as voice over IP as shown in Fig. 2.10b.

C. *Assured forwarding (AF) PHB*

The IETF defines the assured forwarding behavior in RFC 2597 [29] and RFC 3260 [30]. This class gives assurance of delivery under defined conditions, i.e., provides assurance of delivery as long as the traffic does not exceed a certain subscribed rate. Traffic that exceeds the subscription rate faces a higher probability of being dropped if congestion occurs. Different multimedia applications, which are characterized by variable bit rate and are sensitive to packet loss, are classified as AF PHB.

The assured forwarding PHB is identified based on the following parameters:

- Guarantees a certain amount of bandwidth to an AF class;
- Allows access to extra bandwidth, if available;
- Packets requiring AF PHB should be marked with DSCP value 'aaaddO' where 'aaa' is a binary value of the class and 'dd' is the drop probability (Fig. 2.10c).

As a result, and in contrast to the two previous classes, the standard defines four separate AF classes within the AF group (AF1, AF2, AF3, and AF4). Within each class, packets are given drop precedence (high, medium, or low), i.e., within each class, the standard now defines three subclasses with a different drop probability. For example, within the AF1 class, the standard now defines three different subclasses AF11, AF12, and AF13, each with a different drop probability as shown in Fig. 2.10c. The combination of classes and drop precedence gives twelve separate DSCP encodings, from AF11 to AF43 as shown in Table 2.3.

In case of congestion between classes, the traffic in the higher class is given priority. For example, AF4 class has higher priority than class AF3. If congestion occurs within a class, the packets with the higher drop precedence are discarded first. For example, the probability of dropping the packets that belong to the subclass AF33 is greater than the probability of dropping packets belonging to the

Table 2.3 AF PHB definition

PHB	Class selector	Drop probability	DS code point	DSCP bit (binary)
AF	1	1: Low	AF11	001 010 (10)
		2: Medium	AF12	001 100 (12)
		3: High	AF13	001 110 (14)
AF	2	1: Low	AF21	010 010 (18)
		2: Medium	AF22	010 100 (20)
		3: High	AF23	010 110 (22)
AF	3	1: Low	AF31	011 010 (26)
		2: Medium	AF32	011 100 (28)
		3: High	AF33	011 110(30)
AF	4	1: Low	AF41	100 010 (34)
		2: Medium	AF42	100 100 (36)
		3: High	AF43	100 110 (38)

subclass AF31, i.e., the priority of a given AF33 subclass packet is lower in comparison with the packet that belongs to the AF31 subclass as shown in Fig. 2.10c.

To allow the AF PHB to be used in many different operating environments, the dropping algorithm control parameters must be independently configured for each defined packet drop precedence value and for each AF class. In practice, this specification allows implementation of various configurations that define policies for packet discard. Moreover, an AF implementation must attempt to minimize long-term congestion within each class, while allowing short-term congestion (usually resulting from bursts). Accordingly, in practice, AF class uses more balanced queue servicing algorithms than , such as fair queuing or weighted fair queuing.

D. Class Selector PHBs

DiffServ defines the class selector PHB in order to maintain backward compatibility with network devices that are still using the precedence field. The class selector code points are of the form ‘xxx000’ as shown in Fig. 2.10d. The first three bits are the IP precedence bits. Each IP precedence value can be mapped into a DiffServ class. If a packet is received from a non-DiffServ-aware router that used IP precedence markings, the DiffServ router can still understand the encoding as a class selector code point. Table 2.4 summarizes DSCP specification and backward compatibility with the IP precedence definition.

As we have previously explained, the QoS mechanism specifies a guaranteed network data performance level, i.e., the mechanism which specifies the implementation of QoS includes the setup of the QoS parameters (bandwidth, average packet delay, jitter, and packet loss) in order to improve network performance and meet the demands of end-users. QoS guarantees are important if the network capacity is insufficient, especially for real-time streaming multimedia applications,

Table 2.4 DSCP specification

IP precedence (3 bits)			DSCP (6 bits)				
Name	Value	Bits	PHB	Class selector	Drop probability	DS code point	DSCP bit (binary)
Routine	0	000	Default				
Priority	1	001	AF	1	1: Low	AF11	001 010 (10)
					2: Medium	AF12	001 100 (12)
					3: High	AF13	001 110 (14)
Immediate	2	010	AF	2	1: Low	AF21	010 010 (18)
					2: Medium	AF22	010 100 (20)
					3: High	AF23	010 110 (22)
Flash	3	011	AF	3	1: Low	AF31	011 010 (26)
					2: Medium	AF32	011 100 (28)
					3: High	AF33	011 110 (30)
Flash override	4	100	AF	4	1: Low	AF41	100 010 (34)
					2: Medium	AF42	100 100 (36)
					3: High	AF43	100 110 (38)
Critical	5	101	EF			EF	101 110 (46)
Inter-network control	6	110					(48–55)
Network control	7	111					(56–63)

since they often require a fixed bit rate and are delay sensitive. In practice, for the successful transmission of voice, video applications, and data, QoS parameters' value must be within the defined ranges.

