



Thi-Thanh-Mai Hoang

Computer Networks, the Internet and Next Generation Networks

A Protocol-based and
Architecture-based Perspective



PETER LANG

Computer Networks, the Internet and Next Generation Networks

European University Studies

Europäische Hochschulschriften
Publications Universitaires Européennes

Series XLI Computer Science

Reihe XLI Série XLI
Informatik
Informatique

Vol./Bd. 46



PETER LANG

Frankfurt am Main · Berlin · Bern · Bruxelles · New York · Oxford · Wien

Thi-Thanh-Mai Hoang

Computer Networks, the Internet and Next Generation Networks

A Protocol-based and
Architecture-based Perspective



PETER LANG

Internationaler Verlag der Wissenschaften

**Bibliographic Information published by the Deutsche
Nationalbibliothek**

The Deutsche Nationalbibliothek lists this publication in the
Deutsche Nationalbibliografie; detailed bibliographic data is
available in the internet at <http://dnb.d-nb.de>.

ISSN 0930-7311
ISBN 978-3-631-62156-1 (Print)
ISBN 978-3-653-01750-2 (E-Book)
DOI 10.3726/978-3-653-01750-2

© Peter Lang GmbH
Internationaler Verlag der Wissenschaften
Frankfurt am Main 2012
All rights reserved.

All parts of this publication are protected by copyright. Any
utilisation outside the strict limits of the copyright law, without
the permission of the publisher, is forbidden and liable to
prosecution. This applies in particular to reproductions,
translations, microfilming, and storage and processing in
electronic retrieval systems.

www.peterlang.de

Contents

1. Introduction.....	15
1.1 What is the Specific Feature of this Book?.....	15
1.2 What are the Contributions of this Book?.....	15
2. Fundamental of Computer Networks, the Internet and Next Generation Networks	18
2.1 Network Reference Models.....	18
2.1.1 OSI Reference Model.....	18
2.1.2 The TCP/IP Reference Model.....	22
2.2 Fixed-Mobile Convergence.....	24
2.2.1 Multimedia Networking over Internet	24
2.2.2 Next Generation Networks.....	27
2.2.3 Mobile Networks.....	28
2.3 Consequences for Network Planning.....	31
2.3.1 Traffic Demand Characterization.....	31
2.3.2 Quality of Service Requirements	32
2.4 Network Planning Consideration	34
2.4.1 Application Considerations.....	34
2.4.2 Infrastructure Consideration	35
3. Traffic Management and QoS Control	37
3.1 Error Control	38
3.1.1 Bit-level Error Control	38
3.1.2 Packet-level Error Control	40
3.1.2.1 Sequence Number	40
3.1.2.2 Acknowledgement.....	41
3.1.2.3 Retransmission Timer	42
3.1.2.4 Packet Retransmission	42
3.1.2.5 Automatic Repeat Request (ARQ).....	42
3.2 Multiple Access Control	44
3.2.1 Static Channel Allocation	45

3.2.1.1 Frequency Division Multiple Access	45
3.2.1.2 Time Division Multiple Access	46
3.2.2 Dynamic Channel Allocation.....	47
3.2.2.1 Dynamic Channel Allocation with Random Access.....	47
3.2.2.1.1 ALOHA and Slotted ALOHA.....	47
3.2.2.1.2 Carrier Sense Multiple Access.....	49
3.2.2.1.3 Carrier Sense Multiple Access with Collision Detection	51
3.2.2.1.4 Carrier Sense Multiple Access with Collision Avoidance.....	54
3.2.2.2 Dynamic Channel Allocation with Taking Turns	55
3.2.2.2.1 Poling Mechanism.....	56
3.2.2.2.2 Token Passing Mechanism.....	56
3.3 Traffic Access Control	56
3.3.1 Traffic Description	57
3.3.2 Traffic Classification.....	59
3.3.3 Traffic Policing and Traffic Shaping	59
3.3.3.1 Traffic Policing by using Token Bucket	59
3.3.3.2 Traffic Shaping by Using Leaky Bucket.....	60
3.3.4 Marking	61
3.3.5 Metering	61
3.4 Packet scheduling.....	63
3.4.1 Requirements.....	63
3.4.1.1 Resource Fair Sharing and Isolation for Elastic Connection Flows	63
3.4.1.2 Performance Bounds	64
3.4.2 Classification of Scheduling Disciplines	65
3.4.2.1 Work-conserving vs. Non-work-conserving.....	65
3.4.2.2 Scheduling for Elastic Flows vs. Real-time Flows	66
3.4.3 First-In-First-Out (FIFO)	67
3.4.4 Priority Scheduling.....	68
3.4.5 Generalized Processor Sharing	68

3.4.6 Round-Robin	70
3.4.7 Weighted Round Robin.....	70
3.4.8 Deficit Round Robin	71
3.4.9 Weighted Fair Queuing Scheduling.....	72
3.5 Congestion Control	74
3.5.1 Classification of congestion control.....	75
3.5.1.1 Feedback-based vs. reservation-based Congestion Control.....	75
3.5.1.2 Host-based vs. network-based Congestion Control	76
3.5.1.3 Window-based vs. rate-based Congestion Control	77
3.5.2 TCP Congestion control.....	78
3.5.2.1 Slow Start and Congestion Avoidance.....	78
3.5.2.2 Fast Retransmit.....	81
3.5.2.3 Fast Recovery	82
3.5.3 Explicit Congestion Notification	84
3.5.3.1 ECN at Routers	84
3.5.3.2 ECN at End Hosts	86
3.5.3.3 TCP Initialization	85
3.5.4 Non-TCP Unicast Congestion Control	87
3.5.4.1 TCP Friendly Rate Control	87
3.5.4.2 TCP Like Congestion Control.....	90
3.5.5 Multicast Congestion Control	90
3.5.5.1 Classification of Multicast Congestion Control.....	91
3.5.5.2 Requirements for Multicast Congestion Control	93
3.5.5.3 End-to-End Schemes.....	94
3.5.5.4 Router-Supported Schemes.....	95
3.6 Active Queue Management.....	96
3.6.1 Packet Drop Policies	97
3.6.1.1 Degree of Aggregation.....	97
3.6.1.2 Drop Position	98

3.6.1.3 Drop Priorities	99
3.6.1.4 Early or Overloaded Drop	99
3.6.2 Dec-Bit	100
3.6.3 Random Early Drop	101
3.6.3.1 Estimating Average Queue Length and Packet Drop Priority	102
3.6.3.2 Packet Drop Decision.....	103
3.6.4 Weighted Random Early Detection	104
3.7 Routing	106
3.7.1 Unicast Routing.....	108
3.7.1.1 Classification of Routing Protocols	108
3.7.1.2 Distance Vector Routing.....	109
3.7.1.3 Link State Routing	111
3.7.2 IP Multicast Routing	115
3.7.2.1 Multicast Addressing	117
3.7.2.2 Internet Group Management Protocol.....	118
3.7.2.3 Building the Multicast Distribution Trees	123
3.7.3 QoS Routing.....	127
3.7.3.1 QoS Routing Algorithms	128
3.7.3.2 Path Selection.....	129
3.7.3.3 Software Architecture of a QoS Routing Protocol.....	132
3.8 Admission Control	134
3.8.1 Basic Architecture of an Admission Control	134
3.8.2 Parameter-based Admission Control	135
3.8.3 Measurement-based Admission Control.....	138
3.8.4 Experience-based Admission Control.....	142
3.8.5 Probe-based Admission Control	142
3.9 Internet Signalling.....	144
3.9.1 Resource Reservation Protocol (RSVP)	145
3.9.1.1 Integrated Services	145

3.9.1.2 RSVP Architecture.....	147
3.9.1.3 RSVP Signalling Model.....	149
3.9.1.4 RSVP Messages	149
3.9.1.5 RSVP Transport Mechanism Issues.....	151
3.9.1.6 RSVP Performance	151
3.9.1.7 RSVP Security	151
3.9.1.8 RSVP Mobility Support.....	153
3.9.2 Next Step in Internet Signalling.....	153
3.9.2.1 Requirements for NSIS	154
3.9.2.2 NSIS Framework.....	155
3.9.2.3 NSIS Transport Layer Protocol.....	157
3.9.2.4 NSIS Signalling Layer Protocols	161
3.9.3 Signalling for Voice over IP	167
3.9.3.1 Architecture and Standard for Voice over IP.....	168
3.9.3.2 H.323	169
3.9.3.3 SIP	171
3.10 QoS Architectures	175
3.10.1 Integrated Services (IntServ)	175
3.10.1.1 IntServ Basic Architecture	175
3.10.1.2 IntServ Service Classes	178
3.10.1.3 IntServ Problems.....	179
3.10.2 Differentiated Services (DiffServ).....	179
3.10.2.1 DiffServ Architecture.....	180
3.10.2.2 DiffServ Routers and Protocol Mechanisms.....	181
3.10.2.3 DiffServ Service Groups.....	182
3.10.3 Multi Protocol Label Switching (MPLS).....	183
3.10.3.1 MPLS Architecture Concept.....	184
3.10.3.2 Label Distribution	186
3.10.3.3 MPLS Routers and Protocol Mechanisms	188

3.11 Mobility Support	189
3.11.1 Mobile IPv4.....	190
3.11.1.1 Architectural Overview	190
3.11.1.2 Agent Discovery.....	192
3.11.1.3 Registration	193
3.11.1.4 Tunnelling	196
3.11.1.5 Routing.....	197
3.11.2 Mobile IPv6.....	197
3.11.2.1 Architectural Overview	198
3.11.2.2 Protocol Design Aspect to Support Mobile IPv5.....	199
3.11.2.3 Movement Detection.....	200
3.11.2.4 Binding Update	201
3.12 Audio and Video Transport.....	202
3.12.1 Transport Protocols	202
3.12.1.1 Real Time Transport Protocol (RTP).....	203
3.12.1.2 Streaming Control Transmission Protocol (SCTP).....	206
3.12.1.3 Datagram Congestion Control Protocol (DCCP).....	212
3.12.2 Architectures	215
3.12.2.1 Voice over IP.....	215
3.12.2.2 Internet Protocol Television (IPTV)	216
3.13 Virtual Private Network	220
3.13.1 VPN Devices	221
3.13.2 Classifications of VPNs	221
3.13.2.1 Site-to-Site VPNs	221
3.13.2.2 Remote Access VPNs	223
3.13.2.3 Service Provider Provisioned Site-to-Site VPNs	224
3.13.3 Protocols to Enable VPNs.....	225
3.13.4 MPLS VPNs.....	227
3.13.4.1 MPLS Layer 2 VPNs	227

3.13.4.2 MPLS Layer 3 VPNs	228
3.13.5 Multicast VPN.....	229
3.14 Summary	232
4. Internet Protocol Suite	237
4.1 Introduction	237
4.2 Physical Layer	238
4.3 Data Link Layer	239
4.3.1 Data Link Layer's Services.....	240
4.3.2 Data Link Layer's Protocol Examples	243
4.3.2.1 Serial Line IP (SLIP).....	244
4.3.2.2 Point-to-Point Protocol (PPP)	244
4.3.2.3 Ethernet	246
4.3.3 Summary	249
4.4 Internet's Network Layer	250
4.4.1 Internet's Network Layer Services	250
4.4.2 Internet's Network Layer Protocols	252
4.4.3 The Internet Protocol IPv4.....	253
4.4.3.1 IPv4 Addressing.....	254
4.4.3.2 IPv4 Datagram Format.....	256
4.4.3.3 IPv4 Basic Mechanisms	257
4.4.3.4 IPv4 Input Processing	259
4.4.3.5 IPv4 Output Processing.....	260
4.4.3.6 IPv4 Packet Forwarding.....	261
4.4.4 The Internet Protocol IPv6	262
4.4.4.1 IPv4 Limitation	262
4.4.4.2 Pv6 Addressing	263
4.4.4.3 IPv6 Datagram Format.....	264
4.4.4.4 IPv6 Basic Mechanisms	265
4.4.5 Unicast Routing Protocols in Internet.....	266

4.4.5.1 Routing Information Protocol Version 1	266
4.4.5.2 Routing Information Protocol Version 2	269
4.4.5.3 Open Shortest Path First	270
4.4.5.4 Border Gateway Protocol.....	273
4.4.6 Multicast Routing Protocols in Internet.....	277
4.4.6.1 Distance Vector Multicast Routing Protocol	278
4.4.6.2 Multicast Extension to Open Shortest Path First	280
4.4.6.3 Protocol Independent Multicast	282
4.4.7 Summary	291
4.5 Transport Layer.....	292
4.5.1 Transport Layer Services	293
4.5.2 Transport Layer Protocols.....	296
4.5.2.1 User Datagram Protocol.....	297
4.5.2.1.1 UDP Segment Format	297
4.5.2.1.2 UDP Protocol Mechanisms.....	297
4.5.2.1.3 Application of the UDP.....	299
4.5.2.2 Transmission Control Protocol	299
4.5.2.2.1 TCP Segment Format.....	299
4.5.2.2.2 TCP Protocol Mechanisms.....	301
4.5.2.2.3 TCP Implementations	305
4.5.2.2.4 Application of the TCP	305
4.5.3 Summary	306
4.6 Application Layer	306
4.6.1 Application Layer Services.....	308
4.6.2 Selected Application Layer Protocols.....	311
4.6.2.1 Simple Mail Transfer Protocol.....	311
4.6.2.2 Simple Network Management Protocol.....	313
4.6.2.3 Hypertext Transfer Protocol.....	321
4.6.2.4 Real Time Transport Protocol.....	327

4.6.3 Summary	327
5. Next Generation Network and the IP Multimedia System	328
5.1 Introduction	328
5.2 Next Generation Network	329
5.2.1 NGN Architecture	330
5.2.2 NGN Functions	332
5.2.2.1 Transport Stratum Functions.....	332
5.2.2.2 Service Stratum Functions	334
5.2.2.3 Management Functions	336
5.2.2.4 End User Functions	337
5.3 IP Multimedia Subsystems.....	337
5.3.1 Introduction	337
5.3.2 IMS Functional architecture	341
5.3.2.1 The Call Session Control Function (CSCF).....	343
5.3.2.1.1 The Proxy-CSCF (P-CSCF).....	343
5.3.2.1.2 The Interrogating-CSCF (I-CSCF)	345
5.3.2.1.3 The Serving-CSCF (S-CSCF).....	346
5.3.2.1.4 The Emergency-CSCF (E-CSCF).....	346
5.3.2.2 The Home Subscriber Server (HSS)	347
5.3.2.3 The Subscription Location Function (SLF)	348
5.3.2.4 The Application Server (AS)	348
5.3.2.5 The Interconnection Border Control Function (IBCF)	349
5.3.2.6 The Media Resource Function (MRF)	349
5.3.2.7 The Breakout Gateway Control Function (BGCF).....	349
5.3.2.8 The Circuit-Switched Network Gateway	350
5.3.3 Fundamental IMS Mechanisms	350
5.3.3.1 IMS Addressing	350
5.3.3.1.1 Public User Identity	351
5.3.3.1.2 Private User Identity	351

5.3.3.1.3 Public Service Identity	352
5.3.3.1.4 Globally Routable User Agent.....	352
5.3.3.2 P-CSCF Discovery	353
5.3.3.3 IMS Session Control	354
5.3.3.3.1 Initial Registration.....	355
5.3.3.3.2 Basic Session Establishment.....	358
5.3.3.3.3 Basic Session Termination.....	365
5.3.3.3.4 Basic Session Modification	366
5.3.3.4 S-CSCF Assignment	366
5.3.3.5 AAA in the IMS	367
5.3.3.5.1 Authentication and Authorization.....	367
5.3.3.5.2 Accounting and Charging	368
5.3.4 IMS Services	371
5.3.4.1 Presence.....	371
5.3.4.2 Messaging	375
5.3.4.3 Push to Talk over Cellular	374
5.3.4.4 Multimedia Telephony	376
5.4 NGN and IMS Solutions	377
5.4.1 Session Border Control	377
5.4.2 Softswitch.....	378
5.4.3 Media Gateway	378
5.4.4 IMS Core	379
5.4.5 Subscriber Databases	379
5.4.6 Application Servers.....	379
5.5 Summary	379
6. References.....	380

1. Introduction

1.1 What is The Specific Feature of this Book?

The subject used for designing and developing computer networks is very complex, involving many mechanisms, different protocols, architectures and technologies. To deal with this complexity, authors of many computer network books used layers to describe the computer networks. Examples are OSI/ISO model with 7 layers and TCP/IP model with 5 layers. With a layered architecture, readers, such as students or computer specialists, learn about concepts and protocols in one layer as a part of this complex system, while seeing a big picture of how it all fits together [Kur-2001]. At each layer, the authors described the protocols, their mechanisms and architectures. Because a protocol can be used in several layers and a protocol mechanism can be used in distinct protocols at several layers and at numerous architectures, describing the fundamental protocols and protocol mechanisms before addressing the layered architecture will reduce the protocol complexity, and providing the readers a good overview about the protocol design through knocking the existing protocol mechanisms together.

Unlike the other computer network books, this book starts with a chapter about fundamental protocol mechanisms. Based on these protocol mechanisms, the layered architecture or the Internet protocol suite as a “bottom-up” principle and the Next Generation Network are then described. Thus, each protocol or protocol mechanism is only illustrated one time and the readers then have a depth overview, in which layer and in which protocol or architecture a given protocol mechanism can be used.

1.2 What are the Contributions of this Book?

The main contributions of this script are described in the following. We first provide a rather self-contained survey of techniques including mechanisms, architectures, protocols and services to control the traffic and to ensure the QoS for data and multimedia applications. Evaluation and analysis of these techniques in respect of layers, communication types, application and of QoS achievement are shown.

We then present a depth overview about the Internet protocol suite in respect of the layered architecture and on the basis of the mechanisms and protocols illustrated in the previous section. At each layer, selected protocols and technologies with used mechanisms are discussed. Finally, the next generation

network architecture, its fundamental mechanisms and the IMS (IP Multimedia Subsystem) are described.

The outline of this script is described as follows. Chapter 2 gives background information about computer networks and their design. Section 2.1 provides a brief description of the basis reference models for communication systems. The Multimedia Networking, Next Generation Networking and Mobile Networking as important drivers for the future of fixed-mobile convergence are presented in section 2.2. Consequences for network planning and the network planning considerations are discussed in section 2.3 and 2.4 respectively.

Chapter 3 provides a rather self-contained survey of techniques including architectures, mechanisms, protocols and services for controlling the traffic and guaranteeing QoS at several layers in multi-service computer networks. It starts with the mechanisms for detecting and correcting the packet level and bit level errors. Following it, section 3.2 represents the multiple access control mechanisms and protocols that allow sharing a single broadcast medium among competition users. Section 3.3 introduces the traffic access control mechanisms allowing the filtering of source traffic flows at the network entry and at the specific points within the network. Section 3.4 investigates packet scheduling mechanisms. Mechanisms for congestion control and avoidance at the transport layer and Internet layer are presented in section 3.5 and 3.6 respectively. Section 3.6 describes fundamental mechanisms for unicast and multicast routing and the Internet routing protocols. QoS routing is also investigated. The mechanisms and protocols for admission control and Internet signaling are illustrated in section 3.8 and 3.9. Section 3.10 summarizes the architectures and technologies developed for guarantee the QoS in Internet. Mobility support for both IPv4 and IPv6 are discussed in section 3.11. Section 3.12 gives a brief background on the new transport protocols developed for support end-to-end multimedia communications. Finally, Virtual Private Network (VPN) including MPLS VPNs and multicast VPNs is described in section 3.13. A summary of all protocol mechanisms discussed in chapter 3 is shown in section 3.14.

Chapter 4 represents a depth overview about the Internet protocol suite on the basic of the protocol mechanisms discussed in the chapter 4. The main goal of this chapter is to introduce the students how to design and develop new protocols on the basic of existing protocol mechanisms. It begins with a short introduction to the TCP/IP reference model covering 5 layers (physical, data link, network, transport and application) and its basic terminologies. The physical layer and its major protocol mechanisms are summarized section 4.2. Main services and selected protocols for the data link layer are discussed in section 4.3. Following this, the network layer services and protocols are illustrated in section 4.4. Transport layer services and transport layer protocols

are described in section 4.5. Chapter 4 ends with the application layer services and protocols.

Chapter 5 gives a survey about the next generation networks covering architectures, functions and the IP Multimedia Subsystem (IMS). The fundamental mechanisms illustrated in chapter 3 are also used in this chapter as a basic for describing the architectures, functions and IMS. Finally, conclusion and outlook are given in chapter 6.

2. Fundamentals of Computer Networks, the Internet and Next Generation Networks

Before embarking on an investigation of traffic management and quality of service (QoS) control together with their analysis and design, this chapter starts with a brief description of the basis reference models used for describing the communication systems. It then gives a selection of important applications driving the future of the Internet and the Next Generation Networks toward fixed mobile convergence. Finally, consequences and a review of significant aspects in the computer network planning.

2.1 Network Reference Models

Computer networks do not remain fixed at any single point of time. They must evolve to accommodate changes in the underlying technologies upon which they are based and changes in the service requirements placed on them by applications. Designing a network to meet these requirements is no small task. In order to help deal with this complexity, the OSI (Open Systems Connection) reference model and the TCP/IP reference model have been developed. These reference models define a common network architecture that guides the design and implementation of networks.

2.1.1 OSI Reference Model

The OSI reference model developed by the ISO (International Organization for Standardization) provides a fundamental theoretical model for partitioning the network functionality into seven layers, where the functionality assigned to a given layer is implemented in a set of protocols. Each layer offers certain services to the higher layers, shielding these layers from details of how the offered services are actually implemented [Tan-2003]. Between each pair of adjacent layers there is an interface that specifies which services the lower layer offers to the upper one. The OSI reference model is shown in figure 2-1.

The significant concepts defined in the OSI reference model are layers, protocols, interfaces and services. These concepts and the seven layers of the OSI reference model will be described in this section.

Layer

When the network system gets complex, the network designer introduces another level of the abstraction. The intent of an abstraction is to define a model

that unambiguously describes functions involved in data communication in a way, which allows the capturing of some important aspects of the system, providing an interface that can be manipulated by other components of the system, and hides the details of how a component is implemented from the users of this component.

Abstraction naturally leads to layering. The general idea of layers is to start with the services offered by the underlying hardware as the physical layer, and then add a sequence of layers, each providing a higher level of services. Each layer is responsible for a certain basis services. The services provided at a layer both depend and build on the services provided by the layer below it.

Dividing communication systems into layers has two main advantages. First, it decomposes the problem of designing a network into more manageable components. Instead of implementing one piece of network software that does every thinks, several layers can be implemented, each of which solves one part of the problem. Second, if the network designers decide to add new services, they only need to modify the functionality of the layers relating to these services, using again the functions provided at all the other layers.

Design issues for the layers include a set of mechanisms, for example identification of senders and receivers, error control, congestion control, routing and admission control etc. These mechanisms will be investigated in chapter 3.

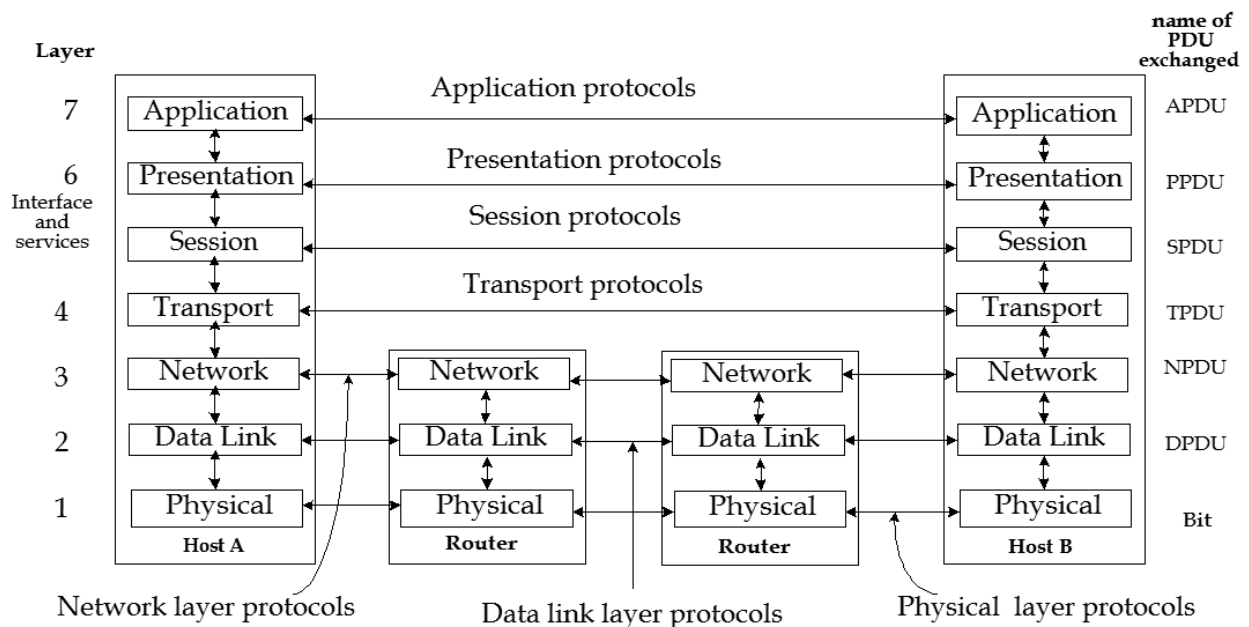


Figure 2-1: The OSI reference model

Protocols

Using the layering concept as a foundation basis, we now discuss the architecture of a network in more detail. Communication between entities at a given layer is performed via one or more protocols. Whereby, a layer-n protocol defines the rules and conventions used in the communication between the layer-n of one system to the layer-n of another system.

In particular, a layer-n protocol defines the message formats and the order of messages exchanged between the layer-n protocol instances of two or more systems, and the actions taken on the sending and receiving of messages or events.

Protocol Data Unit (PDU)

A protocol Data Unit (PDU) is a message unit (e.g. packet, datagram, segment) delivered through a layer of telecommunication systems. A Protocol Data Unit at layer N consists of a header and a payload part. While the header contains the control information (e.g source address, destination address) used for handling this PDU at the layer N, the payload part contains the headers of the upper layer protocols and the user data.

Interfaces and Services

The communication between entities at a given layer is invoked via the interface with the layer below. An interface defines a set of services the lower layer offers to the upper one.

Services can be classified into two classes: connection-oriented and connection-less services. When a connection-oriented service is used, the service user first establish a connection with its communication entity, uses this connection to delivery the data, and then teardowns the connection after finishing the data transfer. In contrast, the user of a connection-less service transmits data to its communication partner without the need of a connection establishment. Also, services can be categorized into reliable and unreliable services. Loosely speaking, reliable service guarantees that data transmitted from a sender to a receiver will be delivered to the receiver in order and in its entirety. Connectionless service does not make any guarantee about the data delivery.

A service is implemented via a set of service functions. Important service function are for example connection establishment, data transfer and connection teardown. A service function is formally specified by a set of service primitives

that are available to a user to access to this service. There are four classes of service primitives: Request, Indication, Response and Confirm [Tan-2003].

The Seven Layers

Starting at the bottom of the figure 2-1 and working up, the seven layers of the OSI reference model is summarized as follows.

- **Physical layer (layer 1):** The functions of the physical layer include all physical aspects used for communicating between two directly connected physical entities. Typically, these physical properties include electromechanical characteristics of the medium or link between the communicating physical entities such as connectors, voltages, transmission frequencies, etc.
- **Data link layer (layer 2):** the data link layer is responsible for getting the data across a link or across a physical medium. It accepts the raw bit stream provided by the physical layer and provides reliable transfer of data between to directly connected layer-2 entities.
- **Network layer (layer 3):** this layer defines necessary functions to support data communication between indirectly connected entities. It provides services for forwarding packets from a layer-3 entity to another via one or more networks until the final destination is reached. In order for routers to know how to forward packets, they must have some knowledge of network topology. This knowledge may be complete or partial, and is dynamically created and maintained via routing protocols. Thus, routing is a key service at the network layer. If too much traffic is present in a network at the same time, this network may get congested. The control of such congestion is also a service provided at the network layer.
- **Transport layer (layer 4):** The purpose of the transport layer is to provide transparent transfer of data between end users. The perspective of layer 4 is of end-to-end communications rather than the hop-by-hop perspective of layer 3. Layer 4 assumes that packets can be moved from network entities to network entities, eventually getting to the final destination host. How this is accomplished is of no concern to Layer 4 functionality.
- **Session layer (layer 5):** This layer provides mechanisms for structuring and managing the interaction between end user application processes. It provides for either duplex or half-duplex operation and establishes checkpointing, termination, and restart procedures.
- **Presentation layer (layer 6):** The presentation layer is concerned with the presentation of user or system data. It presents the data into a uniform

format and masks the difference of data format between two dissimilar systems. It also translates the data from application to the network format. The presentation layer is also responsible for the protocol conversion, encryption, decryption and data compression.

- **Application layer (layer 7):** The application layer defines the interfaces for communication and data transfer. At this layer, communication partners are identified, quality of service is addressed, user authentication and privacy are considered, and any constraints on data syntax are classified.

2.1.2 The TCP/IP Reference Model

The Internet is based on the TCP/IP reference model, which is the successor of the OSI reference model described above. This reference model differs from its predecessor by layer functionalities. The TCP/IP model does not exactly match the OSI model. There is no universal agreement regarding how to describe TCP/IP with a layered model but it is generally agreed that there are fewer levels than the seven layers of the OSI model. Most descriptions present from four to five layers. In this section, TCP/IP reference model is described with five layers shown in figure 2-2.

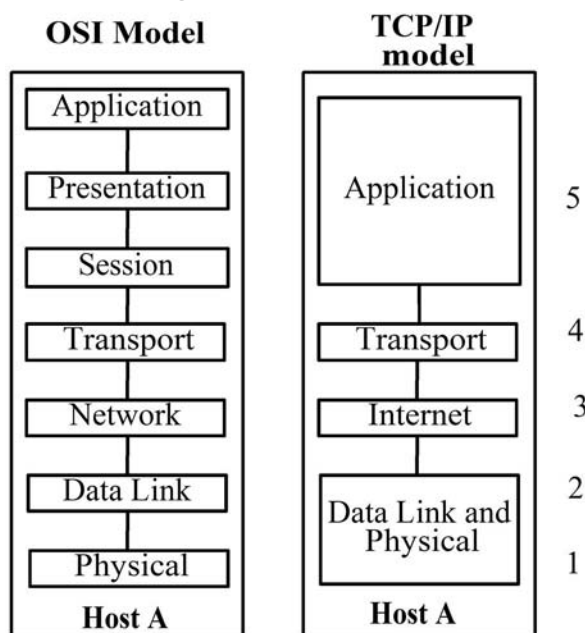


Figure 2-2: The TCP/IP reference model

The TCP/IP protocol stack made up of four layers is shown in figure 2-3. With the IETF public Request for Comments (RFC) policy of improving and updating the protocol stack, TCP/IP protocol model has established itself as the protocol suite of choice for most data communication networks.

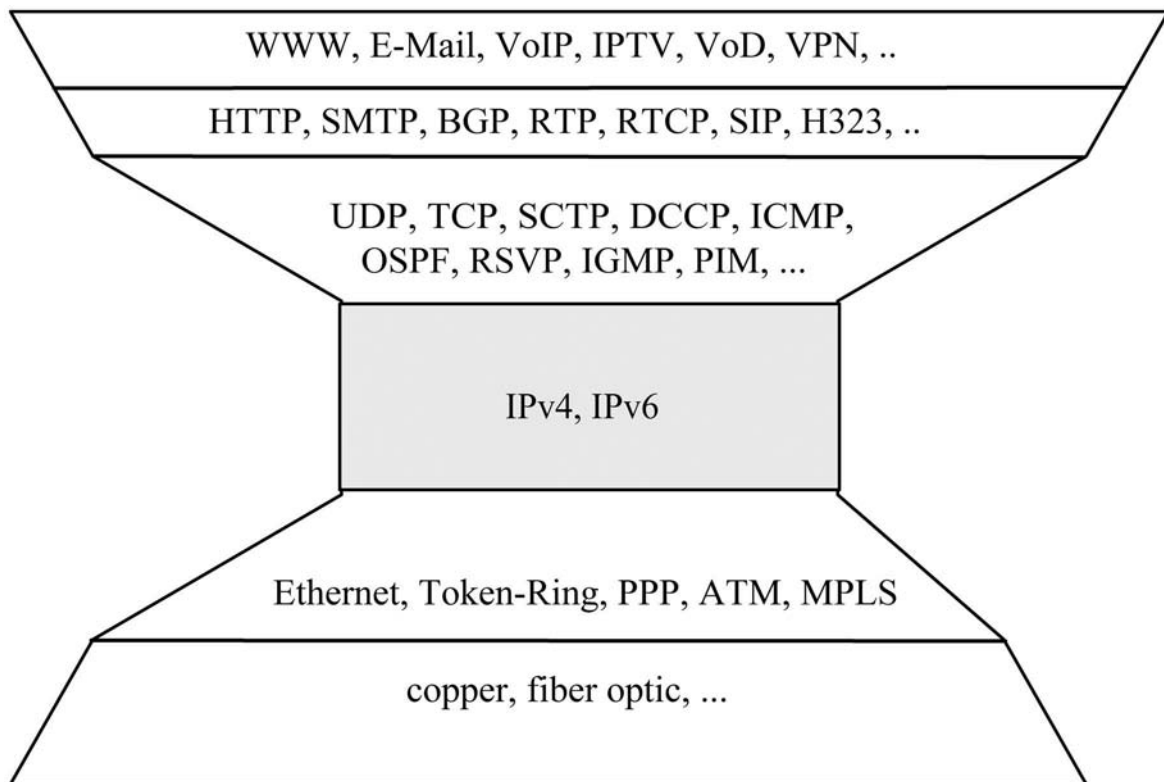


Figure 2-3: Protocol examples in the TCP/IP protocol stack

The layers of the TCP/IP model are:

- Data link and physical layer:** In TCP/IP model, the data link layer and physical layer are generally grouped together. The data link layer offers services to delivery data between network entities, as well as to detect and possibly correct errors that may occur in the physical layer. Important protocol mechanisms of this layer are the medium access control, framing, addressing, checksum, and error control. The data link layer protocols are for example Ethernet, Token-Ring, FDDI and X.25. The characteristics of the hardware that carries the communication signal are typically defined by the physical layer. Examples of physical layer standards are RS-232C, V.35 and IEEE 802.3
- Internet layer:** This layer provides functions to enable logical communication between end hosts. The internet layer protocol instance accepts request to send data from the transport layer, converts the transport data to IP packet format, and sends them down to the data link layer for further processing. Services provided at the Internet layer are for example addressing, segmentation/reassemble, routing, congestion control, buffer management, switching, and admission control. Important

protocols at the Internet layer are IPv4, IPv6, ICMP, ICMP, ARP, packet processing mechanisms and the routing protocol OSPF.

- **Transport layer:** This layer provides services that enable logical communication between application processes running on different end hosts. Examples of services provided at the transport layer are multiplexing, demultiplexing, connection management, congestion control and flow control. Two well known protocols at the transport layer are TCP and UDP. Each of these protocols provides a different set of transport layer services to the involving applications.
- **Application layer:** The application layer provides the services which directly support an application running on a host. It contains all higher-level protocols, such as FTP, HTTP, SMTP, DNS and Telnet etc.

2.2 Fixed-Mobile Convergence

Today, multimedia applications are becoming more popular, but they put additional problems for computer networks. The problems associated with multimedia communications include coding of multimedia data, transporting this data from one end to another end, and achieving the required QoS. To solve these problems, computer networks must be able to offer not only the traditional best-effort service but also services for enabling the multimedia communication so that they can transport combined data, voice and video traffic. Such computer networks are called multi-service computer networks. Since multimedia applications (such as VoIP and video on demand) require a certain QoS from the network sites, these networks should evolve to provide QoS guarantee to the users.

In order to facilitate multi-service networks, several technologies have been developed in the last years. Typical technologies are ATM, MPLS, multicast, VPN, VoIP, IPTV, IntServ and DiffServ. Together with these technologies, new mechanisms and protocols for managing and controlling the QoS in multi-service network have been developed.

In the following sections, we first present a selection of important drivers that mainly influence the development and the use of multi-service computer networks.

2.2.1 Multimedia Networking over Internet

Computer networks were originally designed to carry the data only. Today, they are increasingly being used for multimedia applications. The reason for this development is low cost for operators on the high performance IP technology and low prices for consumers.

But, providing an unreliable data transmission and operating as datagram switching, IP networks are not naturally suitable for real-time traffic. Thus, to run multimedia applications over IP networks, several issues must be solved.

Problem Issues

Firstly, in comparison with traditional data applications, some multimedia applications require much higher bandwidth. A single video stream consumes between 1.6 Megabits/s [Mbps] und 12 Mbps depending on the encoding method and whether the stream is standard definition or high definition. Thus the hardware devices have to provide enough buffer bandwidth. But, for most multimedia applications, the receiver has a limited buffer. If no measure is taken to smooth the data stream when data arrives too fast, the buffer will overflow and some data packets will be lost, resulting in bad quality. When data arrives too slowly, the buffer will underflow and the application will starve.

Second, most multimedia applications require the transfer of real-time traffic that must be played back continuously at the rate they are sampled. If the data does not arrive in time, it will be dropped later at the end systems. Some new transport protocols must be used to take care of the timing issues so that audio and video data can be played back continuously with correct timing and synchronization.

Third, there are a lot of multimedia applications that require guaranteed bandwidth when the transmission takes place. So there must be some mechanisms for real-time applications to reserve resources along the transmission path.

Fourth, in addition to the delay, network congestions also have a lot of effects on the quality of the real-time traffic. Packet losses most often occur due to congestion in the routers; more and more packets are dropped at the routers when congestion increases. While the packet loss is one of thinks that make TCP efficient and fair for non real-time applications, the effect of packet losses is a major issue for real-time applications using RTP over UDP and do not support congestion control. This is because the UDP does not have any reaction on packet losses. The transport protocols designed for multimedia applications must take into account the congestion control in order to reduce the packet loss.

Fifth, various multimedia applications are related to the multicast. For example, in video conference, the video data needs to be sent to all participants at the same time. Or in Internet protocol television, a TV channel needs to be sent to all receivers of this channel at the same time.

Solutions

The Internet as multi-service networks carries all type of traffic (e.g. data, video, voice); each of them has different traffic characteristics and QoS requirements. If enough bandwidth is available, the best-effort service fulfils all of these requirements. But when resources are inadequate, however, real-time traffic will suffer from the congestion.

The solution for multimedia networking at the Internet layer is to prioritize all traffic and to provide the service differentiation and QoS for all of this traffic. Technologies developed for this are first of all IPv6, MPLS, DiffServ, IntServ, RSVP, IP multicasting, VPNs, and mechanisms for regulating the traffic and controlling the QoS for these multimedia applications [Hag-2006, Arm-2000, Sna-2005]. Moreover, multicast services need to be taken into consideration in order to reduce the traffic and thus the bandwidth consumption. Thus, IP multicast protocols are specified. Examples are IGMP, PIM (PIM-SSM, PIM-SM, PIM-DM) and DVMRP [FHH-2006].

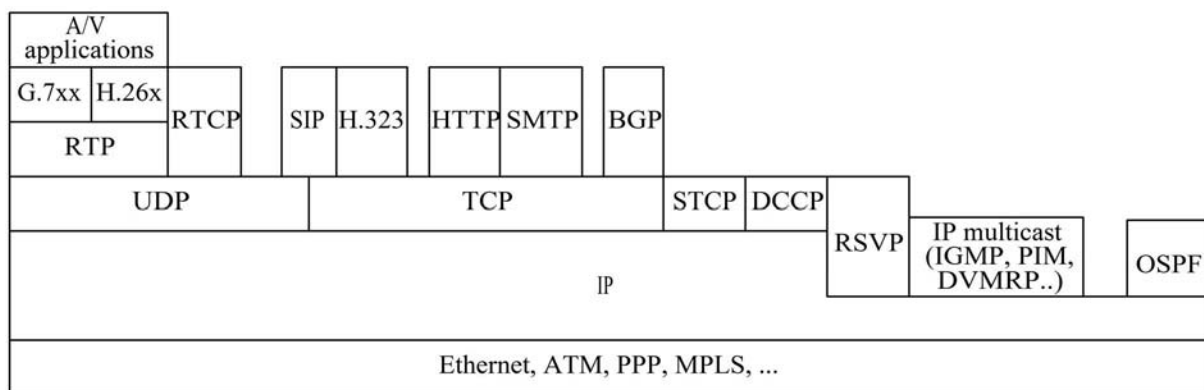


Figure 2-4: Protocols for multimedia communications

In order to provide timing, synchronization and congestion control for multimedia applications, new transport protocols are added into the transport layer. They are RTP, SCTP and DCCP [SCF-2003, CB-2006, KHF-2006]. In comparison with the services provided by TCP and UDP, these new protocols additionally offer several new services such as time reconstruction, loss detection, multi-homing, multi-streaming and congestion control, in respect of multimedia applications. Furthermore, new congestion control mechanisms for multimedia applications and new reliable multicast congestion control protocols are also developed.

At the application layer, services need to be added to compress the multimedia data before sending them over a compute network. This compression will reduce the bandwidth needed for this multimedia data since

multimedia applications require very high bandwidth. Since the best-effort Internet architecture does not provide service to multimedia applications, to support voice transfer over the Internet, two major architectures have been specified. The ITU-T has created H.323 that provides a framework for real-time service in an IP environment [DBP-2006]. The other one is the Session Initiation Protocol (SIP) [RSC-2002; SJ-2006] developed by IETF. SIP is an application-layer signaling protocol for creating, modifying, and terminating multimedia sessions such as the Internet telephony call.

An example of a TCP/IP protocol stack including protocols specified for multimedia communications over the Internet is depicted in the figure 2-4. Details about the protocols and mechanisms for supporting multimedia networking will be described in chapter 3.

2.2.2 Next Generation Networks

A Next Generation Network (NGN) is a packet-based network that enables on the one hand the deployment of access independent services over converged fixed and mobile networks, and on the other hand the use of multiple broadband and QoS-enabled transport technologies in which service-related functions are independent from underlying transport-related technologies [TR-180.000]. NGN is one of four current solutions (GAN – cellular integration; 3GPP – WLAN internetworking; Femtocells; NGNs) for the Fixed Mobile Convergence (FMC), which is the convergence technology offering a way to connect a mobile phone to a fixed line infrastructure so that operators can provide services to their users irrespective of their location, access technology and end terminal.

Next Generation Networks are based on Internet technologies including Internet Protocol (IP) and Multiprotocol Label Switching (MPLS) as the transport technology, and Session Initiation Protocol (SIP) at the application layer. Based on these technologies, NGNs allow the transport of various types of traffic (voice, video, data and signalling). Triple play services (Voice, Internet and TV) are available via Cable and xDSL already. The NGN brings mobility in to the picture and the opportunity for further bundling of high revenue services for customers.

At the core of a NGN is the IP Multimedia Subsystem (IMS), which is defined by 3GPP and 3GPP2 standards and organisations and based on Session Initiation Protocol (SIP). IMS is a framework consisting of a set of specifications that describe the NGN architecture for implementing Voice over IP (VoIP) and multimedia services. The IMS standard defines architecture and concepts that enables the convergence of data, voice, video, fixed network technologies and mobile network technologies over an IP based infrastructure.

IMS provides an access independent platform for any type of access technologies such as a fixed line, CDMA, WCDMA, GSM/EDGE/UMTS, 3G, WIFI or WiMax. IMS allows features such as Presence, IPTV, Messaging, and Conferencing to be delivered irrespective of the network in use. IMS is anticipated that we are moving into an era where rather than having separate networks providing us with overlapping services, it is the relationship between the user and service that is important and the infrastructure will maintain and manage this relationship regardless of technology. The most obvious overlap currently is between fixed and mobile networks, and the IMS has been identified as a platform for the FMC technology.

Chapter 5 will describe the next generation network architecture, its fundamental mechanisms and the IMS as the core of each NGN and the main platform for the fixed mobile convergence.

2.2.3 Mobile Networks

With the proliferation of mobile computing devices and wireless networking products that demand for accessing to the Internet to get information and services at any time and any where, there is a strong need for the Internet infrastructure to provide mobile devices to connect to the Internet while roaming, preferably without interruption and degradation of communication quality. Mobility support for the Internet refers to the ability to keep active communication of an IP-based device continues during changing of its topological point of attachment to different networks.

Since the Internet was originally designed for communications between fixed nodes, it does not well consider the host mobility problem. The main limitations of the traditional TCP/IP protocol suits for mobility support include the following:

- Limitation of IP addresses: In mobile scenarios, the IP address of a mobile device has to be changed to indicate the change of its point of attachment to the network. But in traditional TCP/IP model, this address change makes other devices impossible to contact with this mobile device, since other devices only know the fixed IP address of the mobile device.
- Limitation of congestion controls at the transport layer: Transport layer protocols use the services provided by the network layer for congestion control. These protocols do not have any mechanisms to discovery the wireless link properties. Thus the congestion control at the transport layer does not distinguish the packet loss caused by wireless link from the normal packet loss in the wired network because of the congestion. It

recognizes the packet loss by wireless link as a congestion, which degrades the transport performance.

- Limitation of applications: Many applications are based on the traditional TCP/IP model and do not support their use in mobile environments. An example is the DNS. Its statically binding a domain name to a host IP address will be invalid because of the dynamic change of IP addresses of mobile devices.

In order to provide the mobility, functional requirements and performance requirements for mobility support in the Internet must be met [LFH-2006]. Functional requirements refer to mechanisms for handover management, location management, multi-homing and security. The performance requirements for mobile environments are specified via a set of performance metrics including handover latency, packet loss, signaling overhead and throughput.

To address these problems, various solutions have been developed that extend the TCP/IP model at several layers to support mobile networking. Some selected approaches will be investigated in the following paragraphs.

Mobility Support in the Network Layer

Mobile IPv4 (MIP4) and mobile IPv6 (MIP6) represent mobility support solutions in the network layer [Per-2002; Mail-2007; JPA-2004; Koo-2007]. MIP4 introduces the address assignment concept that enables a mobile node to get a permanent home network IP address and a foreign network IP address. In order to relay packet between correspondence node (CN) and mobile node (MN), MIP4 additionally defines two new components, the home agent (HA) and the foreign agent (FA). MIP6 resolves the triangle routing problem of MIP4 through a direct communication between the mobile node and the home agent, no foreign agent is needed in MIP6.

Mobility Support in the Transport Layer

In the transport layer, a wide range of investigations has been made to provide mobility support. A lot of solutions for performance improvement and mobility enhancement of TCP has been developed over past years [BB-1995, YB-1994, BKG-2001, BPS-1996, HA-1997, FYT-1997, TXA-2005]. The concept proposed in [BB-1995, YB-1994] is to split a TCP connection between a fixed host and MN into two connections: between the fixed host and the so called mobile support station (MSS) and between MSS and MN. While the first connection is handled by normal TCP, the second connection is optimized for the wireless link.

The authors in [BKG-2001] developed the so called Performance Enhancing Proxy (PEP) network agents that break the end-to-end TCP connection into multiple connections and use different parameters to transfer the data. PEPs are used to improve degraded TCP performance caused by characteristics of specific link environments, for example, in satellite, wireless WAN and wireless LAN environments.

The authors in [FYT-1997] developed the so called TCT Redirection (TCP-R) that keeps connection actively via revising the pair of addresses in the outgoing TCP connections when the IP address associated to the TCP connection is changed by TCP redirection options.

For new transport protocols SCTP and DCCP, mobility support has been proposed [RT-2007; EQP-2006; Koh-2005]. An extension of SCTP to support mobility is proposed in [RT-2007] and called MSCTP. In MSCTP, a MN initiates an SCTP association with the CN by negotiating a list of IP addresses. One of these addresses is selected as the primary address for normal transmission, the other addresses are defined as active IP addresses. When reaching a new network and obtaining a new IP address, MN informs its CN of the new IP address via sending the Address Configuration Change (ASCONF) chunk to CN. On receiving of the ASCONF, CN adds the new IP address to the list of association addresses and reply to MN. While moving, the MN changes the primary path to the new IP address obtained for the new subnet. Thus, the SCTP association can continue to transmit the data while moving to a new network.

Extension of DCCP for supporting mobility is proposed in [Koh-2005]. There are three new features need to be added to DCCP: DCCP-Move packet, two new DCCP packets of mobility capable features, and mobility ID feature.

In order to inform CN that the MN would like to enable to change its address during connection, the MN first sends a Change L option of Mobility Capable feature. On receiving this message, CN sends a Change R option to confirm MN. In response to the Change R option message, MN sends to CN a value of mobility ID feature that will be used to identify the connection. CN replies MN by sending Conform L option. When MN reaches a new network and obtains the new IP address, it informs CN by sending a DCCP-Move packet containing mobility ID value that was chosen for connection identification. On receiving DCCP-Move packet, CN sends DCCP-Sync message to MN, and changes its connection state and begins using new address of MN.

Now we have investigated several solutions for extending the TCP/IP protocol stack for mobility support. It is clear to see that the IP and the transport protocols are considered as key technologies, since their adoptions are expected to create substantial synergies.

2.3 Consequences for Network Planning

New applications and services will change the nature of traffic in future computer networks, having an effect on the amount of traffic and its characteristics. Furthermore, multimedia and mobile applications require QoS-enabling technologies. The significances for network planning and analysis are outlined in this section.