Voice applications have smooth transmission characteristics, and these applications are sensitive to delay, delay variation, and packet loss. Voice applications demand the following values of QoS parameters (one-way requirements):

- Latency lower than 150 ms;
- Jitter lower than 30 ms;
- Loss lower than 1%;
- Voice transmission requires guaranteed priority bandwidth per call. In practice, QoS implementation must provide 150 bps (+ Layer 2 overhead) guaranteed bandwidth for voice-control traffic per call.

In contrast to voice applications, video applications have bursty transmission characteristics but they are also sensitive to delay and packet loss rate. Video applications demand the following values of QoS parameters:

- Latency lower than 150 ms;
- Jitter lower than 30 ms;
- Loss lower than 1%;
- Minimum priority bandwidth guarantee required is as follows:

- Video stream plus 20% of required bandwidth;
- For example, a 384-kbps stream would require 460 kbps of bandwidth.

Data transmission requirements:

- Different applications have different traffic characteristics;
- Different versions of the same application can have different traffic characteristics.

Based on the previous analyses, one of the most frequently used QoS policies in a multiservice network defines the following traffic classes:

- Voice traffic—Low latency, low jitter, and low packet loss;
- Mission critical—Guaranteed latency and guaranteed delivery;
- Transactional traffic—Guaranteed traffic;
- Best-effort traffic—No delivery guarantee.

Different voice, video, and data applications are further classified and marked to belong to EF class, AF subclasses, and BE class. An example of the possible classification and marking in service provider’s network is presented in Fig. 2.11.

Application	L3 Classification			L2		SP Service Classes
	IPP	PHB	DSCP	CoS		
Routing	6	CS6	48	6	EF AF41 → CS5	Controlled Latency 35%
Voice	5	EF	46	5		
Videoconferencing	5	AF41	34	4		
Streaming video	4	AF42	36	4		
Mission-critical data	3	AF31	26	3	CS6 AF31 AF21→CS3	Controlled Load 1 43%
Call-signaling	3	AF32	28	3		
Transactional data	2	AF21	18	2		
Network management	2	CS2	16	2		
Bulk data	1	AF11	10	1	AF11 → AF21	Controlled Load 2 15%
Scavenger	1	CS1	8	1		
Best effort	0	0	0	0		Best Effort 7%

Fig. 2.11 Example of DSCP implementation

2.4 QoS Mechanisms

QoS mechanisms represent a set of techniques or tools which allow traffic classification in an IP network based on models of differentiated services and enable a given family of basic parameters of QoS (bandwidth, delay, delay variation, and packet loss percentage) to be maintained in pre-defined boundaries on a source to destination packets' route. QoS mechanisms can be divided into three main categories (Fig. 2.12):

- A. Classification;
- B. Scheduling;
- C. Network provisioning.

2.4.1 Classification

As previously explained, the lowest service level that a network can provide is the best-effort service, which does not provide QoS support. Within the best-effort service, all traffic is handled equally regardless of the application or host that generated the traffic. However, some applications need QoS support, requiring better than best-effort service, such as differentiated or guaranteed service.

For a network to provide selective services to certain applications, first of all, the network requires a classification mechanism that can differentiate between different applications. The classification mechanism identifies and separates different traffic into flows or groups of flows (aggregated flows or classes) for identifying a behavior aggregate to which a packet belongs. Therefore, each flow or each aggregated flow can be handled selectively.

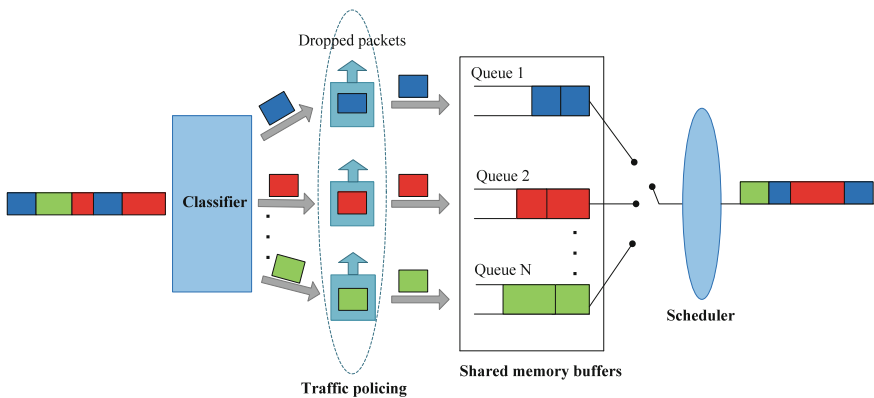


Fig. 2.12 QoS mechanisms

As explained above, packages can be classified either within the second layer of the OSI (Open Systems Interconnection) model using CoS bits (defined in the tag field of the Ethernet frame header [31]), or within the third layer of the OSI model using IP precedence or DSCP bits (defined in the ToS filed of the IP packet header) as shown in Fig. 2.13. The classification of packages should be made as close as possible to the source package so that they would have appropriate treatment in the path through the network. The classification mechanism can be implemented in different network devices (end hosts, intermediate devices such as switches, routers, access points). Figure 2.14 shows a simplified diagram of a classification module that resides on the end host and on the intermediated device.