2.3.1 Traffic Demand Characterization

The new applications have different effects on the traffic characteristics. Stream applications, such as video on demand and IPTV, generate highly asymmetric traffic stream with the majority of the data flowing from the server to the client. The amount of traffic depends on the coding scheme, the preferred bit rate, which can be set by the user, as well as the duration of the streaming session. The interactive applications such as telephony and video conferencing typically establish bi-directional sessions between two or more hosts. This results in symmetric traffic flows between end systems.

In comparison with traffic generated by streaming applications and interactive real-time application, the web traffic is sent as small request into one direction that followed by large data transfers into the opposite direction. The characteristics of traffic generated from new applications are different from traffic generated by traditional data applications. These traffics differ in their call-level, packet-level and buffer-level through various traffic variables such as traffic distribution, arrival time, service time, packet size and the scheduling used to serve them. The behavior of traffic variables specified at the packet levels depends on the flow control and congestion control. Web applications use TCP as transport protocol. Thus, TCP flow control and congestion control parameters mainly affect the characteristic of web traffic. But multimedia applications do not use TCP as their transport protocol. They use RTP/UDP with additionally rate-based or TCP-friendly congestion control. Thus the traffic characteristics of these multimedia applications at the call-level and packet-level are big different from the web traffic.

The characteristics of traffic by various applications and at different level need to be considered during network planning process. Especially, for network dimension these characteristics can be exploited for cost savings and for QoS achieving.

2.3.2 Quality of Service Requirements

Originally, the TCP/IP protocol suite was developed to support a single class of best-effort service without guarantee of data delivery and quality of service. The history of this TCP/IP protocol suite shows a clear focus on developing a technology that seeks out and establishes connectivity through sites and end systems. Given knowledge of a packet's ultimate destination, the network will (if at all possible) find a path through any available links that enables the packet's ultimate delivery. The actual time it takes to achieve delivery is at best a secondary consideration. If no path is available, because of either long-term or short-term problems within the network, packets may be discarded. If network experiences the congestion, some packets may also be dropped by routers. If guaranteed delivery is required, the source and destination must utilize additional end-to-end mechanisms, for example the transport protocol TCP, to determine whether their packets are being delivered successfully and, retransmit lost packets if they are not. On the way to destination, all traffic flows share the same resources (bandwidth, buffer space) and receive similar quality of service.

The Need for QoS

Thus, the traditional TCP/IP network mainly focuses on where to send packets and not on the when to send packets as well as not on which packets should be sent first. This has never a problem as long as most applications were data-based and therefore had similar traffic characteristics and QoS requirements. However, in the real world importance is attached to the multimedia and interactive applications, such as chat sessions, audio streaming, video streaming, Voice over IP (VoIP) and Internet Protocol television (IPTV). These multimedia applications generating traffic across an IP network have their own requirements to meet. In particular, these applications are typical less elastic and less tolerant of delay variation and packet loss than data applications. Such applications require some guarantees of quality of service from the network, e.g. a maximum packet delay or a minimal bandwidth.

To provide QoS requirements for multimedia and interactive applications, TCP/IP services must be supplemented with some added features to the nature that can differentiate traffic and provide different service levels for different users and applications.

What is QoS?

Quality of Service is the ability of a network element (application, host, router, and switch) to have some level of assurance that its traffic and service

requirements can be satisfied. To achieve QoS, cooperation of all network layers from top-to-bottom and of every network elements from end-to-end is required.

There are four different viewpoints of QoS: customer's QoS requirements, QoS offered by service provider, QoS achieved by service provider, and QoS perceived by customer [Dvo-2001]. Customer's QoS parameters are focused on user perceived effects, and do not depend on the network design. These parameters might be assured to the user by the service providers through a contract.

QoS offered by service providers is a statement of the level of quality expected to be offered to the customer by the service provider for Service Level Agreement (SLA). Whereby, each service would have its own set of QoS parameters. QoS achieved by the service provider is a statement of the level of quality actually achieved and delivered to the customer. It is expressed by values assigned to QoS parameters. Based on the customer's QoS requirements, QoS offered and achieved by the service provider will be different from the QoS perceived by the customer.

There is more than one level of criteria to satisfy the different types of traffic (e.g. Voice, video, Internet television, interactive game, chat). The important parameters needed to describe the QoS requirements of these traffics are:

- End-to-end delay indicates the time taken to send a packet from the sender to the receiver. The end-to-end delay is composed of propagation delay, transmission delay, queuing delay and protocol delay.
- Jitter is the variable of end-to-end delay between arrivals of packets at the receiver.
- Throughput is the observed rate at which data is sent through a channel.
- Packet loss rate is the ratio of lost packets and the total packets transmitted
- System-level data rate indicated the bandwidth required, in bits per second.
- Application-level data rate indicates the bandwidth required, in application-specific units such as video frame rate
- Reliability is the percentage of network availability depending upon the various environmental.

In the last years, several fundamental mechanisms [Kes-2001] (e.g. new scheduling disciplines, new congestion controls, new admission controls and new signalling mechanisms) and protocols have been proposed - offering multi-level of services and provisioning QoS for multimedia applications. Moreover, various architectures and technologies (e.g. IntServ, DiffServ, MPLS, VPN) [Hus-2002; San-2006] have been developed that incorporate fundamental

QoS mechanisms within one architecture so that comprehensive QoS-enable networks can be achieved.

These architectures, QoS mechanisms and protocols as well as QoS parameters are necessary but insufficient to provide any service guarantee without considering them within the network planning process. They determine the constraints and objective of network planning and optimisation problems.

2.4 Network Planning Considerations

In order to design a perfect computer network, two important aspects must be considered. They are applications and the network infrastructure. These aspects will be investigated in this paragraph.

2.4.1 Application Considerations

As presented in section 2.1, the highest layer of TCP/IP model is the application layer referring to applications and services they require. Services provided by networks to the applications and the resources required by applications must be taken into consideration when designing the computer networks. In respect of applications, there are a set of issues that must be investigated for the network design.

Bandwidth requirement

Different applications require varying amounts of network bandwidths. For example, a single email application via SMTP does not have the same bandwidth requirement as a video demand application. Bandwidth sensitive applications, such as Internet telephony, require a given amount of bandwidth so that they are able to transmit data at a certain rate to be effective. But elastic applications, such as web transfer or electronic mail, can make use of as much or as little bandwidth as happen to be available.

It is therefore obvious that the bandwidth requirements of applications a network will need to provide, determine link capacities and the node types of the network you will finally design. Thus considering the bandwidth requirements for different types of applications are necessary needed during each network planning process.

Protocol requirement

The TCP/IP application layer supports various application protocols. Choosing an application protocol for a network application directly indicates the selection

of a transport protocol (e.g. TCP, UDP, RTP, SCTP, DCCP). Since TCP and SCTP provide the reliable connection-oriented service and congestion control, and UDP does not support this, the bandwidth requirement for applications using TCP (or SCTP) differs from bandwidth requirement for application using UDP. Moreover, there are applications that require multicast at the network layer. Therefore the routing and the bandwidth requirement for these multicast applications differ from those of the unicast applications. Thus, protocols used by the network applications also need to be considered in the network planning process.

Quality of Service and Type of Service (QoS and ToS)

The reason to consider QoS and ToS is that some user's data is more important than others. Thus there is a need to handle them with different services, for example premium service, controlled load service and best-effort service [San-2002].

The requirement for QoS and ToS has implications for the network planning. For example, routers and switches have to ensure the premium delivery of the traffic for a Voice over IP so as to support the QoS/ToS requirements of this application.

Multicast Communication

Multicast has been proven to be a good way for saving the network bandwidth. It is a main component of Internet Protocol TIVI (IPTV). Thus, multicast service must be taken into consideration while planning the network that supports IPTV or other multicast applications.

2.4.2 Infrastructure Considerations

Network applications running at the end systems need a transport mechanism to transmit user data and control information between them. The transport mechanism is provided by the underlying network infrastructure.

The network infrastructure is an important component in computer network planning. It grows as business expands. Moreover, it not only provides the delivery of user data, but it is also able to adapt to network changes.

In order to build a network infrastructure, several layer of the TCP/IP model must be taken into consideration. Moreover, there are various technologies available for building a network. The design of the Internet Protocol IP over different protocols depends on a set of mechanisms:

- **Encapsulation and overhead:** Because each data link layer protocol has its own frame format and its own transfer mechanisms, the encapsulation of the IP packets into the data link layer frame and the resulting overhead should be evaluated for the network planning purpose.
- **Routing:** Routing is needed to determine the path a packet should follow to reach its final destination. Therefore, selecting a routing protocol to be used for a service will affect the network infrastructure that need to be designed. Thus, routing consideration is very important for the network planning
- **Maximum Transmission Unit (MTU):** Different data link layers have different MTU sizes. The MTU size has an impact on total number of IP packets generated to transmit a piece of user data. Therefore, it has influences on the capacity consumption of the links and nodes of the network infrastructure. Because of this, MTU need to be considered in the IP network design over different data link layer protocols.

Design a network infrastructure involves several decision making processes that take technologies used for the infrastructure (e.g. Ethernet, ATM, and IP/MPLS), the equipments required, the cost for devices and protocols required into consideration.

3. Traffic Management and Quality of Service Control

Protocols are needed for controlling the sending and receiving of messages within the Internet. A protocol may consist of one or several protocol mechanisms. Each of these protocol mechanisms is a method describing a complex sub-function of a protocol. It can be implemented in various communication systems, in different layers and in several protocols. For example, The Internet checksum is a protocol mechanism implemented in TCP, UDP, IP, OSPF, Ethernet, etc. and in different layers of the TCP/IP protocol stacks. In order to develop a new protocol or architecture, it is significantly to have an overview of fundamental protocols and mechanisms. The fundamental mechanisms for traffic management and QoS control will be described in this chapter.

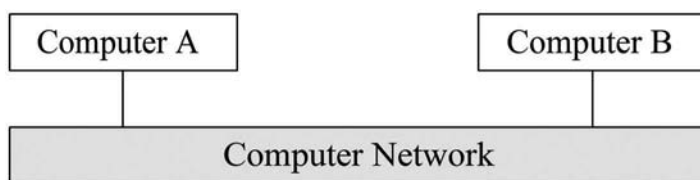


Figure 3-1: Basic scenario for data communication over the Internet

Supposed that the computer A and B are directly connected via a computer network and will exchange data through this network (figure 3-1). During the data transmission between A and B, transmission errors such as delay, loss, duplication and out-of-date of messages may occur. In order to eliminate these errors, a lot of questions must be answered, for examples:

- What is the reason for the errors? How should these errors be recognized and recovered? The answer of these questions deals with the protocol mechanisms for error detection and correction.
- How should senders, receivers and intermediate routers react to overload situations so that the packet losses will be minimal? The solutions for this question deal with the protocols and mechanisms for flow control and congestion control.
- How should senders, receivers and intermediate routers prevent the overload so that the congestion will not arise in the near future? The answer to this question addresses the mechanisms for congestion avoidance and resource reservation.
- How does a network choose a path between two nodes? What if the user wants to choose a path that has least delay, or least cost, or the most

available capacity? How can we send the same data to a group of receivers? The answer to this question addresses multicast routing protocols.

This chapter deals with fundamental mechanisms, protocols and architectures for traffic management and QoS control in Internet.

3.1 Error Control

Communication errors may occur at both bit-level and packet-level. Bit-level errors occur because of the corruption of the bits during the transmission, such as the inversion of an 0 bit to an 1 bit or an 1 bit to a 0 bit. The main reason for this error type relies on the transmission channel, which is not optimal because of noises, loss of synchronization, and of hand-off and fading. For example, the receiver has received the signal of 3 volt although the 0 volt signal was sent. The packet-level errors arise because of the corruption of the PDU (protocol data unit), e.g. packet loss, duplication or reordering of PDUs. Furthermore, the detected but uncorrected bit-level errors are treated as packet-level errors. There are several reasons for packet-level errors, e.g. overload in routers and at the end hosts, too early retransmission of packets, failure of nodes and/or transmission links, etc.

Communication errors discussed above may arise in all layers of computer networks, and, thus for a reliable data transmission, mechanisms for detection and correction of such errors are necessary needed. These mechanisms will add significant complexity to the protocols so that it can provide reliable service if this service is not already offered from the layers below. Communication protocol with an assumption of error-free transmission is very simple to implement but does not have any practical application.

In the next following sub-sections, fundamental mechanisms for detecting and correcting the bit-level and packet-level errors will be described.

3.1.1 Bit-level Error Control

The main principle of the bit-level error control is to add redundancy bits (called error detection code EDC) to the transmitted data at the sender so that the receiver can detect and/or correct the arrived data by using of this redundancy. Mechanisms for bit-level error control can be classified into two classes: bit-level error detection mechanisms, and, bit-level error detection and correction mechanisms. Error detection is done by having the sender only to set enough error-detection bits in the PDU to allow the receiver to deduce that an error occurred. The error correction is similar to error detection, except that a

receiver cannot only detect whether errors have been introduced in the frame but can also determine exactly where in the frame the errors have occurred and hence correct these errors [Kur-2004].

The basis schema for bit error detection is shown in figure 3-2. Supposed, that a datagram of d bits should be sent to a receiver. The sender first adds the error detection code (EDC) to d data bits and transmits the $(D+EDC)$ together to the receiver through a bit-error prone link. When the datagram D' arrives at the destination, the receiver computes the new error detection code EDC' for the incoming datagram and compares with the EDC from the sender to detect the error.

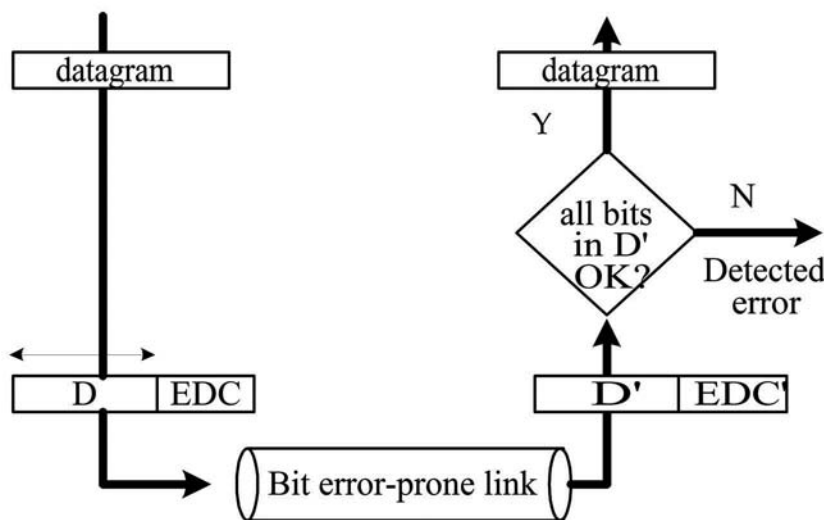


Figure 3-2: Bit error detection schema [Kur-2004]

There are several mechanisms for the bit error detection and correction. Fundamental well-known mechanisms used in Internet are for example parity check, Internet checksum, cyclic redundancy check and forward error correction (FEC) [Kur-2004, Tan-2002, LD-2003].

- **Parity check.** The basic idea of parity check is that the sender includes one addition bit to the data and set its value equal to 1 if the total number of 1s in the $d+1$ bits (d data bits plus a parity bit) is even. The sender then sends these $d+1$ bits to the destination. If these bits arrive at the receiver, the receiver counts the number of 1s. If an odd number of 1-valued bits are found with an even parity bit, the receiver knows that at least one bit error has occurred.
- **Internet checksum.** The d bits of data in figure 3-2 are treated as a sequence of 16-bit integers. The concept of Internet checksum is to sum these 16-bit integers and uses the resulting sum as the error detection bits. The sender sends the data together with the calculated Internet checksum.

If the data packet arrives at the receiver, the receiver again calculates the checksum over the received data and checks whether it is equal to the checksum carried in the received data. If it does not match, the receiver recognizes that there are bit errors in the data packet. Internet checksum is implemented in several TCP/IP protocols, for examples TCP, UDP, IPv4, OSPF routing protocol, Ethernet etc.

- **Cyclic redundancy checks (CRC).** CRC is based upon treating bit strings as representations of polynomials with coefficients of 0 and 1 only. A k -bit frame is regarded as the coefficient list for a polynomial with k terms, ranging from x^{k-1} to x^0 . The sender and receiver must agree a generator polynomial $G(x)$ in advance. For given d data bits D , the sender will choose r addition bits, EDC, and append them at the end of D in such a way that the polynomial represented by $d+r$ bit pattern is exactly divisible by $G(x)$ by using the modulo 2 arithmetic. The sender then sends this $d+r$ bits to the destination. When this data arrives at the receiver, the receiver divides the $d+r$ bits by $G(x)$. If the remainder is nonzero, the receiver knows that a bit error has occurred; otherwise the data is accepted as being correct.
- **Forward Error Correction (FEC).** FEC enables the receiver to detect and correct the bit errors. The sender adds redundant information to the original packet and sends it to the receiver. The receiver uses this redundant information to reconstruct approximations of exact versions of some of lost packets. FEC is implemented in a lot of protocols used for multimedia communications, e.g. Free Phone and RAT [Kur-2004].

3.1.2 Packet-level Error Control

The packet level error control refers to mechanisms for detecting and correcting the packet-level errors such as loss, duplication and reordering of PDUs. Like bit-level error control, the packet-level error control can be implemented in several protocols and in different layers of communication systems. There are several fundamental mechanisms for detecting and correcting the packet errors, such as sequence number, acknowledgement, timer management, retransmission, automatic repeat request (ARQ), etc. These mechanisms are founded in a lot of communication protocols, e.g. TCP, STCP, and OSPF. Until now, these mechanisms are only described superficial within a particular protocol and were not considered as a separate one. In this section, mechanisms for detecting and correcting the packet level errors will be described.

3.1.2.1 Sequence Number

Senders and receivers use sequence numbers for implementing a reliable data transfer service. The basic principle of packet error detection by using of sequence numbers is very simple. A sequence number in the header of a PDU indicates its unique position in the sequence number of PDUs transmitted by the source. For using the sequence number, before sending the PDU to its destination, the sender labels each PDU that has not been previously transmitted with a consecutive sequence number in the PDU header. The receiver knows which sequence numbers already arrived and thus which sequence number the receiver expects to receive as the next. When a PDU arrives at the receiver, the receiver then reads the sequence number from the header of this PDU and compares with the expected sequence numbers. By this way, the receiver can detect losses, duplication and reordering of packets. If this sequence number is less than receiver's expected sequence number, the receiver knows that this PDU is duplicated and drops it. If this sequence number higher than receiver's expected sequence number, the receiver knows that the packet with the expected sequence number was lost and thus it can request the sender to resend it. Otherwise, if the sequence number is equal to the expected sequence number, the receiver knows that the PDU is correctly arrived. The receiver then processes its header, taking actions depending on header information and on the events arrived.

A well-known example for the using of sequence numbers is found in TCP. Each TCP segment header contains a 32-bit sequence number field. Each TCP segment carries its own sequence number not only during data transmission but also during the TCP connection establishment and release.

3.1.2.2 Acknowledgement

In order to develop a reliable data transfer service, acknowledgement mechanism is used together with sequence number. Acknowledgement enables the receiver to let the sender know whether its data is correctly received or a packet error has occurred. Thus, acknowledgement is used for detecting the packet level error. This mechanism functions as follows. Each time when the data arrives at the receiver, the receiver sends an acknowledgement PDU to the sender of this data. The acknowledgement number field in each acknowledgement PDU will tell the sender about the PDUs arrived at the destination. There are four variants of acknowledgements which can be implemented in each reliable protocol:

- **Positive acknowledgement (ACK):** The receiver informs the sender that it correctly received the data.

- **Negative acknowledgement (NACK):** The receiver informs the sender that it did not receive the data in which it sent a NACK when it detects a gap in sequence numbers of PDUs it received. An NACK contains a range of sequence number of PDUs that have been lost and must be retransmitted. On receiving NACK, the sender retransmits these PDUs. The TCP protocol also implements the negative acknowledgement mechanism in which the TCP receiver sends duplicate acknowledgements when it detects a missing segment. When the TCP sender receives up to 3 duplicate acknowledgements, it knows which TCP segment is missed and retransmits this segment.
- **Selective acknowledgement (SACK):** an SACK is a positive acknowledgement for a particular PDU. Using SACK, an receiver can only acknowledge the sender for one correct received PDU per round-trip-time (RTT)
- **Cumulative acknowledgement (CACK):** a CACK is a positive acknowledgement for a set of PDUs. Using CACK, a receiver can inform the sender about several correct received PDUs per RTT.

3.1.2.3 Retransmission Timer

Other mechanism for packet error detection is to use a retransmission timer. Every time the sender sends data, it starts a timer with a timeout interval. If the sender does not receive the acknowledgement for this data within this timeout, it guesses that the data sent to the destination or the acknowledgement from the receiver was lost and thus the sender retransmits the data.

3.1.2.4 Packet Retransmission

In order to recover the packet errors detected by the mechanisms discussed above, packet retransmission is needed. This mechanism allows the sender to retransmit the missing PDUs if it knows that these PDUs were lost. To provide a reliable data transport service, these four packet error control mechanisms described above should work together within a particular protocol. In the following sections, we take a closer look at how these mechanisms operate together on examples of Automatic Repeat Request (ARQ), Stop-and-Wait, and Go-Back-N.

3.1.2.5 Automatic Repeat Request (ARQ)

Automatic Repeat Request (ARQ) involves the using of a combination of acknowledgements and retransmission timeout to achieve the reliable data transmission. The receiver sends an acknowledgement to the sender to indicate

that it has correctly received a protocol data unit (PDU). When the sender does not receive the acknowledgement before the timeout occurs, the sender retransmits the PDU until it is either correctly received or the number of retransmissions exists a given bound. Three types of the ARQ protocol are Stop-and-Wait ARQ, Go-Back-N ARQ and Selective Repeat ARQ [Tan-2002, PD-2003]. These three protocols are described as follows.

- **Stop-and-Wait ARQ.** Stop-and-Wait is simple ARQ algorithm. Its principle is straightforward: After sending one PDU, the transmitter waits for an acknowledgement from the receiver before sending the next PDU. If the acknowledgement does not receive before the retransmission timeout occurs, the sender retransmits the original PDU. To recognize the duplication of PDUs because of acknowledgement lost or of timeout runs out before PDU reaches the receiver, a bit sequence number is defined in the PDU header. The sequence number alternates from 0 to 1 in subsequent PDUs. When the receiver sends an ACK, it includes the sequence number of the next PDU it expects. By this way, the receiver can detect duplicated PDUs by checking the sequence numbers. The disadvantage of this protocol is that it only can send one PDU per round-trip-time, and therefore the throughput may be far below the link's capacity.
- **Go-Back-N ARQ** [PD-2003]. Go-back-N ARQ protocol improves the Stop-and-Wait protocol in which the sender is allowed to send a number of PDUs specified by a credit window size without waiting for an acknowledgement from the receiver. If a timeout occurs, the sender resends all PDUs that have been previously sent but have not been yet acknowledged. Go-back-N can achieve better throughput than Stop-and-Wait, because during the time that would otherwise be spent waiting, more PDUs are being sent. However this protocol results in sending PDUs multiple times if the PDUs were dropped in the first time or the acknowledgement for them was dropped. To avoid it, Selective Repeat ARQ can be used.
- **Selective Repeat ARQ** [Tan-2002]. This protocol avoids unnecessary retransmission by having the receiver stores all the correct PDUs following the bad one and having the sender retransmits only those PDUs that it suspects were received in error. Each sender and receiver maintains its own sliding window (defined as the sending window by the sender and the receiving window by the receiver). The receiver continues to fill its receiving window with subsequence PDUs, keeps track of the sequence numbers of the earliest PDUs it has not received and sends these sequence numbers in the ACK to the sender. If a PDU from the sender does not

reach the receiver, the sender sends subsequent PDUs until it has emptied its sliding window. The sender must also keep a buffer of all PDUs which have been sent, but have not yet been acknowledged, until the retransmission is complete. The recovery of lost or corrupted PDUs is handled in following four stages: First, the corrupted PDU is discarded at the receiver; second, the receiver requests the retransmission of missing PDU using a control PDU (called Selective Reject acknowledgement). The receiver then stores all out-of-sequence PDUs in the receiver buffer until the requested PDU has been retransmitted; Third, upon receiving a Selective Repeat acknowledgement, the sender then transmits the lost PDU(s) from its buffer of unacknowledged PDUs. The sender then continues to transmit new PDUs until the PDUs are acknowledged or another selective repeat request is received; Fourth, the receiver forwards the transmitted PDUs to the upper layer protocol instance, and all subsequent in-sequence PDUs which are held in the receive buffer. The selective repeat ARQ is employed by the TCP transport protocol.