Application traffic (at the end host), or incoming traffic from other hosts (at the intermediate device), is identified by the classification mechanism and is forwarded to the appropriate queue awaiting service from other mechanisms, for example, the packet scheduler. The granularity level of the classification mechanism can be per-user, per-flow, or per-class depending on the type of QoS services provided. For example, per-flow QoS service requires per-flow classification while per-class QoS service requires per-class classification. To identify and classify the traffic, the traffic classification mechanism requires some type of tagging or marking of the packets. There are a number of traffic classification approaches. Some of approaches are suitable for end hosts and some for intermediate hosts. Figure 2.13 shows an example of some traffic classification approaches which are implemented in the different OSI layers.

Most QoS mechanisms include some type of classification, but a small number of mechanisms also include marking capability as shown in Fig. 2.15.

OSI model	TCP/IP model	Classification Techniques
Application layer	Application layer	User/Application identification
Presentation layer		
Session layer		
Transport layer	Transport layer	Flow
Network layer	Internet layer	IP ToS, DSCP
Data-link layer	Network access layer	802.1P/Q Classification
Physical layer		

Fig. 2.13 Example of the existing classification on each layer of OSI and TCP/IP model

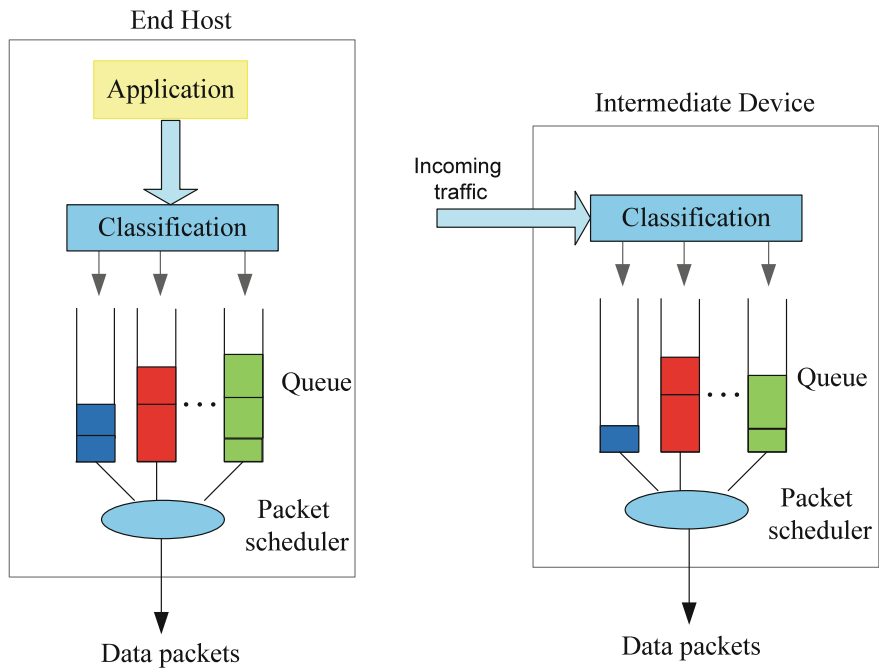


Fig. 2.14 Implementation of classification in network devices

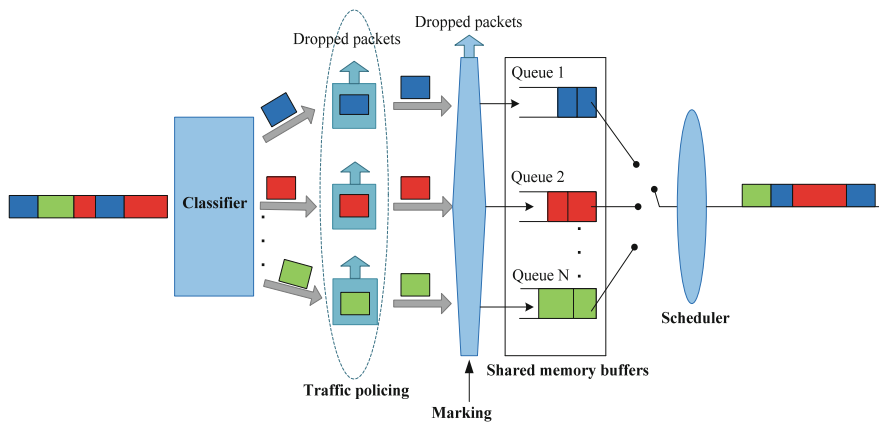


Fig. 2.15 Classification and marking

Marking is a term used for coloring packets by applying a class-identifying value to one of the following markers: IP precedence, DSCP, QoS group (value is local to a router), MPLS experimental bits (can be used only in MPLS-enabled networks), ATM CLP bit (value can be used only within ATM networks), Frame Relay DE bit

(value can be used only within Frame Relay networks), or IEEE 802.1Q priority bits (value can be used within LAN-switched networks) [31]. In practice, marking is used after classification and traffic policing. Traffic policing configuration has to determine whether the packet is in a pre-configured profile or out of the profile. Based on that information and the configured parameters, marking configuration must determine whether to pass through, mark down (change the DSCP, or CoS, or any of the previously mentioned parameters), or drop the packet. Furthermore, packets are queued based on the pre-configured (marked) parameters and scheduled for transmission.

2.4.2 Scheduling Mechanism

Packet scheduling is the mechanism that selects a packet for transmission from the packets waiting in the transmission queue. Scheduling mechanism decides which packet from which queue and station are scheduled for transmission in a certain period of time. Packet scheduling controls bandwidth allocation to stations, classes, and applications.

As shown in Fig. 2.16, there are two levels of packet scheduling mechanisms:

- Intra-station packet scheduling: the packet scheduling mechanism that retrieves a packet from a queue within the same host;
- Inter-station packet scheduling: the packet scheduling mechanism that retrieves a packet from a queue from different hosts.

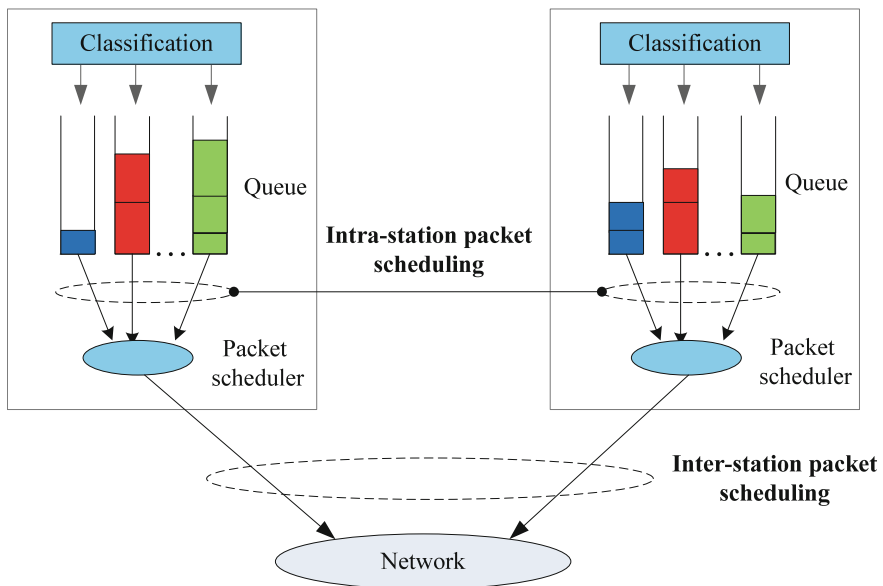


Fig. 2.16 Packet scheduling mechanism

In the intra-station packet scheduling, the status of each queue in a station is known, and hence, the intra-station packet scheduling mechanism is identical to a queuing mechanism. On the contrary, the inter-station packet scheduling mechanism is different from a queuing mechanism because queues are distributed among hosts and there is no central knowledge of the status of each queue. Therefore, inter-station packet scheduling mechanisms require a signaling procedure to coordinate the scheduling among hosts and resource allocation. Because of the similarities between packet scheduling and queuing mechanisms, we further introduce four queuing schemes: FIFO (First In, First Out), SPQ (Strict Priority Queuing), CQ (Custom Queuing), and WFQ (Weight Fair Queuing) and discuss how they support QoS services.

A. *Queuing mechanism*

Transmission buffers in network switches and routers tend to fill rapidly in high-speed networks, and therefore buffering, not bandwidth, represents the key QoS issue. The overloaded buffers cause packet drops, which in turn causes voice or video clipping and skipping. Every point in the network where there is a router or a switch may be a source of transmission or buffering challenges potentially giving rise to poor QoS.

There are several approaches to solving a problem of insufficient bandwidth:

- The best approach is to increase link capacity to accommodate all applications and users with some extra bandwidth to spare. This solution sounds simple enough, but in practice, it brings high costs in terms of money and time it takes to be implemented. Oftentimes, there are also technological limitations related to the process of upgrade to a higher bandwidth;
- Another option is to classify traffic into QoS classes and place packets into queues which are then processed on a priority basis or in accordance with classes. Since a separate queue for each class of traffic is defined, it enables the achievement of appropriate levels of service quality and greater control over the operation of networks. Since this is a cost-effective solution, we further discuss the queuing algorithms below.

Queuing on routes is necessary to accommodate burst and congestion when the arrival packet rate is greater than the departure rate. This is usually the case because the input interface is faster than the output interface, or because the output interface receives packets from other multiple input interfaces. The initial implementation of queuing used a single FIFO queue. More complex queuing mechanism includes the definition of following terms (Fig. 2.17):

- Hardware queue—always uses FIFO strategy, which is necessary for the interface drivers to transmit packet one by one;
- Software queue—schedules packets into the hardware queue based on the QoS requirements and configuration.