3.2 Multiple Access Control

There are two basic types of communications – the point-to-point and the broadcast. A point-to-point communication is performed through a medium with exactly two endpoints – a single sender at one end and a single receiver at another end. In order to achieve the data delivery between these two endpoints, hierarchical addressing and routing are needed. In comparison with the point-to-point communications, a broadcast communication can have multiple sending and receiving nodes connecting to the same shared medium (e.g. shared wire, shared wireless). No routing is needed here. When a node transmits a message, the shared medium broadcasts the message and each node receives a copy of this message. However when multiple nodes send the messages into the shared medium at the same time, all of the nodes receive multiple messages at the same time. As a result, the transmitted messages collide at all receivers. Thus, all messages involved in collision are lost. In order to ensure that shared medium performs correctly when multiple nodes are active, coordinating the transmission of multiple active nodes is needed. This is the job of the so called multiple access control (MAC) that deals with mechanisms for sharing a single broadcast medium among competition users.

A classification of the multiple access control mechanisms is shown in the figure 3-3. While the static channel allocation divides channel to individual users so that no collision arrives absolutely, the dynamic channel allocation tries to minimize the incidence of collision to achieve a reasonable usage of the

shared medium. In a random access mechanism, an active node always transmits data at full rate. When there is a collision, each node involved in the collision retransmits the message until the message gets through without collision. The basic idea of the taking turn control is to use either a polling mechanism to poll each active node in round-robin fashion to give them permission to transmit their data or a token-passing method to allow a node to send data if it holds a token.

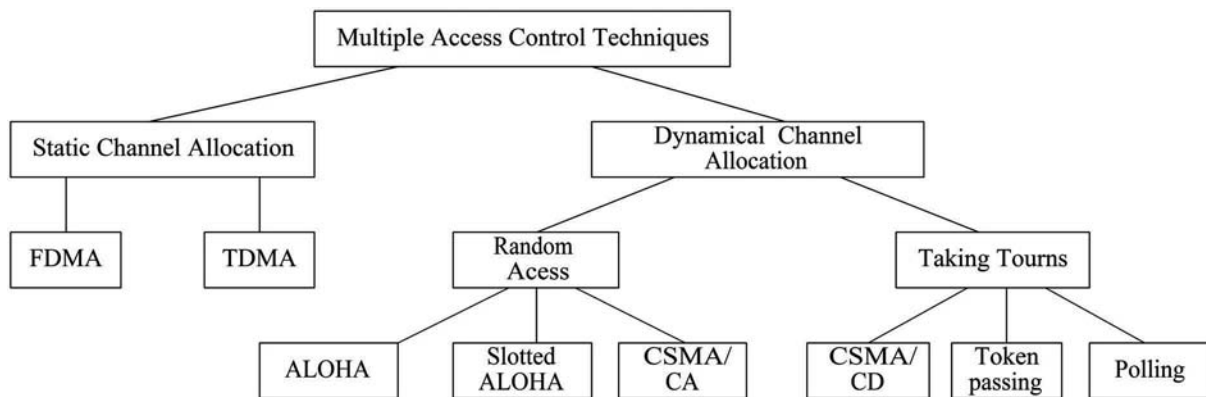


Figure 3-3: A classification of multiple access control mechanisms

In the next following sessions the fundamental mechanisms for multiple access control will be described. The session starts with an overview of static channel allocation mechanisms. After that, mechanisms for random access and for taking turn will be illustrated.

3.2.1 Static Channel Allocation

The Frequency Division Multiple Access (FDMA) and the Time Division Multiple Access (TDMA) are two static channel allocation techniques that can allocate a single channel to multiple competing nodes independent of their activity. The basis principle of both these techniques is to partition the time bandwidth space into slots which are assigned to node population in a static predefined fashion.

3.2.1.1 Frequency Division Multiple Access

The FDMA shares the W bps broadcast channel into different frequencies, each with a bandwidth of W/N and allocate each frequency to each of N stations. Every station involved in the data transfer sends its data at different frequency. Since each station has a private frequency band, there is no collision between the stations. Figure 3-4 shows an example of FDMA for 3 sending stations (s_1 , s_2 , s_3) and three receiving stations (r_1 , r_2 , r_3). For each pair of sending and receiving node, a frequency band is allocated. The s_1 sends data to r_1 via channel 1 for

which a frequency band 2001-4000 Hz is assigned. The s2 sends data to r2 via channel 2, and the s3 sends data to r3 via channel 3.

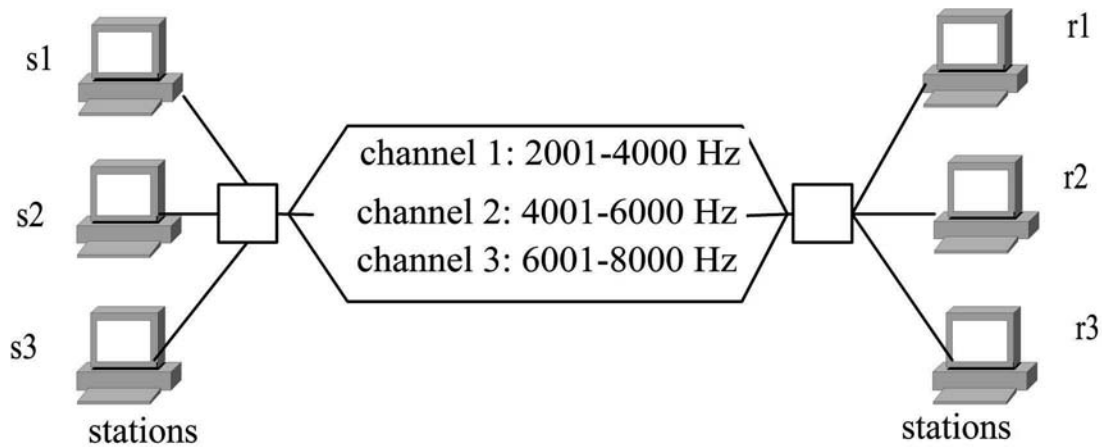


Figure 3-4: FDMA for three pairs of sending and receiving stations

The advantage of the FDMA is that it avoids the collision via sharing the bandwidth among the participating nodes. But the main disadvantage of the FDMA is that every station is limited to a bandwidth of W/N , even when only a few of N stations has data to send.

3.2.1.2 Time Division Multiple Access

While the FDMA splits the W bps channel into different frequencies, the TDMA divides the broadcast channel in time. If there are N stations, TDMA divides the shared channel into time frames, and then divides each time frame into N time slots (see figure 3-5).

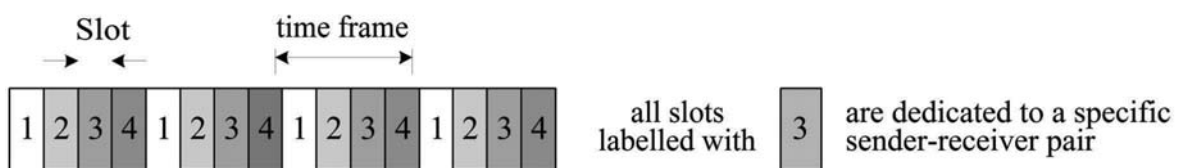


Figure 3-5: TDMA Principle

Each time slot is then assigned to one of N stations. Whenever a station has a frame to send, it transmits the frame's bits during its assigned time slot in the revolving TDMA frame. The TDMA eliminates the collisions in which each station gets a dedicated transmission rate of W/M during each frame time. TDMA shares both advantage and disadvantage of FDMA. In addition to these, a station in TDMA must always wait for its turns in the transmission sequence, even when only one node has data to send.

3.2.2 Dynamical Channel Allocation

If two frames are transmitted simultaneously via a shared medium, they overlap in time. Do the transmitted signals are arrived at the same time by a receiver, these overlapped signal can't not be separated from each other. This situation is called collision. In order to avoid such situation, either the channel is divided to individual users so that no collision arrives absolutely, or the incidence of collision is minimized by using of random access or taking turn to achieve a reasonable usage of the shared medium. The first one is the static channel allocation, which is illustrated in the last sub session. And the second one is the dynamic channel allocation that will be described in this sub session.

The sub session starts with the random access mechanisms. Following this, the taking turn algorithms will be addressed.

3.2.2.1 Dynamic Channel Allocation with Random Access

The initial basic idea of the random access is to allow multiple stations to access on a shared communication medium in which a transmitting station always transmits with the full rate of the channel. When there is a collision, each station involved in this collision repeatedly retransmits its frame until the frame gets through without collision. But if all stations involved in the collision send its frame at the same time, the collision will get more. Therefore, the important key of the random access is that when a station experiences a collision it doesn't necessary retransmit its frame immediately, it waits for a random delay before transmitting the frame. Thus the collision can be recovered via transmitting the frame after a random delay. The random delays for each station are chosen independent. It is possible that one station will determine a random delay which is less than delays of other colliding stations and will be able to sneak its frames into the channel without collision.

The random access protocols are ALOHA, slotted ALOHA, CSMA, CSMA/CD and CSMA/CA. These protocols and their mechanisms will be illustrated in this session.

3.2.2.1.1 ALOHA and Slotted ALOHA

The ALOHA was proposed from Norm Abranson in 1970. It is the first random multiple access protocol developed for radio-based communication in Hawaii. In ALOHA, when a station has generated a new frame, this frame is transmitted immediately. The station does not need to observe the channel before sending the frame. After the transmission, the station waits for an acknowledgement from the receiver. If no acknowledgement is received within a predefined period, the transmitted frame is assumed to have been lost because of a collision;

the station waits for a random amount of time before retransmitting the frame. The basic principle of ALOHA is illustrated in figure 3-6.

Let t be the time required to send a frame in ALOHA. If any other station has generated a frame between t_0+t and t_0+2t , the end of that will collide with the beginning of the shared one (see figure 3-7). Moreover, if any other station has generated a frame between t_0+t and t_0+2t , the beginning of that will collide with the end of the shaded one. Therefore the critical interval of ALOHA is $2t$ (see figure 3-7).

Slotted ALOHA was developed in order to reduce the collision within the critical interval in ALOHA. In slotted ALOHA, all frames are assumed consisting of exactly L bits. The time to transmit one frame is divided into slots of size L/R seconds where R bps is the throughput of each station. In order to eliminate the collision, stations start to transmit frames only at the beginning of slots. The synchronization between stations enables each station to know when the slots begin.

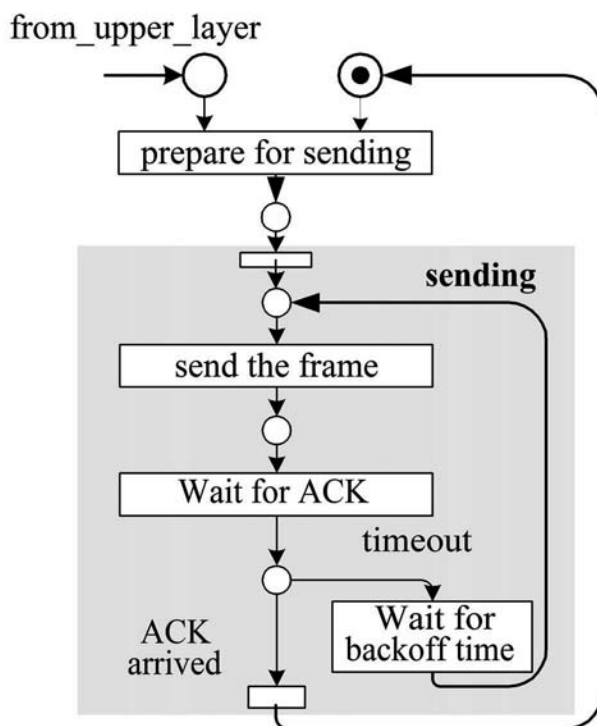


Figure 3-6: ALOHA Principe

The basic idea of the slotted ALOHA is described in figure 3-8 and can be formulated as follows. When a station has a new frame to send, it waits until the beginning of the next slot and transmit the entire frame in the slot. If there isn't a collision, the station can prepare a new frame for further transmission if it has one. Otherwise, if there is a collision, the station detects the collision before the

end of the slot. And the station retransmits the frame with a probability p (a number between 0 and 1) until the frame is transmitted without collision.

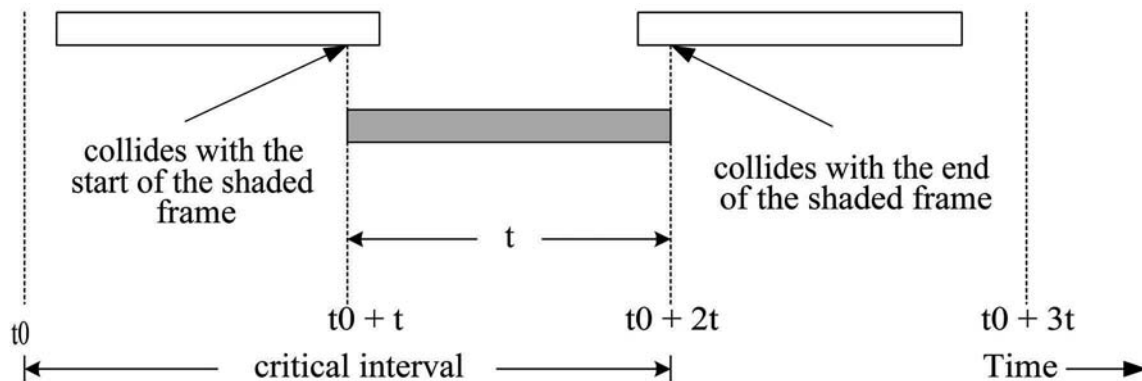


Figure 3-7: Critical Interval of ALOHA

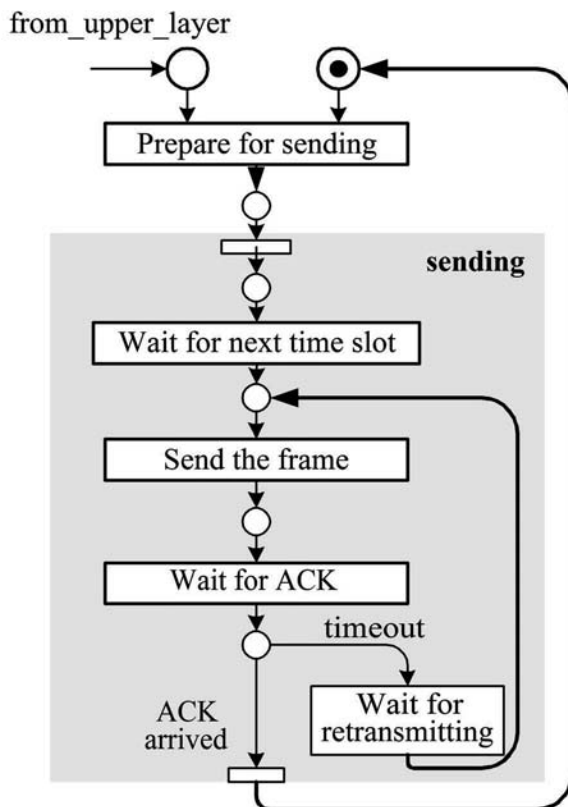


Figure 3-8: The basic principle of the slotted ALOHA

3.2.2.1.2 Carrier Sense Multiple Access

The drawback of ALOHA and slotted ALOHA is that a station transmits data as it will, without paying attention to what the other station are doing. Thus, many collisions may occur. A well-known solution for wired LANs is the so called

Carrier Sense Multiple Access (CSMA). Its key idea is that each station must be able to detect what other stations are doing and therefore this station can adapt its behaviour accordingly. In CSMA, stations listen for a carrier (i.e. a transmission) to see if there are signals on the cable. If no signals on the cable, station can send the data. Otherwise the station keeps listening to the channel (figure 3-9).

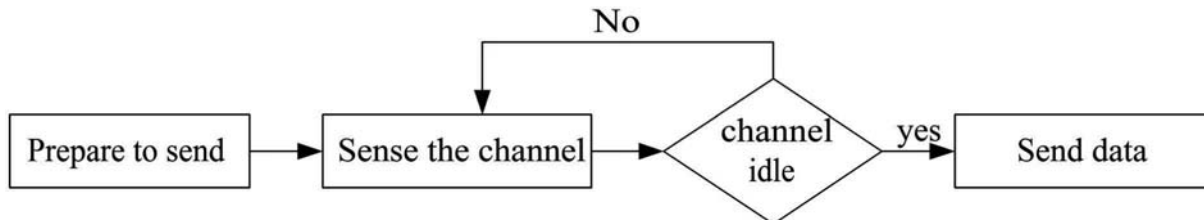


Figure 3-9: Concept of the CSMA

There exist several versions of CSMA. They are 1-persistent CSMA, p-persistent CSMA and the CSMA/CD.

3.2.2.1.2.1 1-persistent CSMA

If a station has data to send, it first listens to the channel to see if anyone else is transmitting at this moment. If the channel is busy, this station waits until it becomes idle. When the station detects that the channel is idle, it immediately transmits a message and waits for acknowledgement from the receiver. If a collision occurs, that means no acknowledgement arriving within a given time interval, the station waits for a random amount of time (back-off time) and starts listen to the channel again. The mechanism is called “1-persistent” because the station transmits with a probability of 1 whenever it finds that the channel is idle.

The problem with 1-Persistent CSMA is that if a number of stations listen to the channel, then they shall all send data as soon as the channel becomes idle, guaranteeing a collision.

3.2.2.1.2.2 p-Persistent CSMA

When a station is ready to send data, it senses the channel. If the channel is busy, this station waits until it becomes idle. And if the station detect that the channel is idle, it transmits a message with a probability of p (see figure 3-10). If a collision occurs, the station waits for a random amount of time (back-off time) and starts listen to the channel again. Because of transmitting a message with a probability of p , the station defers with a probability of $1-p$ until the next time slot.

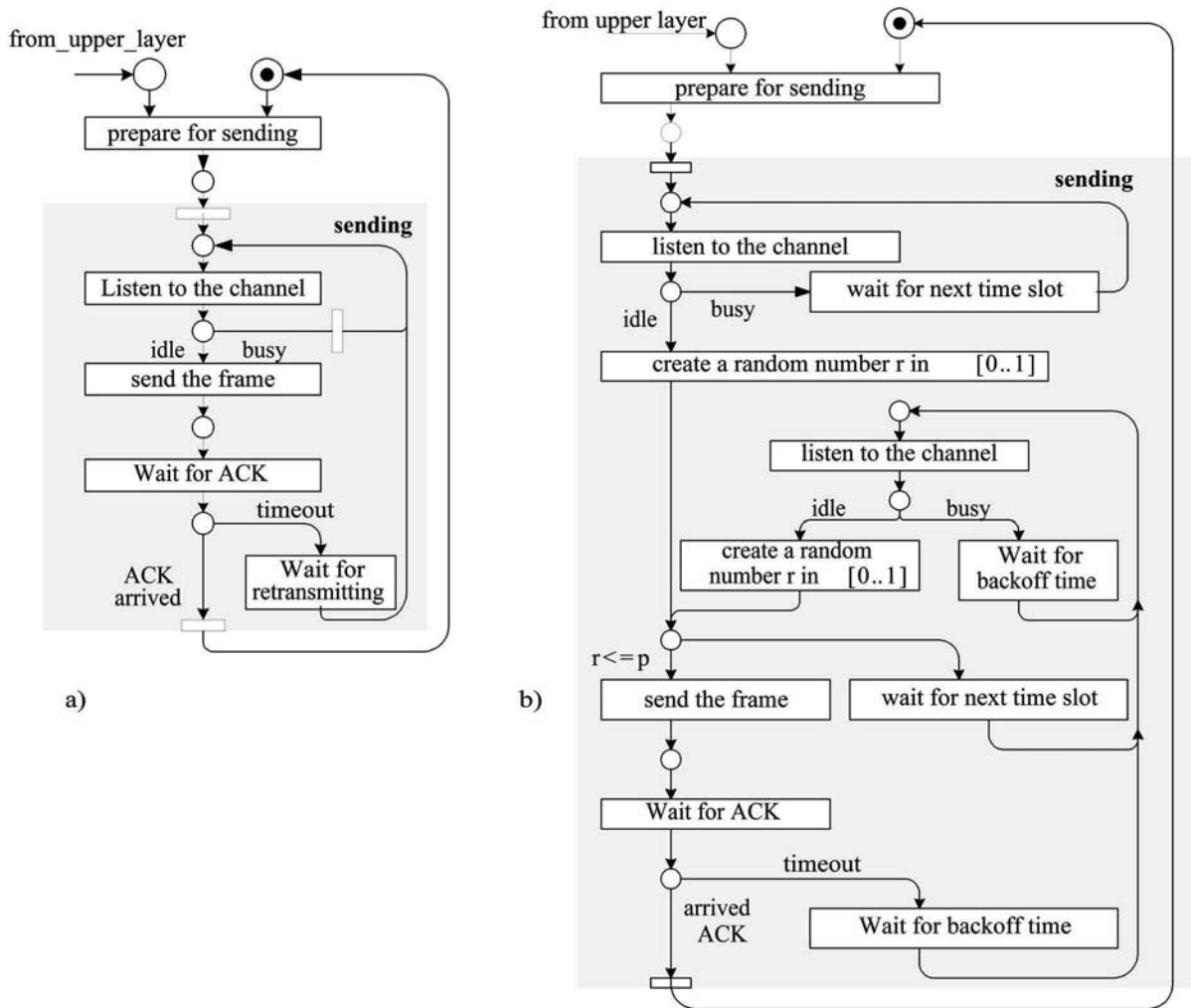


Figure 3-10: (a) The 1-persistent CSMA (b) p-Persistent CSMA

3.2.2.1.3 Carrier Sense Multiple Access with Collision Detection

The multiple access mechanisms illustrated in the last sub session do recognize the collision by detecting the acknowledgement missing which results in a timeout. In these algorithms, if a collision occurs, each of these mechanisms waits for a random amount of time and starts the data transmission again. The drawback if ALOHA, slotted ALOHA and CSMA is the transmission delay because of waiting for the timeout to recognize the collision and of waiting for the next time slots.

In order to reduce the waiting time in ALOHA, slotted ALOHA and CSMA, each station must be able to detect the collision without waiting for the timeout. Carrier Sense Multiple Access with Collision Detection (CSMA/CD) is a solution for this problem. CSMA/CD is the IEEE standard 802.3 for 1-persistent CSMA/CD LAN and is used in the Ethernet. It operates at the data link layer. Because most part of data link layer protocols is implemented in an adapter,

instead of using station at the link layer we use the adapter. In CSMA/CD, each adapter may begin to transmit at any time. That is, no time slots are used. Before the transmission and during the data transmission, each adapter senses the channel when some other stations are transmitting and detects a collision by measuring voltage levels. An adapter never transmits its data when it senses that some other stations are transmitting. That is, it uses the collision detection. A transmitting adapter aborts its transmission as soon as it detects that other station is also transmitting. In order to avoid that many adapters immediately start transmitting the data when the channel becomes free, an adapter waits for a random time before attempting a retransmission. The advantages of the CSMS/CD is that no synchronization is needed – each adapter runs CSMA/CD without coordination with other adapters. By this way, transmission delay will be reduced.

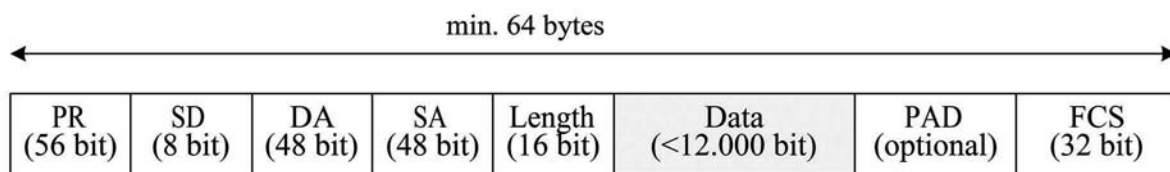


Figure 3-11: CSMA/CD Frame format

The CSMD/CD is specified as the IEEE standard 802.3 for 1-persistent CSMA/CD LAN. The CSMS/CD is used in the Ethernet. Figure 3-11 shows the CSMA/CD frame format [Tan-2002].

- **Preamble (PR):** The preamble for the synchronization of the receivers.
- **Start-of-frame delimiter (SD):** SD shows the start of frame.
- **Destination address (DA):** the destination address the frame should be sent to.
- **Source address (SD):** the address from which the frame is sent
- **Length:** the number of octets in the data field
- **Data:** this field carries PDU of the upper layer
- **Padding (PAD):** the field is used to extend the data field
- **Frame check sum (FCS):** the field is used for bit error detection.

Figure 3-12 shows how CSMA/CD works:

1. The adapter obtains a network layer PDU. It prepares an Ethernet frame and put the frame in an adapter buffer.
2. If the adapter senses that the channel is idle (that is, there is no signal energy from the channel entering the adapter), it starts to transmit the frame. If the adapter senses that the channel is busy, it waits until it senses no signal energy and then starts to transmit the frame.