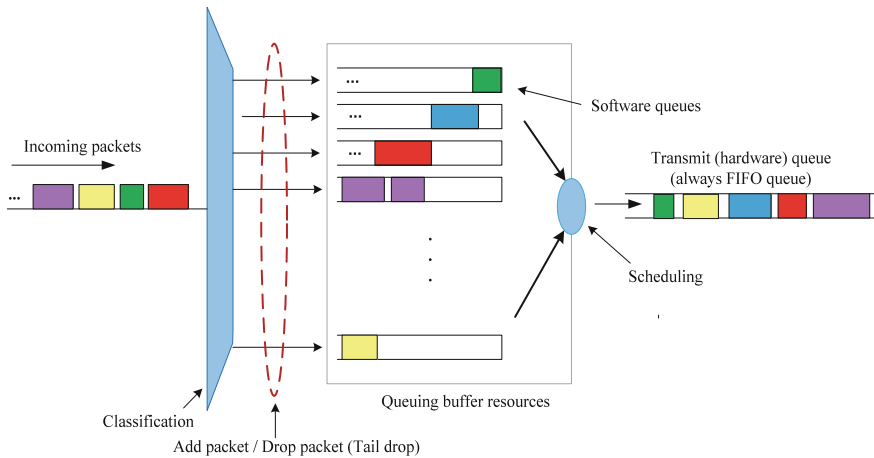


Fig. 2.17 Queuing implementation

Generally, a full hardware queue indicates interface congestion and software queuing is utilized to manage it. When a packet is being forwarded, the router will bypass the software queue if the hardware queue has space in it (no congestion). The implementation of software queuing is optimized for periods when the interface is not congested.

Most queuing mechanisms include the classification of packets. After a packet is classified, a router has to determine whether it can put the packet into the queue or it has to drop the packet. Most queuing mechanisms will drop a packet only if the corresponding queue is full. This is called tail drop. Some mechanisms use a more intelligent dropping scheme, such as random early detection (RED) [20]. If the packet is allowed to be queued, it will be put into the FIFO queue for that class or flow. Packets are then taken from the individual per-class queues and put in the hardware queue.

In practice, to enable queuing, traffic has to be classified according to importance, i.e., business-critical traffic should get enough bandwidth, voice should get enough bandwidth and prioritized forwarding, and the least important traffic should get the remaining bandwidth. Based on each data packet's classification, the packet is placed into the appropriate transmission queue. Time critical voice and video data are classified in such a way that they are placed into a delay- and drop-sensitive queue. Separate queues allow time critical data, such as audio and video, to be transmitted in a priority fashion. Queuing gives the delay-sensitive voice and video data a higher priority in the network switch or router ensuring that the voice or video packet will be transmitted in a timely manner.

In case of network congestion, the queuing mechanism avoids the unnecessary dropping of packets by storing them for a while, hoping that congestion will be alleviated, and that they can be then dispatched. When congestion is heavy and the queue overflows, new arriving packets have to be dropped since there is no other

choice for them. There is a wide variety of available mechanisms that provide bandwidth guarantees but, as we have previously mentioned, we will focus on the FIFO queuing algorithm as the default one among the majority of network equipment, priority queuing, custom queuing, and weighted fair queuing.

B. *FIFO (First In, First Out) queuing*

FIFO is the default queuing discipline used by most vendors. In fact, it has its benefits and limitations, but it is still very simple for implementation. Packets arriving from different flows are treated equally in terms of their placement into the queue and their order of arrival is strictly respected. While still being in the queue, they are dispatched in the same order they entered, which means that the first packet that comes in is the first packet to go out as shown in Fig. 2.18.

FIFO queuing algorithm is the simplest one among the congestion management methods. All packets are treated equally, placed into a single queue, and serviced in the order they were received. FIFO provides best-effort service, i.e., it does not provide service differentiation in terms of bandwidth and delay. The high bandwidth flows will get a larger bandwidth portion in comparison with the low bandwidth flows. Aggressive flows send a large number of packets which occupy the majority of bandwidth. Time-sensitive flows send a modest number of packets, and most are dropped due to the queue always being full of aggressive flow packets. It is possible to improve QoS support by adding a traffic policing mechanism, for limiting the rate of each flow, and by implementing admission control. In practice, the size of queue affects delay, jitter, and packet loss. Namely, the queue has a finite

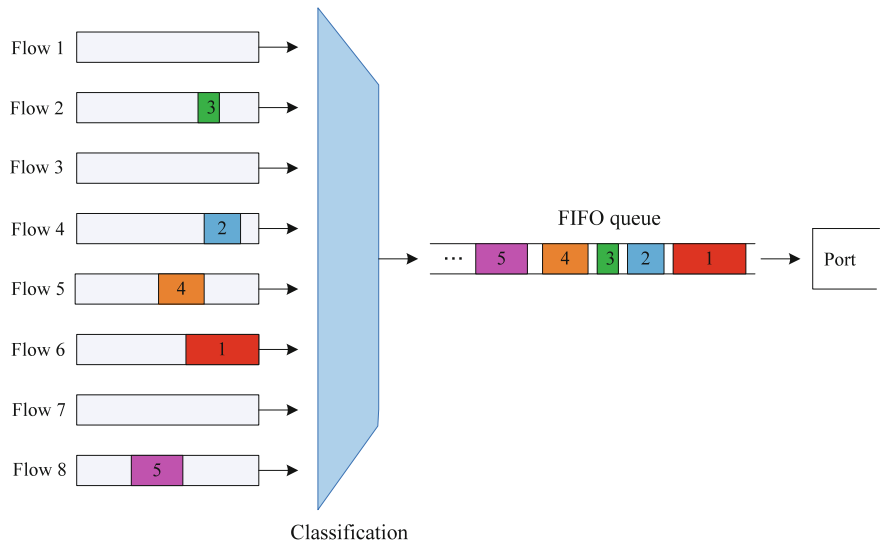


Fig. 2.18 FIFO queuing

size; hence, in the case of congestion, the packet that needs to be added to the queue could be dropped (tail drop mechanism). To overcome this problem, the queue could be lengthened, which decreases the probability of tail drop. With a longer queue, however, the average delay and jitter increases, because packet may be queued behind a large number of other packets.

FIFO queuing offers the following benefits:

- FIFO queuing places an extremely low load on the system in comparison with other queuing mechanisms;
- FIFO queuing is predictable, and delay is determined by the maximum length of the queue;
- FIFO queuing does not add significant queuing delay at each hop as long as the queue length remains low.

FIFO queuing also poses the following limitations:

- FIFO queuing is extremely unfair when an aggressive flow contests with a time-sensitive flow. Aggressive flows send a large number of packets, many of which are dropped. Time-sensitive flows send a modest number of packets, and most are dropped due to the queue always being full of aggressive flow packets. This behavior is called starvation;
- When the queue is full, packets entering the queue have to wait longer. When the queue is not full, packets entering the queue do not have to wait until;
- A single FIFO queue does not allow routers to organize buffered packets and service one class of traffic differently from other classes of traffic. Therefore, FIFO queuing can result in increased delay, jitter, and a reduction in the amount of output bandwidth consumed by TCP applications.

C. *Priority Queuing*

Priority packet scheduling schedules packets based on the assigned priority order. Packets in higher priority queues are always transmitted before packets in lower priority queues. A lower priority queue has a chance to transmit packets only when there are no packets waiting in a higher priority queue.

As shown in Fig. 2.19, a scheduler first empties the higher priority queue, next the medium one, and finally the low priority one, i.e., as soon as there are packets in the high priority queue, it must be served first until it becomes empty, then the medium priority queue and finally the low priority one. SPQ provides differentiated services in both bandwidth and delay. The highest priority queue always receives bandwidth (up to the total bandwidth), and the lower priority queues receive the remaining bandwidth. Therefore, higher priority queues always experience lower delay than the lower priority queues. The aggressive use of bandwidth by the high priority queues can starve the low priority queues. As it was the case with the FIFO algorithm, the QoS support can be improved by traffic policing, which limits the rate of each flow, and by the implementation of the admission control mechanism.

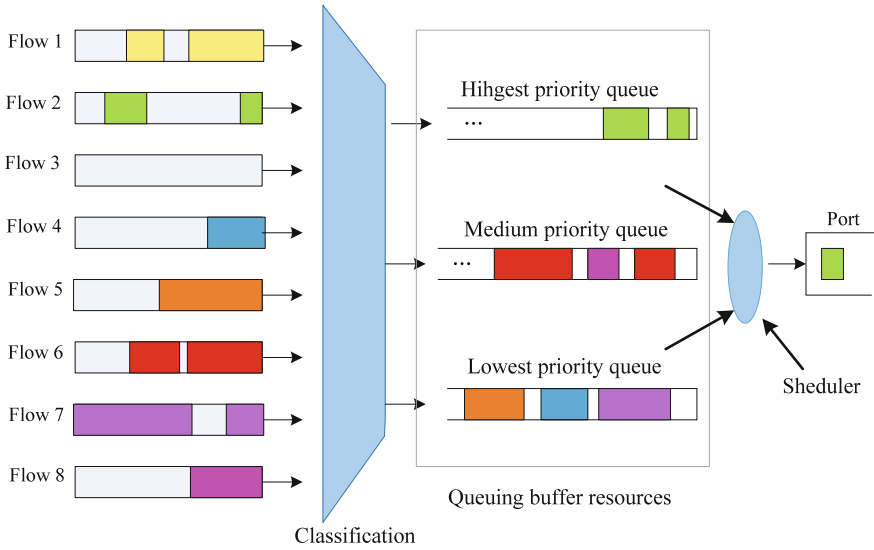


Fig. 2.19 Implementation of priority queuing

D. Custom Queuing

CQ is one of the most popular queuing strategies. CQ was originally implemented to address the clear shortcomings of PQ. It enables us to configure how many queues are to be used, what applications will use which queues, and how the queues will be serviced. While PQ has only four queues, CQ is composed of 17 queues, numbered from 0 to 16. Queue 0 is a system queue reserved for the control traffic, and hence, application data cannot starve the critical network control traffic. Consequently, this queue cannot be set by the user and the remaining queues, queues 1–16, are the settable queues.