3. While transmitting, the adapter monitors for the presence of signal energy coming from other adapters. If the adapter transmits the entire frame without detecting signalling energy from other adapters, The adapter is finished with the transmitting of this frame.
4. If the adapter detects signal energy from other adapters while transmitting, it stops transmitting its frame and instead transmits a 48-bit jam signal to all adapters to tell them that there has been a collision.
5. After sending the jam signal, the adapter enters an exponential back-off phase. After experiencing the n th collision, the adapter chooses a value for K as random from $\{1, 2, 3, \dots, 2^{m-1}\}$ where m is chosen as minimum of n and 10. The back-off time is then set equal to $K \times 512$ bit times. The adapter waits for a random amount of time (back-off time) and then return to the step 2
6. After receiving a jam signal, a station that was attempting to transmit enters an exponential back-off phase. It waits for a random amount of time and then returns to step 2.

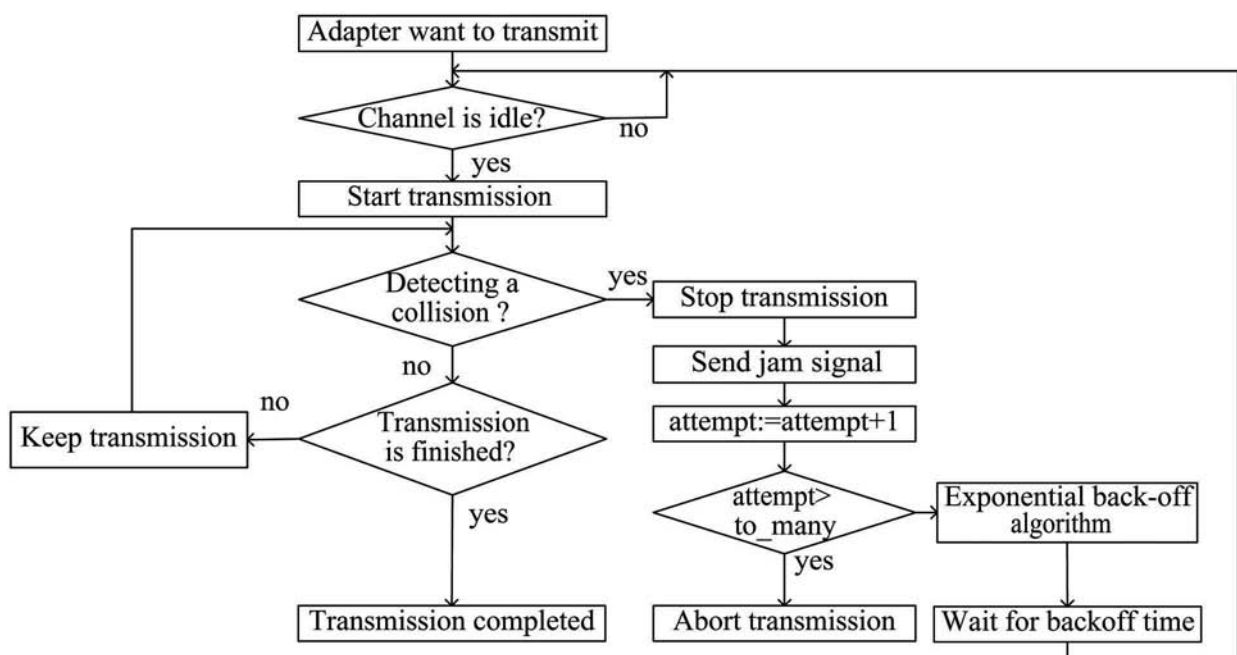


Figure 3-12: The CSMA/CD Protocol

3.2.2.1.4 Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA)

CSMA/CD cannot be used in a wireless LAN environment for following main reasons. Implementing a collision detection mechanism requires the

implementation of full duplex ration, but wireless communications are in half-duplex. In a wireless environment, we cannot assume that all stations hear other (which is the basic assumption behind the collision detection schema), and the fact that a station wants to transmit and senses that the medium is free doesn't necessary means that a medium is free in the receiver's area. If we had an antenna to listen and another to transmit we should be able to detect a collision while we transmit. This time the medium is the air and the power of the transmitting antenna will confuse the receiving one thus making detection almost impossible.

CSMA/CA RTS packet

Frame control (16 bit)	Duration (16 bit)	Receiver address (48 bit)	Transmitter address (48 bit)	Frame Check Sequence (32 bit)

CSMA/CA CTS packet

Frame control (16 bit)	Duration (16 bit)	Receiver address (48 bit)	Frame Check Sequence (32 bit)

Figure 3-13: CSMA/CA RTS and CTS Packet

The IEEE 802.11 standard Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) utilizes the congestion avoidance mechanism together with a positive acknowledgement scheme. A station willing to transmit a packet will first transmit a Request To Send (RTS) packet to the destination. The destination station will respond (if the medium is free) with a CTS (Clear to Send) packet with the same duration information. All stations hearing either RTS and/or CTS know about the pending data transmission and can avoid interfering with those transmissions. Receipt of CTS will indicate to the transmitter that no collision occurred. If the channel is sensed idle for DISF (Distributed Inter Frame Space) sec then it transmits the entire frame. Otherwise, if the channel is sensed busy, the station waits for a random back-off time and tries again. If the frame is received correctly and completely at the receiver, the receiver returns an explicit ACK back to the sender after SIFS (short inter-frame space) (figure 3-14).

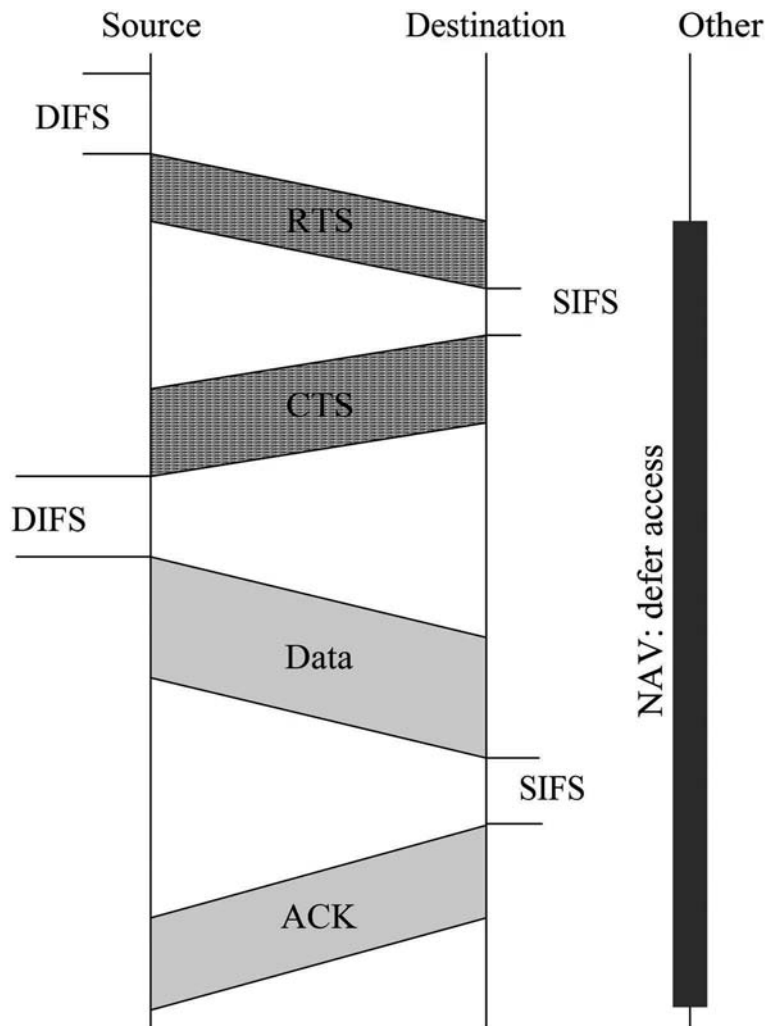


Figure 3-14: The basic principle of CSMA/CA

3.2.2.2 Dynamic Channel Allocation with Taking Turns

The static channel allocation mechanisms and the CSMA can share the channel fairly, but a single station cannot use it at all. In comparison with this channel allocation class, random access channel allocation mechanisms allow a single station to full channel rate, but cannot share the channel fairly. Taking turns protocols achieve both fairness and full rate, but at the expense of some extra control overheads. This overhead is either polling from a master station or using a control token.

There are a lot of taking turns protocols. Two well-known protocols and their algorithms will be discussed in this session. They are polling and token passing protocol.

3.2.2.2.1 Polling Mechanism

The basic principle of the polling mechanism is to assign a station as the master station which polls each of other stations in a round-robin fashion. In particular, the master sends the so called “request to send” message to a slave station to request this station to transmit the data. The slave station that receives the “request to send” responds to the master with “Clear to send” and it can transmit up to some maximum number of messages. After this slave station transmits some data, the master tells the next slave node that it can transmit up to some maximum number of messages. The polling mechanism guarantees that no collision may occur. But it has some disadvantages. The first one is that the mechanism produces a polling delay – the time the master needs to inform a slave station that it can transmit data. The second disadvantage is the single point of failure. If the master node fails, no data transmission is possible.

3.2.2.2.2 Token Passing Mechanism

The token passing mechanism doesn’t need the master node. Instead of this, a special packet known as token is exchanged between the stations in a pre-defined fixed order. For example, station 1 may send token to the station 2, station 2 may always send token to station 3. When a station receives a token, it only holds the token if it has some data to send, otherwise it passes the token to the next station. If a station does have data to send when it receives a token, it sends up to maximum number of data and forwards the token to the next station. In comparison with the polling algorithm, the token passing is a decentralized approach. But it also has problems as well. Fair of a station can break down the whole channel. Also token may be lost if the station holding the token has some accident.

The token passing mechanism is used in the token ring protocol [Tan-2002].

3.3 Traffic Access Control

Traffic access control covers mechanisms for filtering the source traffic flows at the entry and at the specific points within the network. Once a flow connection is accepted, its emitting traffic to the network should conform the traffic descriptor. Otherwise, the excess traffic can be dropped, marked with a lower priority, or delayed (i.e., shaped). Well-known mechanisms for traffic access control are traffic description, traffic classification, policing, marking, shaping and metering. These mechanisms will be described in the following paragraphs.

3.3.1 Traffic Description

The traffic description issue deals with describing a set of parameters that can be used to classify expected characteristics of source traffic arriving at a router. Such a set of parameters is called traffic descriptor. A traffic descriptor first forms the basis of a traffic contract between the source and the network: If the source sends traffic confirming to its descriptor, the network promises a particular quality of service; if a source violates its traffic descriptor, the network cannot guarantee it a performance bound. Furthermore, the traffic descriptor is the input to a shaper, which delays the traffic in a buffer when the source rate is higher than its expected rate. Moreover, the descriptor is also the input to a policer and to a meter. While the policer drops the source traffic that violates the descriptor, the meter marks/remarks the traffic as out-of-profile if the source traffic does not confirm its descriptor.

Several traffic descriptors have been proposed in the literature. Three common descriptors are peak rate, average rate and linear bounded arrival process that are described as follows [Kes-2001]:

- **Peak-rate descriptor.** The peak rate is the highest rate at which a source can ever generate data. For networks with fixed-size packets, the peak rate is defined as the inverse of the closest spacing between the arriving times of consecutive packets. For variable-sized packets, the peak rate is specified along with a time window over which the peak rate is measured. The peak rate is defined as the total number of packets generated over all windows of a specified size. The peak rate descriptor is easy to be computed, but it can be a very loose bound. The key problem with the peak rate descriptor is that a single outlier can change this descriptor. Therefore, peak rate descriptors are useful only if the source traffic is very smooth.
- **Average-rate descriptor.** The average rate is the rate measured over a period of time. The motivation of average-rate descriptors is that averaging the transmission rate over a period of time will reduce the effect of outliers. Two well-known types of average-rate descriptors have been proposed – the jumping-window descriptor and the moving-window descriptor. Both descriptors use two parameters – the time window t over which the rate is measured, and the number r of bits that can be sent in a window of time t . Using the moving-window descriptor, the source claims that over all windows of length t seconds, no more than r bits of data will be transmitted. Using the jumping-window descriptor, a new time window starts immediately after the end of the early one, so that a traffic source

claims that over consecutive windows of length t seconds, no more than r bits of data will be injected into the network.

- Linear bounded arrival process (LBAP). The LBAP descriptors basically include at least two parameters: the long term average rate r allocated by the network to the source and the longest burst s a source may send. Based on LBAP, number of bits a source sends in any time interval of length t is bounded by $rt+s$. Examples of mechanisms for regulating an LBAP descriptor are the token bucket and leaky bucket.

3.3.2 Traffic Classification

Routers classify packets to determine which flow they belong to, and to decide what service they should receive. Thus, packet classification refers to mechanisms for categorizing packets by examining the contents of some fields in IP header and/or some fields in the transport header. Its basis principle is to pick some set of N bits in a packet header (and payload) to differentiate up to 2^N classes of packets. This set of N bits is called classification key. The act of classifying a packet involves matching the fields in the classification key against a set of classification rules. Thus, a classification rule is defined as a set of actions should be performed for each classification type at the end system and at the intermediate network device. For example, when a packet arrives at a router, the router checks to see if the packet's classification type matches the type specified in a rule. If it does, then the actions defined in that rule will be applied to the packet.

Two common packet classification schemes are the multiple field (MF) classification and the behaviour aggregate (BA) classification [Cisco-1, Cisco-2]. The MF uses a key covering multiple fields of the IP packet header and payload. These multiple fields may be some or all of the IP packet header fields that usually define an IP flow – typically, the IP source and destination addresses, protocol field, and source and destination TCP/UDP port numbers. The MF classification scheme provides the greatest amount of context to router's subsequent processing stages. However, when the network developer only needs a small number of traffic classes to be differentiated at any given hop, the usually scheme is the behaviour aggregate (BA) classification. Its main principle is to assign a handful of bits at a fixed, known location within the packet header. For examples, the IPv4 type of service, IPv6's traffic class, IPv6's flow label and DiffServ code point are the fixed bit combinations used for classification.

3.3.3 Traffic Policing and Traffic Shaping

Each traffic class has a traffic profile defining certain limits to its allowable temporal behaviour – a limit on arrive rate of the packets at a router and burst size during some specified interval. Policing and shaping are mechanisms defining the actions taken by a router when it determines that a packet is outside the limits assigned to the traffic class to which the packet belongs. Both policing and shaping mechanism monitor the packet arrival for conformity with the predefined traffic profile. While traffic policing simply drops out-of-profile packets, traffic shaping delays these out-of-profile packets in a buffer and sends them for later transmission over increments of time. By this way, traffic shaping and policing can control the volume of traffic sent into the network, the rate at which traffic is being sent, and the burst of traffic. For this reason, these two mechanisms are commonly implemented at the network edges to regulate the rate at which a flow is allowed to put packets into the network. By doing so, they can reduce the network congestion and therefore improve the performance. Four important criteria used by traffic policing and shaping are:

- Mean rate — specifies how much data can be sent or forwarded per unit time on average. A network may wish to limit the long-term mean rate at which packets belonging to a flow can be sent into the network.
- Peak rate — is a constraint that limits the maximal arrive rate of a flow over a short period of time.
- Burst size — specifies (in bytes) how much traffic can be sent within a given unit of time. The network may also wish to limit the maximum number of packets a flow can sent into the network over an extremely short interval of time.
- Time interval — specifies the time quantum in seconds per burst.

Because traffic policing uses the token bucket and traffic shaping uses the leaky bucket mechanisms, the token bucket and leaky bucket mechanisms will be first explained.

3.3.3.1 Traffic Policing by using Token Bucket

A token bucket is a formal characterization of a rate transfer. It has three components: a bucket, a rate r and a time interval Dt (figure 3-15). The bucket can hold at most q tokens. New tokens are generated at a rate of r tokens per second and added to this bucket if the bucket is filled with less than q tokens; otherwise the newly generated tokens are discarded. Each token is permission for the source to send a certain number of bits into the network. To send a packet, the policing must remove from the bucket a number of tokens equal in

representation to the packet size. If there are not enough tokens in the bucket to send a packet, the policing simply drops this arriving packet. Because at most q tokens can be in the bucket, the maximum burst size for a policed flow is q packets. Furthermore, token is generated at the rate r , therefore the maximum number of packets that can be sent into the network at any interval of time of length Dt is limited to $(r.Dt + q)$.

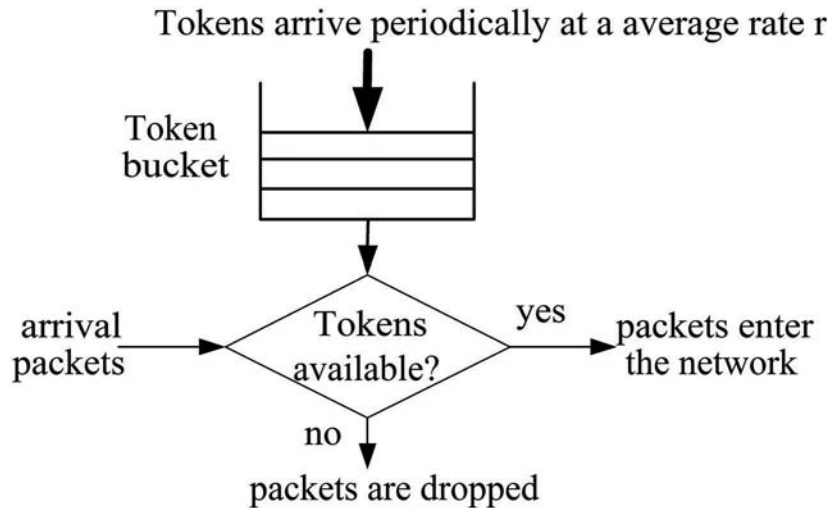


Figure 3-15: The token bucket principle

3.3.3.2 Traffic Shaping by using Leaky Bucket

A leaky bucket consists of a data buffer of size x , and a token bucket that can hold at most q tokens. New tokens are generated at fixed rate r measured in token per second, and added to the bucket if the bucket is filled with less than q tokens; otherwise the newly generated tokens are ignored.

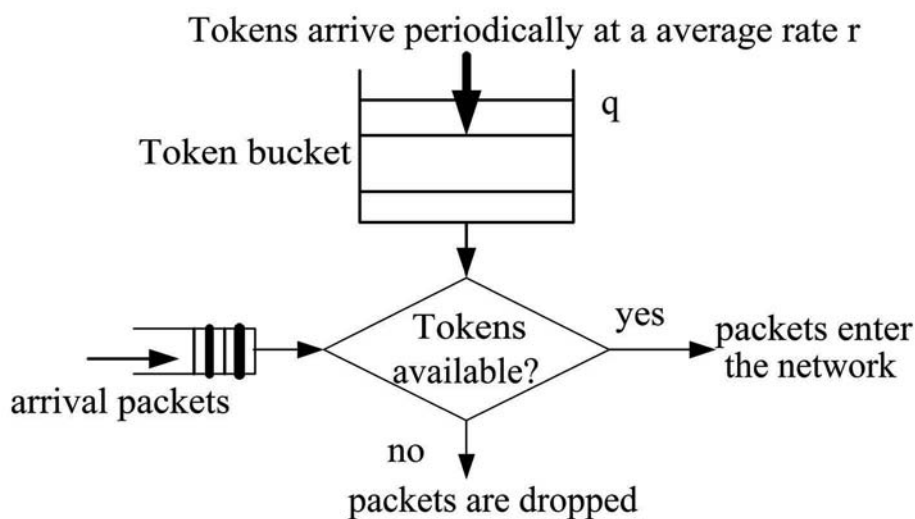


Figure 3-16: The leaky bucket principle

When a packet arrives, a token is removed from the bucket, and the packet is sent into the network. If the token bucket is empty and the data buffer is not full, traffic shaping simply delays the packets in the data buffer. Otherwise the packet is dropped. The packets delayed in the data buffer will be sent into the network if tokens are available. Shaping and policing are implemented e.g. in Cisco IOS release 12.2 [Cisco-3, Cisco-4].

3.3.4 Marking

Packet marking mechanisms enable routers and end hosts to modify some bits inside an IP header and/or transport header to indicate the service level this packet should receive from other network devices. Packets can be marked in several fields in their IP headers (e.g. IPv4 precedence (3 bits), the DiffServ code point (6 bits), ToS (4 bits), IPv6's traffic class (6 bits) and the flow label (20 bits)) and in their payloads. Packet policing and marking are closely related actions taken by a router when it observes a packet is outside the limits assigned to the traffic class this packet belongs to. While policing drops the out-of-profile packets, marking modifies one or more header's bits of these packets and passes them to the routers' output queuing and scheduling.

3.3.5 Metering

The traffic metering can be used by routers and end hosts to determine whether the arriving packets are "in profile" or "out of profile". It basically compares the current traffic characteristics with the traffic profile defined in the traffic description at the network devices. Each traffic class has certain limits to its allowable temporal behaviour – a limit of how fast packets may arrive or a limit on a number of packets that may arrive during some specified time interval. Packets are recognized to be out-of-profile if their observed parameters are outside the limits assigned to their traffic class. Packets are defined to be "in profile" if their measured parameters are inside the limits assigned to the traffic class these packets belong to. For example, by traffic class with peak rate description (PRD), packets are defined as "out of profile" if their peak rates are more than the peak-rate defined by PRD for this traffic class. Otherwise, the packets are defined as in-of-profile.

Traffic policing can implemented via a simple token bucket mechanism shown in figure 3-17. Tokens are periodically generated in the token bucket at a rate of r tokens per second. When a packet arrives and if there are enough tokens in the bucket to send this packet, some tokens are removed and the packet is marked as in-of-profile and is then sent to the network. Otherwise, the packet enters the network but it is marked as out-of-profile packet. By this way, traffic

metering will inform the routers, which are in congestion situation, to first drop out-of-profile packets.

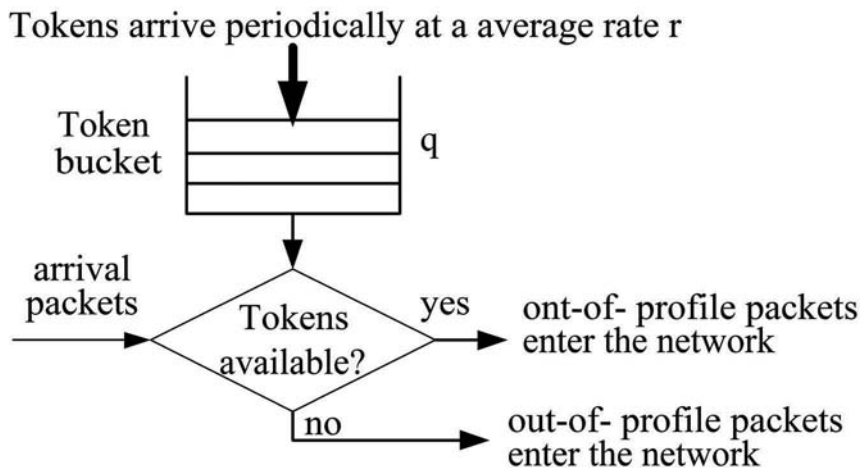


Figure 3-17: Traffic metering by using a simple token bucket

For metering several traffic classes, multiple token buckets can be configure to run simultaneously, associated with bucket size (q) and bucket rate (r) parameters (Figure 3-18). When a packet arrives from the classification, a token bucket is selected for metering this packet. For example, voice over IP packets are metered by the bucket1, video packets are metered by bucket2, and default packets are metered by bucket3. By each token bucket, the packets are marked as in-of-profile or out-of-profile as discussed by the simple token bucket metering.

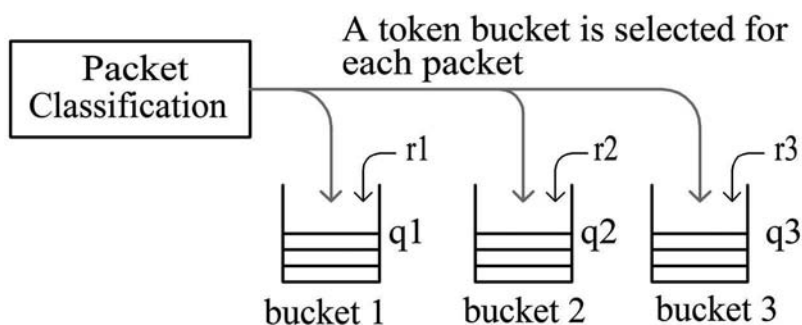


Figure 3-18: traffic metering by using of multiple token buckets

3.4 Packet Scheduling

Packet queuing is a process of buffering incoming packets at the entrance of a communication link into a queue (or multiple queues). In particular, queuing defines a position in the queue to locate each incoming packet – whether at the

beginning of the queue, at the end of the queue or at a random position. Thus, queuing manages the buffer of packets waiting for services.