Custom queuing mechanism is based on the weighted round-robin (WRR) approach [20]. In this approach, packets are accessed in the round-robin style but queues can be given priorities called ‘weights.’ The idea behinds CQ is the assignment of the percentage of the available bandwidth to each of the 16 queues until the 100% of total availability is reached and this is achieved by the application of the ‘weight’ definition. Namely, the WRR approach places a certain weight on various queues to service a different number of bytes or packets from the queues during a round-robin cycle. WRR mechanism implemented in the custom queuing takes a certain predetermined amount of bytes defined by the weight factor from each queue during each pass as shown in Fig. 2.20. In practice, weight factors, i.e., number of bytes, can be defined and this allows administrators to approximately specify how much of the bandwidth each queue will receive.

However, although more efficient than the previously described schemes, there are still three main problems that exist in custom queuing.

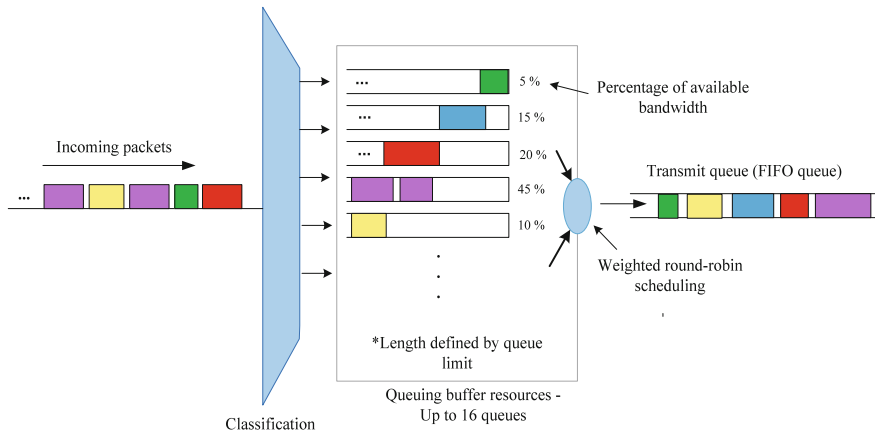


Fig. 2.20 Implementation of custom queuing

The first problem of custom queuing arises from the fact that weight of the queue is configured in bytes, and therefore, the accuracy of the WRR depends on the weight (byte count) and the MTU (Maximum Transmission Unit). Namely, the defined percentage shares cannot be used to set the queues, and instead, the ‘byte count,’ the value of which is dependent on flow’s packet size in the queue, must be used. If the ratio between the byte count and the MTU is too small, the WRR queuing will not allocate bandwidth accurately. On the other hand, if the ratio is too large, WRR will cause long delays. Since queues may have different average packet sizes (e.g., voice packets of 60 bytes and TCP packets of 1500 bytes), this may lead to the undesirable bandwidth distribution ratio. For example, if queue byte counter is 100 bytes and a packet of 1500 bytes is in the queue, the packet will still be sent anyway since the counter is nonzero. As a result, the queue will receive fourteen times more bandwidth (1400 bytes more instead of 100 bytes) in this round than it should have received. This is the consequence of the limitation of the round-robin scheduling that cannot take less than one packet from the each queue. In order to make the distribution fair, every queue’s byte counter should be proportional to the queue’s average packet size, which is, in practice, very difficult to achieve.

The second problem is the classification, i.e., the implementation of the CQ scheme. The classification can be based on different parameters (protocol, source interface, source and destination addresses, and ports) where the control by address demands the device configuration be set in advance.

The last problem arises from the fact that in CQ, the traffic within each queue competes directly with all other traffic in the same queue. So, for example, if one user sends a burst of application traffic that fills one of the queues, this will cause tail drops for other users whose traffic occupies the same queue. This will cause a smaller version of the global problem of a FIFO queue.

E. Weighted Fair Queuing

WFQ is a flow-based queuing algorithm used in QoS that schedules interactive traffic to the front of the queue to reduce response time, and it fairly shares the remaining bandwidth between high bandwidth flows as shown in Fig. 2.21. A stream of packets within a single session of a single application is known as flow or conversation. WFQ is a flow-based method that sends packets over the network and ensures packet transmission efficiency which is critical for interactive traffic. This method automatically stabilizes network congestion between individual packet transmission flows.

WFQ was introduced as a solution to the problem for the previously described FIFO queuing mechanism, including starvation, delay, and jitter. Moreover, it successfully resolves the problems introduced in PQ and CQ mechanisms, starvation of other lower priority classes, and long delays among others. For scheduling purposes, in WFQ, the length of the queue is not measured in packets but by means of the time it would take to transmit all the packets in the queue. WFQ adapts the number of flows and allocates equal amounts of bandwidth to each flow. Small packet flows which are usually interactive flows (voice and video) typically receive better service because they do not need a lot of bandwidth. Weight can be determined as the required QoS (IP precedence, RSVP), or inversely proportional flow throughput. For example, a link data rate of R , with N active data flows with weights $w_1, w_2, \dots, w_N$ and data flow number i , will achieve an average data rate of

$$R_i = \frac{Rw_i}{(w_1 + w_2 + \dots + w_N)} \quad (2.1)$$

The weight factors can be calculated from the IP precedence value. The result is that the router gives flows with higher IP precedence values a larger share of the

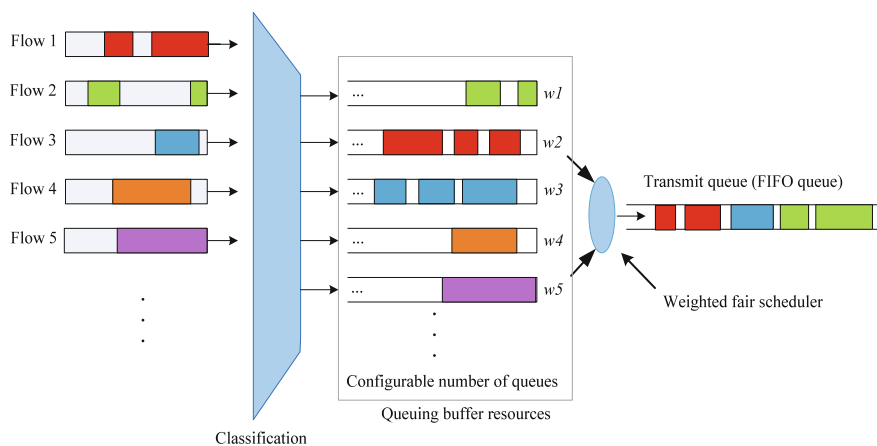


Fig. 2.21 Implementation of weighted fair queuing

bandwidth than those with lower precedence. For example, a flow with flash override IP precedence will get five times the bandwidth of a packet with routine IP precedence. After calculation, a weighting factor is assigned to each competing connection prior to data transmission. The weighting factor is chosen proportionally to the connection's bandwidth share. The WRR scheduler serves all connections in a cyclic way (round-robin) where each connection is allowed to send only a number of packets according to its weight factor.

These fair queueing algorithms tend to do three things. First, they prevent individual flows from interfering with one another. Second, they tend to reduce queueing latency for applications with smaller packets. And third, they ensure that all of the packets from a given flow are delivered in the same order they were initially sent. It is a fact that, when using a network with WFQ, an end-to-end delay bound can be guaranteed. By regulating the WFQ weights dynamically, WFQ can be utilized for controlling QoS, for example, to achieve guaranteed data rate.

Although bandwidth is allocated fairly among all flows, unfairness is reinstated by introducing weight (IP precedence) to give proportionately more bandwidth to flows with higher weights. WFQ has to classify individual flows using the following information taken from the IP/TCP/UDP headers. These parameters are used as input for a hash algorithm that produces a fixed length number that is used as the index of the queue:

- Source IP address;
- Destination IP address;
- Protocol number to identify TCP or UDP;
- Type of service field;
- Source TCP/UDP port number;
- Destination TCP/UDP port number.

The main benefits of WFQ are as follows:

- Simple configuration (classification does not have to be configured);
- Guarantees throughput to all flows;
- Dedicated queues for each flow (referred to as conversations), messages are sorted into conversations reducing starvation, delay, and jitter within the queue;
- Allocating bandwidth fairly and accurately among all flows, reducing scheduling delay and guaranteeing service;
- Drops packets of most aggressive flows;
- Supported on most platforms.

The main drawbacks are as follows:

- All drawbacks of FIFO queueing within a single queue;
- Multiple flows can end up in one queue;
- Does not support the configuration of classification;
- Cannot provide fixed bandwidth guarantees;
- Performance limitations due to complex classification and scheduling mechanisms.

2.4.3 Network Provisioning

Networking provisioning requires a thorough network analysis to determine parameters for services that are being deployed in the network. The result of monitoring techniques and provisioning is the allocation of bandwidth among all classes in times of congestion. Services are implemented by defining PHB properties. PHBs are implemented by using the available QoS mechanisms in networks devices.

The main benefits of provisioning are as follows:

- QoS does not create bandwidth;
- QoS manages bandwidth usage among multiple classes;
- QoS gives better service to a well-provisioned class with respect to another class.

In order to be effective in providing network QoS, it is necessary to affect the provisioning and configuration of the traffic handling mechanisms consistently, across multiple network devices. Provisioning and configuration mechanisms include the following:

- RSVP signaling;
- Policy mechanisms and protocols;
- Management tools and protocols.

The Emerging WDM EPON

Radivojević, M.; Matavulj, P.

2017, XXI, 235 p. 152 illus., 151 illus. in color.,

Hardcover

ISBN: 978-3-319-54222-5