In contrast to queuing, scheduling is responsible for enforcing resource allocation to an individual flow connection. When there is not enough resource to accommodate all flows, packets will wait in the queue for the service. Given multiple packets waiting in a queue, scheduling defines which packet to be served next. By this way, the scheduling decides the order in which it serves the incoming packets. The packet scheduling is very important because the performance received by a connection principally depends on the scheduling discipline used at each multiplexed server along the path from source to destination. At each output queue, the server uses a scheduling discipline to select the packets for next transmission. Thus, the server can allocate different main delays to different connections by its defining of service order. It can assign different bandwidths to connections by serving at least a certain number of packets from a particular connection in a given time interval. Moreover, it can allocate different loss rate to connections by giving them more or less buffers. To build a network that provide performance guarantee for given applications, scheduling disciplines are required to support delay, bandwidth, and loss bound for each particular connection flow or for a set of aggregated connections.

In this section we first discuss the basic requirements and design choice for packet scheduling and then describe some popular scheduling mechanisms for supporting QoS.

3.4.1 Requirements

A scheduling discipline providing QoS must satisfy following two basic requirements [Kes-2001, Kle-2011]. Firstly, this scheduling must support a fair sharing of the resources and isolation between competing flows. Secondly, the scheduling must provide the performance bounds for real-time multimedia applications. These requirements are described in more detail in this paragraph.

3.4.1.1 Resource fair sharing and isolation for elastic connection flows

The elastic traffic doesn't require any performance guarantee from the network. However if there are multiple competing elastic flows, the scheduling is required to provide fair allocation of the resources, such as buffer space and bandwidth. A scheduling allocates a share of the link capacity and queue size to each flow it serves. An allocation is called fair sharing if this allocation satisfies the max-min fair allocation criterion discussed below. Isolation means that misbehaviour by

one flow sending packets at a rate faster than its fair share should not affect the performance received by other flows.

Max-min fair share

The max-min fair share is an algorithm used for fair sharing of the resources among a set of competing flow connections, while some connections require more resource than other. The max-min fair share allocation is defined as follows:

- Resources are allocated in order of increasing demands,
- Flow connections get no more resource than they need,
- Connections, which have not been allocated as their demands, will get an equal share of the available resource.

The basis principle of the max-min fair share is detailed described in the following. Consider a set of flow connections 1, 2, .., N that have resource demands $x_1, x_2, \dots, x_N$ with $x_1 \leq x_2 \leq \dots \leq x_N$. Let C is the given capacity of the resource shared among N connections, m_n is the actual resource allocated to the connection n with $1 \leq n \leq N$, M_n is the resource available to the flow connection n. The parameters m_n and M_n are determined as follows:

$$M_1 = \frac{C}{N}; \quad (3.1)$$

$$m_1 = \min(x_1, M_1) \quad (3.2)$$

$$M_n = \frac{C - \sum_{i=1}^{n-1} m_i}{N - n + 1}; \text{ for } 2 \leq n \leq N \quad (3.3)$$

$$m_n = \min(x_n, M_n); \text{ for } 2 \leq n \leq N \quad (3.4)$$

3.4.1.2 Performance Bounds

The second major requirement for a scheduling discipline is that it should allow a network provider to guarantee per-connection performance bounds, restricted only by the conservation law. Performance bounds can be expressed either deterministic or statically bounds that are described via several common performance parameters.

Deterministic and statistical bounds

While a deterministic bound holds for every packet sent on a connection, a statistical bound is a probabilistic bound on performance. For example, a deterministic bound on a end-to-end delay of 5 s means that every packet sent on a connection has delay smaller than 5 s. A statistical bound of 5 s with a parameter of 0.97 indicates that the probability, for which a packet has a delay greater than 5 s, is smaller than 0.03.

Common performance parameters

Four common performance parameters are widely used in literatures: bandwidth, delay, delay-jitter, and loss.

- A *bandwidth bound* defines a requirement that a connection receives at least a minimum bandwidth from the network.
- A *delay bound* can be a deterministic or statistical bound on some delay parameters such as worst-case delay or mean delay. While the worst-case delay is the largest delay suffered by a packet on a connection, the average delay is the delay over all packets of every connection in the system. Because the true average delay is impossible to define precisely, the mean delay is often used. It is measured over all packets sent on a connection.
- A *delay-jitter bound* describes a requirement that the difference between the largest and smallest delay received by packets on a connection must be less than some bound.
- A *packet loss bound* expresses a constraint that the packet loss rate on a connection must smaller than some bound.

3.4.2 Classification of scheduling disciplines

There are two basic ways for classifying the scheduling disciplines: work-conserving vs. non-work-conserving and scheduling for elastic flows vs. scheduling for real-time flows. These classifications will be discussed in this paragraph.

3.4.2.1 Work-conserving vs. non-work-conserving

Scheduling disciplines can be classified into two fundamental classes – the work-conserving scheduling and the non-work-conserving scheduling. A scheduling discipline is called work conserving if it is never idle when there are packets waiting in the queue to be served. In contrast, non-work-conserving may be idle even when any of the queues have packets waiting for services. The difference to a work-conserving scheduling is that a non-work-conserving does

not serve a packet as soon as it arrives, it first wait until this packet is eligible and then sends the packet. The reason for the idle time at the non-work-conserving scheduling is to reduce the burstiness of traffic entering the network.

A simplest work-conserving scheduling is the First-In-First-Out (FIFO), which transmits incoming packets in the order of their arrive at the output queue. The disadvantage of FIFO is that the scheduling cannot provide the isolation between different connections and cannot differentiate among these connections. Thus, this scheduling cannot assign some connections lower mean delay than other connections. Although several scheduling disciplines can achieve this objective, the conservation law [Kle-1975b] states that if a scheduling is work-conserving then the sum of mean queue delay received by a set of multiplexed connections, weighted by their share of link's load is independent of the scheduling disciplines. This conservation law is given by the following equations:

$$\sum_{i=1}^N \rho_i q_i = \text{constant}; \quad (3.5)$$

$$\rho_i = \lambda_i x_i; \quad (3.6)$$

where

ρ_i = the mean utilization of packets belonging to connection i

λ_i = Mean arrival rate of packets belonging to connection i

x_i = Mean service time of packets belonging to connection i

q_i = Mean wait time of a packet belonging to connection i at the scheduler

N = Number of connections

Since the right-hand side of the equation (3.5) is independently of the scheduling discipline, a connection can receive lower delay from a work-conserving scheduling only at the expense of another connection.

3.4.2.2 Scheduling for elastic flows vs. real-time flows

Two basic types of Internet applications are elastic applications and real-time multimedia applications. This makes the case that the scheduling disciplines for elastic and real-time flows are different. For elastic flows, the scheduling disciplines should provide a max-min fair allocation described above. For real-time flows, the scheduling disciplines should provide performance guarantees for each flow or for aggregated flows. To support QoS for real-time

and elastic flows, a scheduling discipline must achieve several goals [San-2002]:

- *Sharing bandwidth and providing fairness to competing flows.* If there are multiple competing elastic flows, the scheduler is required to perform fair allocation of the resources.
- *Meeting delay guarantees and reducing jitter.* A scheduler can allocate different mean delay to different flows by its choice of service order. Thus, the service order has impact on delay suffered by packets waiting in the queue. And, a scheduler is capable of guaranteeing that the delay will be below a minimal level.
- *Meeting loss guarantees.* The scheduler can allocate different loss rates to different flows by giving them more or fewer buffer. If the buffer is of limited size, packets will be dropped. Thus, the service order has impact on packet losses and a scheduler is capable to guarantee that the delay will be below a minimal level.
- *Meeting bandwidth guarantees.* A scheduler can allocate different bandwidth to packets from a flow by serving a certain number of packets from this flow within a time interval. Thus, a scheduler is capable to guarantee that a flow will get a minimal amount of bandwidth within a time interval.

3.4.3 First-In-First-Out (FIFO)

The FIFO scheduler transmits incoming packets in order of their arrival at the queue. This means that the packets, which first arrive, will be first transmitted (figure 3-19). Its working principle is very simple. Packets from all flows are buffered into a common queue and the FIFO scheduler serves the packet from the header of the queue. Packets that arrive to a full queue are dropped.

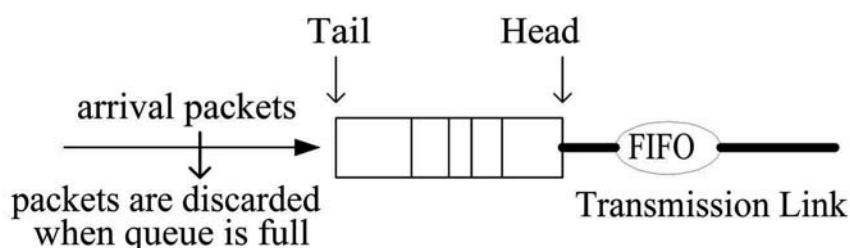


Figure 3-19: FIFO scheduling

The FIFO is a very simple scheduling discipline and is implemented in most conventional Internet router. Its advantage is that it is a useful and simple scheduling discipline and it may be used in conjunction with other advanced scheduling disciplines. The main disadvantage of FIFO is that it does not

support flow isolation, and without flow isolation it is very difficult to guarantee delay bound or bandwidth for specific flows. Because of this, FIFO has a major limitation for supporting multimedia applications. If different services are required for different flows, multiple queues are needed to separate the flows.

3.4.4 Priority Scheduling

Priority scheduling refers to a class of scheduling disciplines that provide differential treatment to flows by using of multiple queues with associated priorities. A priority scheduling maintains multiple queues with different priority level (figure 3-20). Depending on priorities defined in the packet headers, packets are queued into queues. If there are packets waiting in both higher and lower queue, the scheduler serves packets from the higher queue before it serves the lower one. For example, the packets with priority 1 in figure 3-20 are always served first. Packets within a priority queue are usually served with FIFO discipline. Packets of priority of i are served only if the queues 1 through $(i-1)$ are empty. Thus, the flow with highest priority packets has a least delay, a highest throughput and a lowest loss.

Nevertheless, this priority scheduling is not max-min fair share because it has the potential of starvation of lower priority classes, i.e. the server will never be able to serve the packets of lower priorities because it always busy serving the packets of higher priorities. However, priority scheduling is very simple to implement as it needs to maintain only a few states per queue.

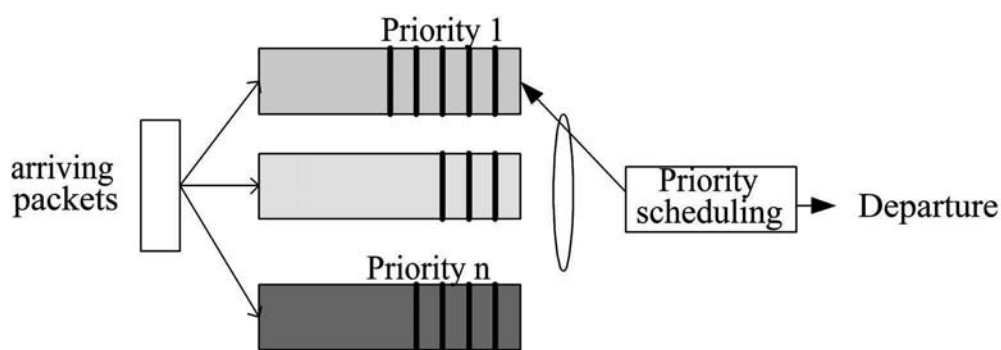


Figure 3-20: Priority scheduling

3.4.5 Generalized Processor Sharing

Generalized processor sharing (GPS) [PG-1994, PG-1994] is an ideal work-conserving scheduling discipline in that it provides an exact max-min fair share. GPS is fair in the sense that it allocates the whole outgoing capacity to all backlogged sessions in proportion to their minimal bandwidth requirements.

GPS assumes that packets of each flow are kept in a separate logical queue. GPS serves an infinitesimal small amount of data from each queue, so that it can visit every non-empty queue at least once within a finite time interval.

Assuming that there are K active flows with equal weights, then the GPS server will allocate each of them a $(1/K)$ th share of the available bandwidth which is their max-min fair share, because the GPS serves an infinitesimal amount of data from each flow in turn. If a queue is empty, the scheduler skips to the next non-empty queue, and the unused resource is distributed among competing flows. Flows can be associated with service weights, and a GPS server can serve data from non-empty queues in proportion of its weight whenever they have packets waiting in the queue. Thus, GPS is also capable to achieve max-min weighted fair share as well. In GPS, a flow is called a backlogged flow whenever it has packets waiting in the queue. Assume that there are N flows being served by a GPS server. Let $r(i)$ the minimum service rate allocated for i -th flow. The associated admission policy should guarantee that

$$\sum_{i=1}^N r(i) \leq c \quad (3.7)$$

where c is the capacity of the outgoing link.

Let $B(t)$ denote the set of backlogged flows at time t . According to [PG-1993], the backlogged flow i will get a service rate $R(i, t)$ such that

$$R(i, t) = \frac{c \cdot r(i)}{\sum_{j \in B(t)} r(j)} \quad (3.8)$$

Furthermore, because of the work-conserving property, the following condition holds at any time during the system busy period

$$\sum_{i=1}^N R(i, t) = c \quad (3.9)$$

The service rate allocation of GPS is described as follows. Let $A(i, t_1, t_2)$ the amount of packet arrivals of connection i in the time interval $[t_1, t_2]$, $S(i, t_1, t_2)$ the amount of service received by connection i in the same time interval, and $Q(i, t_1, t_2)$ the amount of connection i traffic queued in the server at time t_2 is calculated via the following equation

$$Q(i, t) = A(i, t_1, t_2) - S(i, t_1, t_2) \quad (3.10)$$

The fairness index of backlogged connection i can be defined as $S(i, t1, t2)/r(i)$. During any time interval $(t1, t2)$, for any backlogged connection i and j , the scheduler is said to be perfectly fair if and only if it satisfies

$$\frac{S(i, t1, t2)}{r(i)} = \frac{S(j, t1, t2)}{r(j)} \quad (3.11)$$

The GPS scheduling is perfectly fair. Thus, by definition, the GPS achieves the max-min fair share.

GPS is an ideal scheduling discipline that achieves max-min fair share. However, GPS is not able to implement, since serving an infinitesimal amount of data is not possible. Some GPS variations, which can be implemented in a real system, are round robin, weighted round robin and deficit round robin. These scheduling disciplines will be described in the following.

3.4.6 Round-Robin

A simple implementation of the GPS is the round robin scheduling, which serves a packet from each nonempty queue instead of an infinitesimal amount of data by the GPS. To solve the fairness and isolation problem by a single FIFO scheduling, the round robin scheduler maintains one queue for each flow. The scheduler serves packets from each flow in a round robin fashion – it takes one packet from each nonempty queue in turn and skips empty queue over. A misbehaving user overflows its own queue, and the other flows are unaffected. Thus, round robin can provide the protection between flows.

The round robin tries to treat all flows equally and provide each of them an equal share of link capacity. It approximates GPS reasonably and provides a fair allocation of the bandwidth when all flows have the same packet size, such as in ATM network. If flows have variable packet sizes, such as packets in the Internet, round robin does not provide the max-min fair share.

3.4.7 Weighted Round Robin

The Weighted Round Robin (WRR) scheduling is a simple modification of round robin. Instead of serving one packet from nonempty queue per turn, WRR serves n packets. Whereby, n is the weight assigned to a flow and corresponds to the fraction of link bandwidth this flow is going to receive. The number of packets to be served in turn by each flow is calculated from this weight and the available link capacity.

Like the round robin scheduling, the WRR provides the max-min fair share if each flow has a fixed packet size. However, WRR has a problem in providing

bandwidth guarantees when flows have variable packet sizes. In this case, the flow with a large packet size will receive more bandwidth than the allocated weight. In order to solve this problem, the WRR needs to know the mean packet size of all sources a priori. And if a source cannot predict its mean packet size, a WRR server cannot allocate bandwidth fairly.

3.4.8 Deficit Round Robin

Deficit round robin (DRR) is a modification of Weighted Round Robin to allow the scheduling to provide max-min fair share among competed flows with variable packet sizes without knowing the mean packet size of each flow in advance.

DRR maintains two variables for each queue. One variable is called quantum that indicates the maximum packet size for this queue. Another variable is called deficit count that defines the maximum size of a packet can be served at this time. Deficit count is initialised as zero. Like RR, the DRR scheduling starts serving each queue that has a packet to be transmitted. At each non empty queue, if the packet size is less than or equal to the quantum, the scheduler serves the packet and the deficit count is decremented by the quantum size. If the packet size is bigger than quantum size, the packet must wait for another round, and its deficit count is incremented by the quantum size. If a queue is empty, the deficit count for this queue is set equal to zero. Figure 3-21 describes the DRR algorithm.

```

for all queue I in a round { /* Initialization*/
    deficit_count[I]:=0;
    quantum[I]:=given_value;/*maximum packet size for queue*/
}
for all queue I in a round
if (there is a packet to be served) then
    begin
        deficit_counter[I]:= deficit_counter[I]+quantum[I];
        if (packet_size≤deficit_counter[i]) then
            begin
                serve the packet;
                deficit_counter[I]:=deficit_counter[I]-
                packet_length;
            end;
        else go to next queue in the round;
    end;
else deficit_counter[I]:=0;

```

Figure 3-21: Deficit round robin algorithm

3.4.9 Weighted Fair Queuing scheduling

Another approximation of the GPS scheduling is the weighted fair queuing (WFQ) scheduling that does not make the GPS's infinitesimal packet size assumption, and, with variable size packets, it does not need to know a connection's mean packet size in advance [San-2002]. The key idea of WFQ scheduling is that each packet is tagged on the ingress router with a value called virtual finish time, which theoretically identifies the time the last bit of packet should be transmitted if a GPS scheduler was used. Each time the scheduler is available to send a packet, the scheduler selects the packet with the lowest finish time. It is important to know that the finish time value is only a service tag indicating the relative order in which the packet is to be served, and has nothing to do with the actual time at which the packet is served.

3.4.9.1 Finish time computation

The virtual finish time $F(i, k, t)$ the router would have finish sending k th packet on the flow i at time t is calculated via the following equation:

$$F(i, k, t) = \max\{F(i, k-1, t), R(t)\} + \frac{P(i, k, t)}{w(i)} \quad (3.12)$$

The parameters in this equation are as follows:

- $F(i, k-1, t)$: the virtual finish time for $(k-1)$ th packet on the flow i ,
- $R(t)$: the round number is defined as the number of rounds a bit-by-bit RR scheduler has completed at a given time,
- $P(i, k, t)$: the time required to transmit i th packet from the flow i ,
- $w(t)$: the weight of the flow i .

Thus, the computation of the finish number depends on the round number. The time taken by each round depends on the actual number of active flows: the more flows served in a round, the longer the round takes. A flow is called an active one if the largest finish number either in its queue or last round number served from its queue is larger than the current round number.

3.4.9.2 Implementation of WFQ scheduling

Each implementation of a WFQ scheduling includes three following steps:

1. *Finish number calculation.* For each arriving packet of a flow, the scheduler updates its current round number and computes the finish number for this packet.

2. *Queuing the packets according to finish number.* Within each flow, a packet is buffered according to its finish number, so that this packet will be served in order of its finish number.
3. *WRR scheduling.* The WFR serves the packets in each queue according to its weight.

•

3.4.9.3 Delay bound by WFQ scheduling

Because the WFQ scheduler uses the WRR scheduling, WFQ gives a bandwidth bound to each flow. Let packets of a flow i passing through K schedulers, where the k -th scheduler ($1 \leq k \leq K$) has a limited rate $c(k)$ at the outgoing link and assigns a weight of $w(i, k)$ for connection i . Based on WFQ scheduling, the flow i is guaranteed to receive a fraction of bandwidth that equals $w(i)/\sum w(j)$, where the denominator represents the weight sum of all flows that have packets waiting for transmission. At the WFQ scheduler k , the flow i will always receive a service rate $R(i, k)$ of

$$R(i, k) = \frac{c(k) \cdot w(i, k)}{\sum w(j, k)} \quad (3.13)$$

If packets of the flow i are constrained by a token bucket $(r(i), q(i))$, where $r(i)$ is the token bucket rate and $q(i)$ is the bucket size, then the last packet completed with the service at the scheduler k will suffer a maximum delay of $d_{max}(i, k)$ given by equation below.

$$d_{max}(i, k) = \frac{r(i)}{R(i, k)} \quad (3.14)$$

Let $R(i)$ be the smallest of the $R(i, k)$ s over all k . If the largest packet allowed on the connection i has a size of $p_{max}(i)$ and the largest packet allowed in the network has a size of p_{max} , then independent of the behaviour of the other flows, the worst case end-to-end queuing and transmission delay $D(i)$ for packets belonging to the flow i through K schedulers is bounded by [GV-1995]:

$$D(i) \leq \frac{r(i)}{R(i)} + \sum_{k=1}^{K-1} \frac{p_{max}(i)}{R(i, k)} + \sum_{k=1}^K \frac{p_{max}}{c(k)} \quad (3.15)$$

WFQ scheduling has three advantages. First, because it approximates GPS, it provides protection between different connections. Second, because it serves packets by using of the so-called finish time, it supports packets with variable sizes. Third, under certain assumptions, a connection can obtain a worst-case end-to-end delay, which is independent on the number of hops it traverses and

on the behaviour of other flows. Because of these advantages, WFQ is used to schedule real-time multimedia flows. However, WFQ scheduling requires per-flow (or per aggregate) state, which can be expensive for schedulers that serve large numbers of flows. Furthermore, WFQ requires a difficult algorithm for updating its round number. Moreover, it requires explicit sorting of the packets in the output queue according to the finish time, which requires time and complex hardware or software.

Despite these problems, WFQ scheduling is implemented in a lot of router and switch products, such as routers from CISCO and ATM switches from FORE systems.

3.5 Congestion Control

Congestion occurs when traffic demand is greater than actual available network resources. By network resources, we mean the bandwidth of links, buffer space of routers, and processing speed at the intermediate routers and at the end systems. The situation, in which the network performance degrades, is called congestion. Symptoms of congestions are e.g. end-to-end delays increase rapidly, PDUs are dropped, throughput decreases, buffers overflow and deadlock. Congestion can occur because of several reasons. Overload on an output line leads a queue to build up and if there is insufficient memory to hold all packets on this line, traffic will be lost. Increasing the router buffer space could not reduce congestion, because by the time packets get to front of the queues, they have already timeout at the transport layer protocols (e.g. by the TCP), and thus duplicates have been sent. This increases the load on all way to the destination and therefore leads to more congestion. Slow processors, low-bandwidth lines can also cause the congestion. Upgrading the lines or/and changing the processors often helps a little, but frequently shifts the bottleneck [Jai-1999, Tan-2002]. That is the reason, why do we need the congestion control. Congestion control deals with mechanisms that enable a data source to match its sending rate to the currently available service rate at the receiver(s) and in the network.

Flow control deals with the techniques that enable a data source to match its sending rate to the current available service rate at a receiver. In particular, the main goal of the flow control is to keep the sender from overrunning receiver in which it throttle the sender into sending no faster than the receiver can handle traffic. The difference between flow control and congestion control is described in [Tan-2002] as follows. Congestion control is global issue, involving the behaviour of all end systems and intermediate systems and the store-and-forwarding processing within the routers. It has to do with making sure that the

subnet is able to carry the offered traffic to the receiver. In contrast, the flow control has to make sure that a fast sender cannot transmit data faster than the receiver can absorb it. It relates to the point-to-point traffic between a given sender and a given receiver and always involves direct feedbacks from the receiver to the sender to tell the sender how fast it can send the data. But in my point of view, flow control is a mechanism of the congestion control relating to sender and receiver, and thus we do not separate between the flow control and the congestion control.

This section discusses the fundamental congestion control mechanisms that can be used for controlling the congestion by unicast elastic applications, by unicast real-time applications and by multicast applications. Also, these mechanisms can be used in several layers of the protocol stack.

3.5.1 Classification of Congestion Control

There are several ways to classify the congestion control mechanisms. In our work, we distinguish among congestion control approaches by using three following aspects: (1) whether the congestion control works with feedback or without feedback; (2) where the congestion control is addressed; (3) whether the congestion control is window-based or rate-based.

3.5.1.1 Feedback-based vs. reservation-based Congestion Control

A feedback-based congestion control is also called a closed-loop congestion control in [Kur-2004, Tan-2002]. It deals with the mechanisms that enable the network to control a set of system parameters relating to the congestion and to inform a source that its service rate has changed. By a feedback-based approach, a delicate handshaking algorithm is necessary needed between the source and the sub-network. In such approaches, some system parameters (e.g. packet loss, buffer size) are monitored and passed back (as explicit feedback or implicit feedback) to portions of subnets that can take actions to reduce or to avoid the congestion.

In an explicit feedback schema, the network devices explicitly convey these system parameters to the source, for example an acknowledgement is an explicit feedback. In an implicit feedback schema, a source infers a change in its service rate by measuring its current local parameters. For example a retransmission timeout at a source is an implicit feedback. TCP congestion control is an example of a feedback-based congestion control.

A congestion control mechanism, which does not need feedback, is called an open-loop congestion control or a reservation-based congestion control. In comparison with feedback-based approaches, a reservation-based congestion

control is much simpler to implement. For example, if there is a data rate negotiated between the sender and the network nodes, the source can send data at this rate without any data loss regardless of the traffic of other sources. Its main idea is to reservation enough resource on the network to prevent the congestion. The main principle of such congestion control approaches can be summarized as follows:

1. A source describes the expected characteristics of its traffic to the network via a set of traffic parameters.
2. During the connection setup, the network reserves enough resource (e.g. bandwidth, buffer) corresponding to the traffic parameters described by the source.
3. During the data transmission, if the source shapes and polices its traffic to match its traffic description, and thus if the network overloads, congestion will be avoided.

However, this open-loop congestion control has several disadvantages. First of all, it is difficult to choose a right set of parameters to describe the source traffic, especially in the Internet. Furthermore, the resource reservations (step 2.) are made without regard to the current network state while transmitting the data.

3.5.1.2 Host-based vs. network-based Congestion Control

A host-based congestion control operates at transport layer of the end system in which the end hosts observe e.g. how many packets are successfully transmitted through the network and thus can adjust their sending rate accordingly. The network layer provides no explicit support to the transport layer for congestion control. The presence of congestion must be inferred by the end systems through observed network behaviours, such as packet loss or delay.

By network-based approaches, the congestion control operates at the routers and at the end hosts. By such approaches, the routers observe the traffic situation e.g. by monitoring some system parameters and then provide explicit feedback to the sender regarding the congestion state in the network. The feedback may be in form of a simple bit indicating congestion at a link. Based on this feedback information, the sender will take actions according to the network behaviour to reduce the congestion. Furthermore, routers also can remark the packets in order to inform the source about the congestion situation. And the source takes actions according to the remarked packets and to the lost packets. Explicit Congestion Notification (ECN) is an example of a network-based congestion control.

3.5.1.3 Window-based vs. rate-based Congestion Control

Basically, a congestion control mechanism can work by either directly limiting their data population in the network or limiting their transmission rates. This yields window-based and rate based congestion control.

- *Window-based congestion control [CQ-2001]*: The receiver keeps a buffer of size W of PDUs, where W is called the window size. In addition, there are maximal W credits. Each PDU waiting in the buffer of the receiver must hold a credit. When a PDU is dispatched from the receiver, it relinquishes the credit, and this credit is sent back to the sender instantaneously. Meanwhile, the sender only transmits data when it gets a credit. Otherwise it stops sending until there is a credit released from the receiver. Credits that are not currently held by data are either stored at the sender or on the way back to the sender, so that the buffer of the receiver will be never overflow.
- *Rate-based congestion control*: Window-based mechanisms control the sending rate by adjusting the size of transmission window. Therefore, the sending rate depends on the window size. In contrast, in a rate-based congestion control, the sending rate does not depend on the window size. That means, losses and retransmissions do not directly affect the rate at which data is transmitted into the network. The basic principle of rate-based mechanisms is as follows. The source and destination negotiate a transfer rate expressed as a set of parameters (e.g. burst size and burst rate, RTT, loss rate) measured during the data transmission, so that the source needs to control the rate only at the granularity of a burst. PDUs are placed in a transmission queue drained at the negotiated rate. If PDUs are lost, retransmitted PDUs are also placed in the same transmission queue so that the source transmission rate is limited to the negotiated rate which is independent of the loss rate.

Window-based and rate-based congestion control mechanisms can be implemented at the application, transport, network or data link layer of protocol stack. The choice of layer depends on the situation. But the most well-known protocols implementing the window-based congestion control are TCP and STCP, which operate at the transport layer.

3.5.2 TCP Congestion Control

TCP implements host-based, feedback-based and window-based congestion control. Its main idea is that the TCP source sends segments, then observes the

loss events and reacts to these events. By this way, TCP source attempts to determine how much capacity is actually available in the network.

The TCP congestion control may include four algorithms [RFC 2581, RFC 2018, RFC 3782, RFC 2001]: slow start, congestion avoidance, fast retransmit, and fast recovery. These algorithms will be discussed in this paragraph. In order to implement these congestion control algorithms, four main variables are managed for each TCP connection:

- *Congestion window (cwnd)*. The congestion window imposes an additional constraint on how much traffic a host can send into a TCP connection. cwnd is initial set equal to one (or two, three) the maximum segment size (MSS) of TCP segments.
- *Receiver's advertised window (rwnd)*. This variable indicates the value in the field "window" of the TCP header. The value of this variable tells the TCP sender, how many more bytes the TCP receiver may accept.
- *Slow start threshold (ssthresh)*. This variable defines the threshold between the slow start and the congestion avoidance phase. It effects how the congestion window grows.
- *Sending window (win) at the TCP sender*. The value of this parameter is defined as the minimum of the congestion window and the receiver's advertised window.

$$win = \min(cwnd, rwnd)$$

The basic principle of the TCP congestion control is described as follows. After finishing the TCP connection establishment, TCP first starts probing for usable bandwidth. Ideally, it transmits data as fast as possible without loss. That means TCP increases the congestion window until the loss occurs. When loss occurs, TCP decreases the congestion window, and then again begins with the increasing the congestion window until the loss occurs. The slow start threshold is used to define how the congestion window size can grow. Namely, when the congestion window cwnd is below the threshold, the congestion window grows exponentially. Otherwise, the congestion window grows linear. Whenever if there is a timeout event, the threshold is set equal to one-half of the current congestion window and the congestion window is set equal to one maximum segment size. Important by this process is that the TCP sender changes its sending rate by modifying the sending window size ($win = \min(cwnd, rwnd)$).

3.5.2.1 Slow Start and Congestion avoidance

Slow start and congestion avoidance algorithm are used to control the amount of outstanding data being injected into the network [RFC 2581]. The TCP uses the

slow start threshold to determine whether the slow start or congestion avoidance algorithm is currently used.

Since TCP begins to transmit data into the network with unknown conditions, it needs to slowly probe the network to determine the available capacity, and thus to estimate how much TCP can send data in order to avoid the network congestion. The slow start algorithm is used for this purpose at the beginning of transfer, or after repairing loss detected by the retransmission timer.

3.5.2.1.1 Slow Start Algorithm

At the beginning of the data transmission, TCP sets the initial value of the congestion window equal to one (or two) maximum segment size (MSS) of TCP segments. TCP stays in slow start if there is no loss event and if the congestion window *cwnd* is below the slow start threshold. For each acknowledged segment, the congestion window is increased by one MSS. Thus, the congestion window is exponential increased per Round Trip Time (RTT). The slow start phase terminates if the congestion window exceeds the slow start threshold or when congestion is observed. If a timeout event occurs, the slow start threshold is set equal to one-half of the congestion window and the congestion window is set equal to the initial value of the congestion window; the TCP then performs the slow start algorithm.

3.5.2.1.2 Congestion Avoidance

TCP performs the congestion avoidance algorithm if there is no loss event and the congestion window is above the slow start threshold. During congestion avoidance, *cwnd* is incremented by 1 full-sized segment per round-trip time (RTT). Congestion avoidance continues to work until TCP observes the loss event via timeout. Figure 3-22 illustrates the slow start and congestion avoidance algorithm in pseudo code and figure 3-23 shows the *cwnd* behaviour during slow start, congestion avoidance and timeout.

```

1  /*Initial */
2  cwnd := 1*MSS;
3  ssthresh := infinite;
4  /* Slow Start algorithm */
5  until (loss_event or cwnd < ssthresh)
6  begin
7      for each segment acknowledged do
8          cwnd := cwnd+1*MSS;
9  end
10 /*Congestion avoidance algorithm*/
11 if (no loss event and cwnd 3 ssthresh) then

```



```

12 begin
13   for every cwnd segments acknowledged do
14     cwnd:=cwnd+1*MSS;
15   end
16 /*do slow start again if event timeout occurs*/
17 if (timeout) then
18   begin
19     ssthresh := max(cwnd/2, 2*MSS);
20     cwnd := 1 MSS;
21     perform slow start algorithm in lines 5-9
22   end

```

Figure 3-22: Pseudo code for slow start, congestion avoidance and loss event

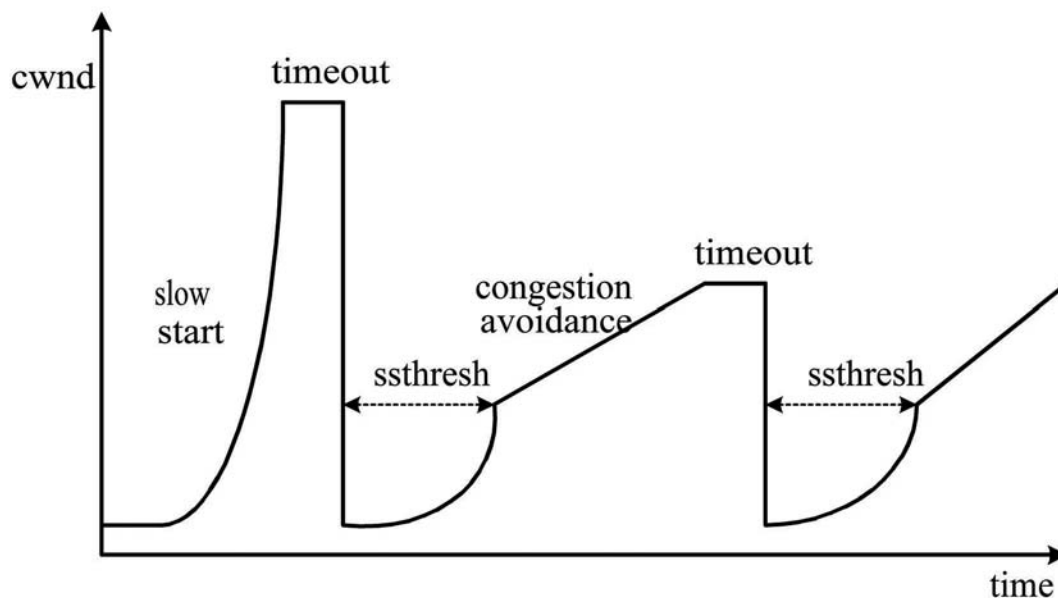


Figure 3-23: cwnd behaviour in dependent on the ssthresh and timeout event

3.5.2.1.3 Disadvantages of Slow Start and Congestion Avoidance

The main problems of the slow start and congestion avoidance algorithm can be summarized in the following. TCP detects a segment loss only via a timeout event and resends this lost segment after a timeout interval. This can cause the end to end delay to increase because of the waiting for the timeout event. Moreover, TCP reduces the congestion window cwnd to 1 MSS by a timeout event, even for only one lost segment, and begins the slow start algorithm. This behaviour leads to rapidly increase the TCP throughput increases rapidly and thus TCP has a lower throughput in the case of a moderate congestion.

Solutions for these problems are the fast retransmit and fast recovery algorithm that will be described in the next following paragraphs.

3.5.2.2 Fast Retransmit

In order to solve the slow start and congestion avoidance problems mentioned above, the TCP congestion control needs a mechanism to detect packet losses without waiting for a timeout event. This is performed via the fast retransmit algorithm. The idea of this algorithm is that a TCP receiver sends an immediately duplicate ACK when an out-of-order segment arrives [RFC 2581]. The purpose of this duplicate ACK is to let the TCP sender know that a segment was received out-of-order and to tell it which sequence number is expected.

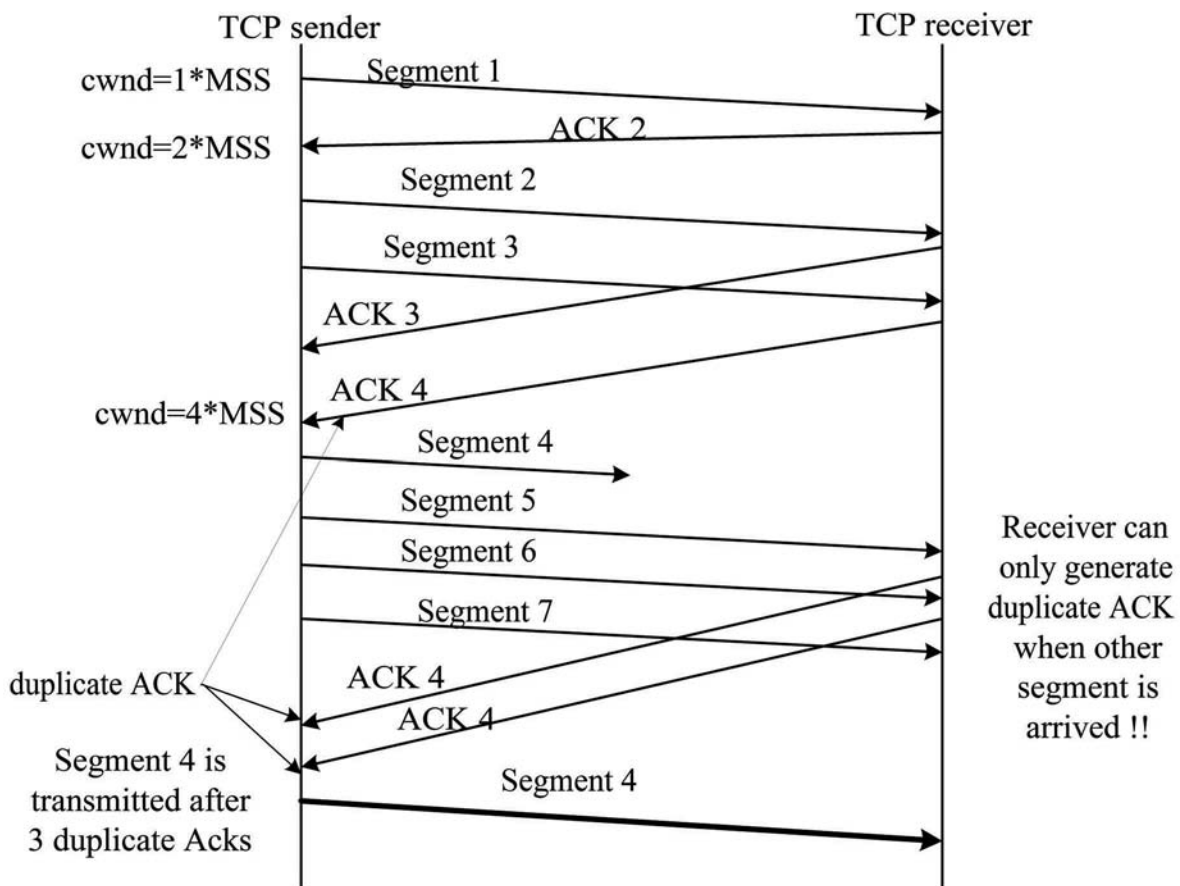


Figure 3-24: Example of the fast retransmit algorithm

The fast retransmit algorithm functions as follows. When the TCP sender sees duplicate ACKs, it assumes that something went wrong. Duplicate ACKs mean the third, fourth, etc. transmission of the same acknowledgement number. If three or more duplicate ACKs are received, it is a strong indication that a segment has been lost. Thus, the TCP sender sets the slow start threshold (ssthresh) equal to one-half of the congestion window (cwnd) and the cwnd equal to one MSS, and then immediately retransmits the missing segment.

without waiting for the retransmission timer to expire. After sending the missing segment, TCP returns to the slow start phase. The sequence diagram in the figure 3-24 illustrates an example of the fast retransmit algorithm described above. Figure 3-25 demonstrates the behaviour of the congestion window of a TCP Tahoe connection by duplicate ACK events. It is to note that TCP Tahoe only implements slow start, congestion avoidance and fast retransmit algorithm. This figure shows that at the 2nd second, the congestion window reduces to one MSS even by loss of one TCP segment.

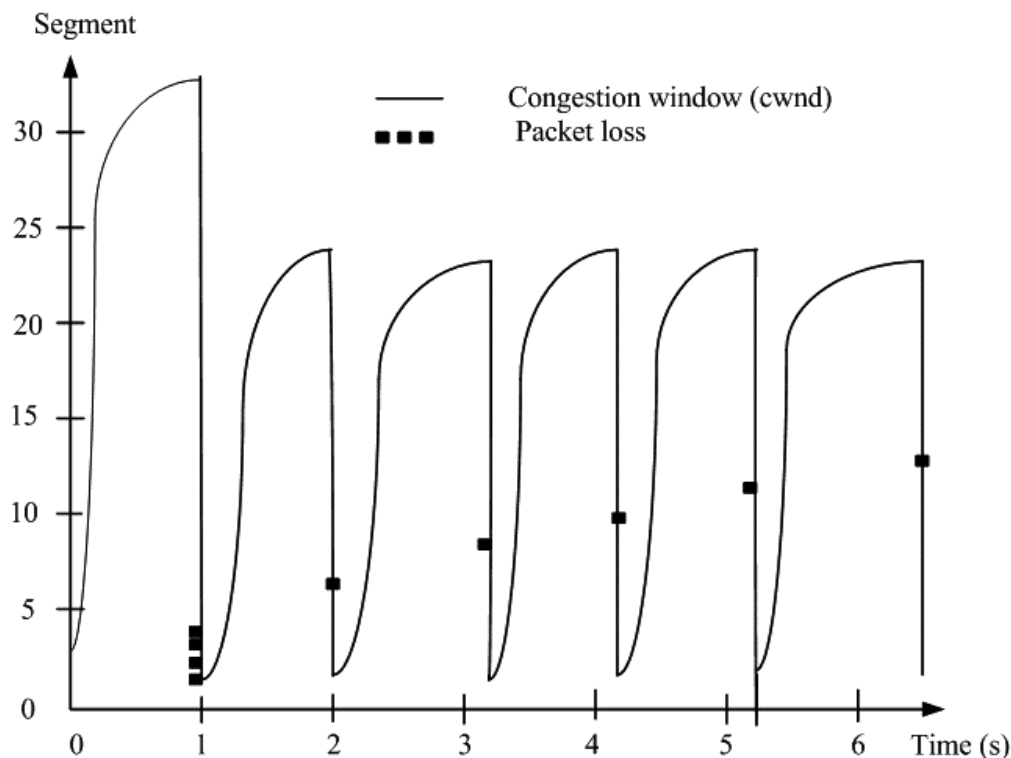


Figure 3-25: TCP congestion window by using only fast retransmit

The main problem by the fast retransmit algorithm is that TCP performs again the slow start algorithm after sending the missing segment. This leads to rapidly decrease the TCP throughput. Since the TCP receiver can only generate duplicate ACKs when another segment is arrived. This segment has left the network and is in the receiver's buffer. This means, there is still data flowing between two ends and TCP does not need to reduce the sending rate rapidly. The solution for this fast retransmit problem is the fast recovery.

3.5.2.3 Fast Recovery

The key idea of the fast recovery algorithm is that, after the fast retransmit resends the missing segment, TCP performs the congestion avoidance algorithm and not the slow start. The reason for not performing the slow start algorithm is

that the receipt of duplicate ACKs not only indicates a segment has been lost, but also tells that segments are most likely leaving the network.

Fast retransmit and fast recovery can be implemented together and work as follows [RFC 2581]:

1. When the third duplicate ACK is received, TCP sets $ssthresh$ to $\max(cwnd/2, 2MSS)$.
2. $ssthresh := \max(cwnd/2, 2MSS)$
3. TCP retransmits the lost segment and set $cwnd$ to $(ssthresh+3*MSS)$.
4. $cwnd := ssthresh+3*MSS$
5. For each additional duplicate ACK received, TCP increments $cwnd$ by one MSS
6. $cwnd := cwnd + MSS$
7. TCP transmits a segment, if allowed by the new value of $cwnd$ and the $rwnd$.
8. When the next ACK arrives that acknowledges the new data, TCP sets $cwnd$ equal to $ssthresh$. This terminates the fast recovery and entering the linear growth phase of $cwnd$ (the congestion avoidance)

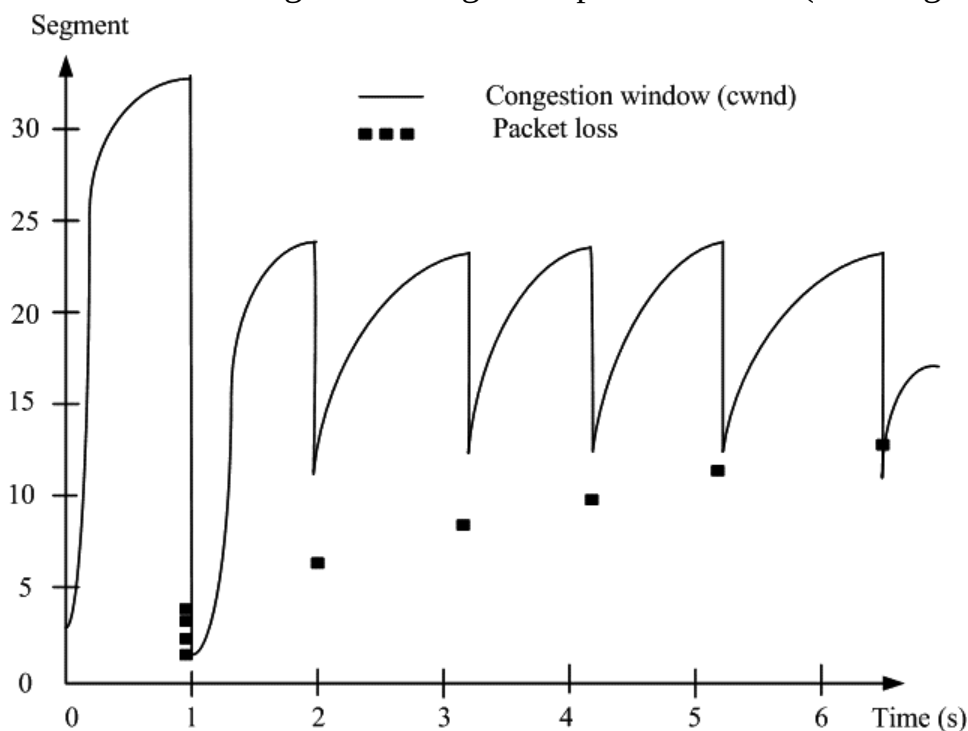


Figure 3-26: TCP congestion window by using fast retransmit and fast recovery

Figure 3-26 illustrates behaviour of congestion windows (simulation with NS2) of a TCP Reno connection that implements the fast retransmit and fast recovery algorithm. It is very clear to see in this figure that TCP stays in the congestion avoidance phase when receiving duplicate ACKs. Thus, the TCP

throughput by moderate congestion with the fast retransmit and with fast recovery algorithm is higher than with only the fast retransmit algorithm.

3.5.3 Explicit Congestion Notification

Explicit congestion notification (ECN) developed from IETF is a congestion control approach, which works on sending explicit signalling of congestions to the sender by marking the packets instead of dropping them [BCC-1998, RFB-2001, SWE-2003].

The basis principle of ECN is that a router, which experiences the congestion, sets a bit in IP header of the incoming packets on their way from the sender host to the receiver host. When these marked packets reach the receiver host, this host responds by setting a bit in the TCP header in next outgoing acknowledgements. When these acknowledgements arrive at the sender host, TCP at the sender host performs the fast retransmit and fast recovery algorithm in responding to the congestion control. The sender host also sets other bits in TCP header in the next outgoing acknowledgements to inform the receiver host that it has reacted to the receiver's congestion notification. The ECN mechanism consists of two parts: (1) ECN at the router and (2) ECN at the end host. These parts are described in the following paragraphs.

3.5.3.1 ECN at Routers

The ECN at the router includes making decision for each arrival packet based on average queue length and on the packet itself, and packet marking. In contrast to RED that drops incoming packets based on the average queue length, an ECN-capable router probabilistically marks incoming packets when average queue length $avq(t)$ is between a predefined minimum threshold and a maximum threshold, and marks all packets when the average queue length exceeds the maximum threshold. The average queue length and the packet marking probability p are calculated by using the equations 3.18-3.19 described in the RED section. The ECN's packet marking algorithm is shown in figure 3-27.

```

For each arriving packet at time  $t$ :
IF  $avq(t) \leq min\_threshold$  THEN packet is accepted
    (No congestion, marking probability is equal zero)
IF  $avq(t) > max\_threshold$  THEN packet is marked
    (high congestion, marking probability is equal 1)
IF  $max\_threshold \geq avq(t) \geq min\_threshold$  THEN packet is marked
    with a probability  $p$ 
  
```

Figure 3-27: ECN packet marking decision at the routers

Marking packets at the routers is performed through two bits in the IP packet header - the ECN capable transport (ECT) bit and the congested experienced (CE) bit. In the IPv4 header, these bits are the 6th bit and 7th bit in the ToS field, but in the IPv6 header, they are the 10th and 11th bit of the traffic class field. While the ECT bit is used by end systems to indicate whether they are capable of ECN, the CE bit is used by the routers to mark the packets on their way from the sender host to the receiver host if the routers are experiencing congestion. The routers are required to mark the CE bit only when the ECT bit is set (figure 3-28). Otherwise they may drop packets.

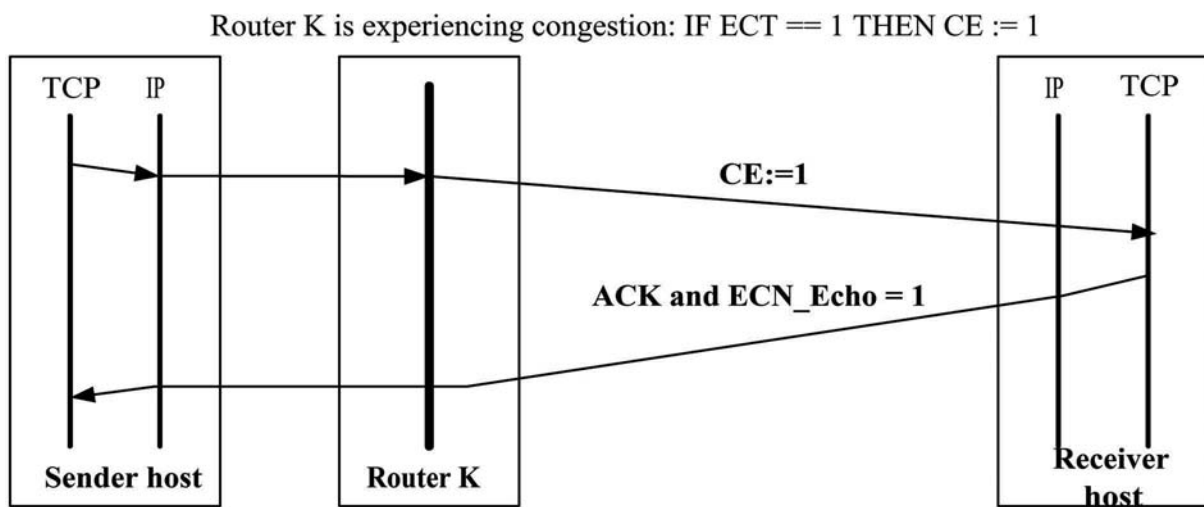


Figure 3-28: Packet marking at the router and at the receiver host

3.5.3.2 ECN at End Hosts

When packets with the marked CE bit reach the receiver host, the receiver responds by setting the ECN-echo bit in the next outgoing acknowledgement for this packet. Whereby the ECN-echo bit (ECE) is the 9th bit in the reserved field of the TCP header. The receiver will continue to set the ECE bit in subsequent TCP acknowledgements as long as congestion exists.

Upon the TCP sender receives the acknowledgement carrying the ECN echo, the TCP sender performs fast retransmit algorithm in which the congestion window (cwnd) is set equal to half of the actual congestion window and the TCP sender performs the congestion avoidance algorithm. Moreover, the TCP sender sets the congestion window reduced (CWR) bit in the header of the next outgoing TCP segment to tell the TCP receiver that it has reacted to the congestion notification. The TCP receiver reacts to the CWR bits by stopping to send the ECN echo if there is no new congestions in the network (figure 3-29).

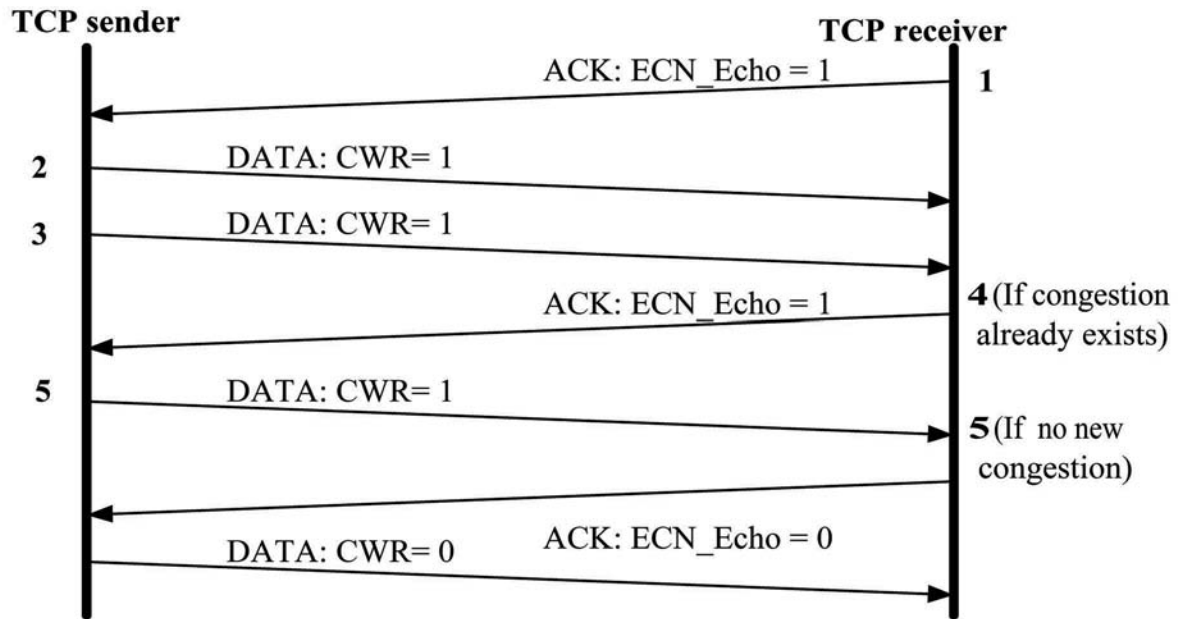


Figure 3-29: Packet marking at the end host and the TCP congestion window

3.5.3.3 TCP Initialization

In the TCP connection setup phase, the source and destination TCP exchange the information about their desire to use ECN.

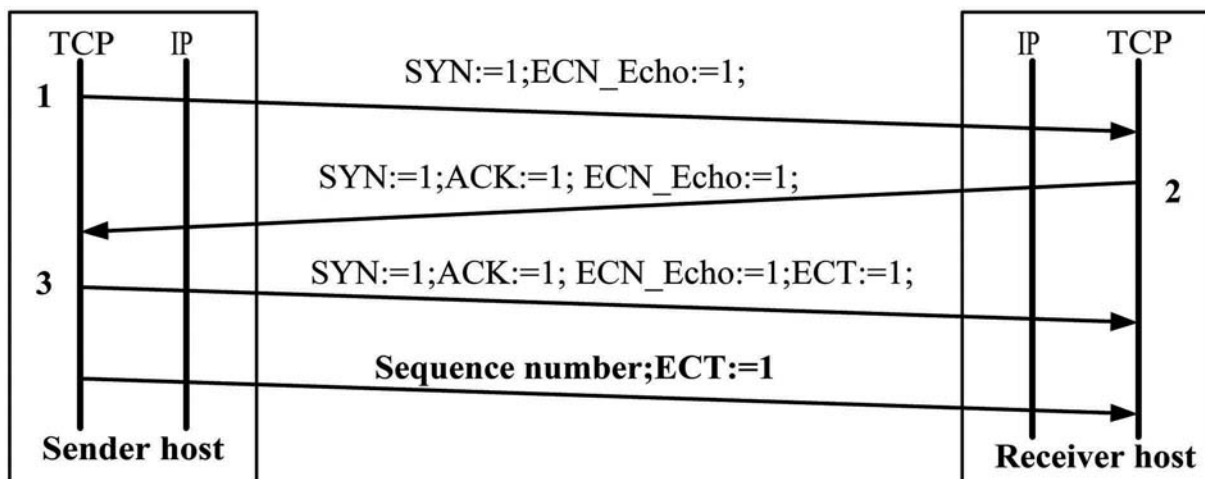


Figure 3-30: ECN negotiation within the TCP connection setup phase

In order to negotiate the information for using ECN, the TCP sender first sets the ECN echo flag in the first SYN segment. On receiving of this SYN segment, the TCP receiver sets the ECN echo bit in the SYN-ACK segment. Once this agreement has been reached at the TCP sender, the IP instance at the TCP sender host set the ECT bit in the IP header of all outgoing TCP segments

(figure 3-30). This ECT bit indicates that the packet is from an ECN capable host.

3.5.4 Non-TCP Unicast Congestion Control

In the past, TCP was used for all of the predominant elastic applications such as HTTP, SMTP, FTP. Therefore, the early congestion control implemented within TCP only focused on controlling and preventing the congestions for this elastic traffic. Unfortunately, the congestion control algorithms in TCP are not ideal for multimedia applications for two main reasons. First, TCP provides 100% reliability, i.e. every lost packet is retransmitted until it is correctly received at the receiver. This can be wasteful if the retransmission attempts delay the packet so much that it is out-of-date when it arrives safely. Second, the congestion control algorithms couple congestion control with loss recovery. This is a good feature for elastic applications, but becomes a problem as wireless components and multimedia applications increasingly become an integral part of the Internet.

Both of these problems become important within the proliferation of real-time multimedia applications and wireless applications on the Internet. This section offers a survey of congestion control mechanisms used for real-time and multimedia applications.

3.5.4.1 TCP Friendly Rate Control

TCP friendly rate control (TFRC) specified in RFC 3448 [HFP-2003] is a congestion control mechanism designed for unicast flows competing with TCP traffic. TFRC could be implemented in a transport protocol such as DCCP, or in an application that uses RTP and RTCP as transport protocol. TFRC is designed to be reasonably fair when competing for bandwidth with TCP flows, where a flow is reasonably fair if its sending rate is generally within a factor of two of sending rate of a TCP flows under the same conditions. However, TFRC has a much lower variation of throughput over time compared with TCP, which makes it more suitable for applications such as telephony or streaming media where a relatively smooth sending rate is of importance [HFP-2003].

The TFRC is rate-based congestion control in which the TFRC directly uses a throughput equation as a function of the loss event rate, round-trip time and packet size for determining its sending rate. TFRC generally works as follows [HFP-2003]:

- The TFRC receiver measures the loss event rate and sends this information back to the TFRC sender.

- The TFRC sender uses the information in these feed back messages to measure the round-trip time (RTT).
- The measured loss event rate and RTT are then fed into the throughput equation, determining the acceptable sending rate.
- The sender adjusts its transmission rate to match the calculated rate.

3.5.4.1.1 Throughput Equation for TFRC

The throughput equation recommended for TFRC [RFC3448] is a slightly simplified version of the throughput equation for Reno TCP from [PFT-1998]. This recommended throughput equation is described as follows:

$$X = \frac{s}{R \cdot \sqrt{\frac{2b \cdot p}{3}} + rto \cdot 3 \cdot \sqrt{\frac{3 \cdot b \cdot p}{8}} \cdot p \cdot (1 + 32p^2)} \quad (3.16)$$

where:

- X is the transmit rate in bytes per second,
- s is the packet size in bytes,
- R is the round trip time (RTT) in seconds,
- p is loss event rate, between 0 and 1.0,
- rto is the TCP retransmission timeout in seconds,
- b is the number of packets acknowledged by a single TCP acknowledgement.

3.5.4.1.2 TFRC Message Content

Since TFRC will be used along with a transport protocol or will be implemented within a transport protocol, it depends on the details of the transport protocol used. Therefore, no packet formats can be specified. But, to enable the TFRC functionality, data packets sent by senders and feedback packets sent by receivers should contain information that will be used for calculating the RTT and the sending rate. In particular, each data packet sent by the TFRC sender contains a sequence number, a time stamp indicating when the packet is sent and the RTT estimated by the sender. Each feedback packet sent by the receiver contains the timestamp of the last data packet received, the time between the receipt of the last data packet and the issue of the feedback message at the receiver, and the loss event rate estimated by the receiver.

3.5.4.1.3 TFRC Sender Functionality

The TFRC sender sends data packets to the TFRC receiver at a calculated rate. By receiving a feedback packet from the TFRC receiver, the TFRC sender changes its sending rate according to the information contained in the feedback

packets. If the sender does not receive a feedback within a given time interval (called nofeedback timer), the sender reduces its sending rate to a half. The TFRC sender protocol is specified in RFC 3448. It operates in the following steps:

- *Measuring the packet size.* The packet size s is normally known to an application. But this may not be so when the packet size varies depending on the data. In this case the mean packet size should be measured.
- *Sender initialisation.* This step deals with setting the initial values for X and for the nofeedback timer.
- *Sender behaviour when a feedback packet is received.* The sender knows its current allowing sending rate (X) and maintains a current RTT and timeout interval. When a feedback packet is arrived at the sender, the sender first calculates a new RTT sample. Based on this RTT sample, it estimates a new RTT and updates it. According to this new RTT, the sender updates the timeout interval and its sending rate. Finally, it resets the nofeedback timer expire after $\max(4 \cdot R, 2 \cdot s/X)$ seconds.
- *Sender behaviour if the nofeedback timer expires.* If the nofeedback timer expires, the sender cuts its sending rate in half. If the receive rate has been changed, the sender updates its sending rate based on the receive rate and the calculated sending rate. Finally the sender restarts the nofeedback timer to expire after $\max(4 \cdot R, 2 \cdot s/X)$ seconds.
- *Scheduling of packet transmission.* This steps deals with mechanisms for sending data packets so that the correct average rate is maintained despite the course-grain or irregular scheduling of operating system.

3.5.4.1.4 TFRC Receiver Functionality

Two main tasks at the TFRC receiver are measuring the loss event rate and periodically sending the feedback messages to the sender.

The receiver performs a loss rate measurement based on the detection of the lost or marked packets from the sequence numbers of arriving packets. TFRC assumes that each packet contains a sequence number, which is incremented by one for each packet sent. The receiver uses a data structure to keep track of which packets have arrived and which are missing. The loss of a packet is detected by the arrival of at least three packets with a higher sequence number than the lost packet

The second main task at the receiver is the transmission of the feedback message to the sender. This feedback transmission is specified in the following steps:

- *Receiver behaviour when a data packet is received.* When a data packet is received, the receiver performs following tasks. First, it adds the packet to

the packet history and sets the previous loss event rate equal to the loss event rate. Second, it then calculates the new loss event rate. If the new calculated loss event rate is less or equals the previous loss rate, no action needs to perform. Otherwise the receiver causes the feedback timer to expire.

- *Receiver behaviour when feedback timer expires.* If data packets have been received since the previous feedback was sent, the receiver performs the following steps. It first calculates the average loss event rate and measured receive rate based on the packets received within the previous time interval. The receiver constructs and sends a feedback packet containing the information described above. Finally it restarts the feedback timer to expire after the RTT value included in the received packet with the maximum sequence number.
- *Receiver initialisation.* This step deal with the initialisation of the receiver if the first packet arriving at the receiver. When the first packet arrives, the receiver sets the loss event rate and the receive rate equals to 0. The receiver then constructs and sends the feedback packet. Finally, the receiver sets the feedback timer to expire after the current estimated RTT value.

3.5.4.2 TCP Like congestion Control

The TCP like congestion control (TLCC) is specified in RFC 4341 [FK-2006] and used as a congestion control mechanism in DCCP, which is unreliable transport protocol for data messages but reliable for feedback messages. This TLCC mechanism is closely follows the mechanism used in SACK-based TCP. The differences between TCP like congestion control and the TCP congestion control are [FK-2006]:

- TLCC is applied to acknowledgements and TCP congestion control is applied to data packets. That means the congestion control by TLCC is performed by loss of acknowledgements and not by loss of data packets.
- Several parameters used for congestion control are specified by TLCC in packets and not in bytes like TCP.

Depending on the loss of acknowledgement, the slow start, congestion avoidance, fast retransmit and fast recovery described by TCP congestion control are performed.

3.5.5 Multicast Congestion Control

The increasing popularity of group communication applications such as teleconference and information dissemination services has led in development of

multicast transport protocols layered on top of IP multicast. These multicast transport protocols could cause congestion collapse if they are widely used but do not support adequate congestion control. In order to scope with this deployment in the global internet, it is necessary to implement congestion control mechanisms in each multicast transport protocol.

This section surveys and discusses fundamental congestion control mechanisms that could be implemented in any multicast transport protocol. The section starts with a discussion of requirements for the multicast congestion control. After that a classification of multicast congestion control schemes will be shown. Finally, the end-to-end and router-supported congestion control mechanisms will be described in detail.

3.5.5.1 Classification of Multicast Congestion Control

The multicast congestion control approaches can be categorized into four classes [ML-2003, PTK-1993]: sender-controlled, receiver-controlled, end-to-end schemes and router-supported schemes.

3.5.5.1.1 *Sender-controlled Congestion Control*

The basis principle of the sender-controlled approaches is that the sender actively adjusts its transmission rate based on feedback information generated from the multicast receivers to avoid overloading the link toward its receivers. Sender-controlled approaches can be categorized into two following classes:

- *Sender-controlled, one group.* Only a single multicast group is used for data delivery. The feedback information from multicast receivers is sent to the sender and the sender uses this information to regulate the sending rate for multicast receivers. The goal is to send data at a rate dictated by the slowest receiver.
- *Sender-controlled, multiple groups.* The initial multicast group is subdivided into subgroups with subdivisions centered on congestion points in the networks. The data is then sent to different groups with the adjusted rate.

A problem of the sender-controlled approaches is that having each receiver frequently reports feedback information would result in a feedback implosion problem at the sender. To reduce the flow of feedback information from receivers, following mechanisms have been proposed:

- *Suppression of feedback messages.* In this approach, a receiver suppresses the transmission of a feedback report if it has noticed that some other receivers have already sent a similar report.

- *Representatives.* In this approach, not all receivers will send their feedbacks to the sender. One solution is to select some receivers as representatives, and only the representatives send their feedbacks. For example, intermediate routers along the multicast tree collect feedback messages from the multicast leafs or nodes connecting to them and sum up the information into a single report which is handed to the router higher in the tree. The problem of this approach is how to choose a suitable set of representatives.
- *Polling.* The polling process is done by having sender and receivers generate a 16 bit random key. The sender sends a control message asking for a feedback with the generated key with all digits marked as significant. Only receivers with similar key are allowed to send feedback information.

In order to adapt the transmission behaviour, rate-based or window-based congestion control mechanisms discussed in section 3.5.2 can be used. Using rate-based congestion control, the sender can adjust the transmission rate directly based on the feedback information from the receivers. The transmission rate could be calculated based on one or several parameters that the sender receives in the feedback packets, such as RTT, packet loss rate or maximum packet size. By window-based congestion control, the sender uses a sliding window to control the amount of data the sender can transmit. This sliding window is updated based on the information from the receivers. The different to the sliding window by TCP is that the window is only increased if all receivers acknowledge the reception of the same packets.

The main disadvantage of sender-controlled approaches is that a single sender transmission rate cannot satisfy the conflicting bandwidth requirements at different sites, because end systems connect via internet through different link capacities and end systems have different processing capacities. Solution for this problem is the receiver-controlled congestion control that will be discussed in the next paragraph.

3.5.5.1.2 Receiver-controlled Congestion Control

The basis idea of the receiver-controlled schemes is that the receivers actively join and leave the multicast groups depending on their measurements of the transmission rate or of the congestion situation in the network. Receiver-controlled approaches are categorized into two following classes [HFW-2000]:

- *Receiver-controlled, one group.* A single multicast group is used for data delivery. The receivers determine if the sender is transmitting too rapidly for the current congestion state. If this is the case, the receivers leave this multicast group.

- *Receiver-controlled, layered organization.* The source data is generated in a layered format and striped across multiple layered multicast groups simultaneously. Receivers join and leave these layered groups depending on their measurements about the congestion state in the network. Receivers decide on how many layers they can join and leave. This approach functions as follows. Source sends multicast data in several layers (multicast groups). Each receiver joins the basis layer containing the minimal information necessary to achieve basis quality, and if no losses were observed, the receiver can join the next higher layer. When noticing some congestion, the receiver leaves its current layer and goes to the next lower layer. Each higher layer provides refinement information to the previous layer. Each receiver must listen to all lower layers up to and including the highest one.

3.5.5.1.3 End-to-End vs. Router-Supported Congestion Control

The end-to-end multicast congestion control schemes mainly require the collaboration of the sender and/or the receiver(s) and don't need any support from any immediate multicast routers. In the router-supported schemes, additional mechanisms are added to the multicast routers to assist in multicast congestion control. Examples of such mechanisms are [HWF-2000]:

- Conditional join specifies a loss rate, which is acceptable for the router to reject the join.
- Filtering traffic at different points in the network depending in the local congestion state.
- Combining the fair queuing scheduling with the end-to-end adaptation.

3.5.5.2 Requirements for Multicast Congestion Control

Each multicast congestion control should meet a set of requirements summarized in the following [RFC2357, BZ-1993, RFC2887]:

- *Receiver's heterogeneity.* Receivers can get different levels of quality of service, which translates into different rates delivered to different receivers. Thus, the first problem that needs to be considered for any multicast congestion control is a method to enable the sender to communicate with several receivers and satisfy receiver's requirements (such as QoS requirements) simultaneously.
- *Scalability.* A multicast congestion control should be able to deal with the heterogeneity of receivers in a scalable manner. In general, the number of receivers and the feedback implosion are the main sources of the scalability problem.

- *Fairness.* The third difficulty of multicast congestion control is the fairness problem. There are many possible ways to define fairness. One popular notion is the max-min fairness discussed in the scheduling section. Other type of fairness definition is global fairness that enables each entity to have an equal claim to the network's scarce resources, e.g. an entity traversing N congested links is using more scarce resources than an entity traversing one congested link. From formats of adjustment algorithms, [GS-1999] defined two other types of fairness: rate-oriented and window-oriented. A rate-oriented fairness tries to achieve equal throughput at the bottleneck resource. A window-oriented fairness achieves throughput proportional to the inverse of round trip time. Since most video applications are based on UDP, which is unfair to TCP, the multicast congestion control should provide fairness by a protocol at a level higher than UDP.

3.5.5.3 End-to-End Schemes

A survey of the end-to-end multicast congestion control schemes is presented in [ML-2003]. In this section some selected approaches will be described.

- *TCP-Friendly Multicast Congestion Control (TFMCC).* TFMCC is an IETF standard described in RFC 4654 [WH-2006]. Its basic principle is that each receiver measures the loss event rate and its RTT to the sender. Each receiver uses this measurement information, together with the TCP equation to derive a TCP-friendly sending rate. TFMCC implements a distributed feedback suppression mechanism, which only allows a subset of receivers to send feedbacks so that the feedback implosion at the sender can be prevented. Receivers, whose feedback is not suppressed, report the calculated transmission rate back to the sender in the receiver reports and measure their RTTs. The sender then selects the current limiting receiver CLR (the receiver that reports the lowest rate) and reduces the sending rate to match the CLR's calculated rate. The congestion control information is contained in the packets sent by sender and in the feedback packets from the receivers.
- *Receiver-driven Layered Multicast (RLM)* [MJV-1996]. RLM is a transport protocol that allows the receivers to adapt the quality of video they receive according to their available bandwidth. In RLM, the video signal is encoded into a number of layers. The lowest layer contains the basis information, and each subsequent layer provides progressive enhancement. The sender sends each video layer to a separate IP multicast group and takes no active role in rate adaptation. Each receiver joins to a corresponding IP multicast group for subscribing to a certain set of video

layers. When a receiver detects congestion, it leaves this layer and joins the lower layer. When there is spare bandwidth available, the receiver adds a layer.

- *Layered Video Multicast with Retransmissions (LVMR)* [LPA-1998]. LVMR is a protocol for distributing MPEG-encoded video over a best-effort network. It uses layered encoding and layered transmission in the same fashion as RLM. In comparison with RLM, LVMP offers two major contributions to the layered multicast. First, LVMR regulates the video reception rate at the receivers using a hierarchy of agents that help receivers decide to join and drop a layer. Second, LVMP introduces the concept of recovery using retransmission from designated local receivers to reduce recovery time.

3.5.5.4 Router-Supported Schemes

The router-supported schemes can be classified into two categories. The first one is the single-rate with packet filtering that is usually based on active queue management mechanisms at the routers. In these schemes, packets are dropped at the routers during congestion based on some criteria such as priorities marked for packets. The second category is multi-rate layered schemes that rely on sending the data in layers and letting routers manage the subscription to the layers and the flow control of each layer. A survey of the end-to-end multicast congestion control schemes is presented in [ML-2003]. In this section some selected approaches will be described.

- *Efficient Congestion Avoidance Mechanism (ECAM)*. This schema belongs to the single-rate with packet filtering category. ECMA is based on a combination of Explicit Congestion Notification (ECN) and Random Early Detection (RED). The basic principle of this schema is that the source sends information in one multicast flow using a single rate. Congestion is detected by routers via monitoring the average queue size and comparing with the RED's threshold. For the packets that are marked by the source as ECN-capable, the router may send an ICMP SQ message to the source to let it know about the incipient congestion, resulting in rate reducing at the source in response to congestion control.
- *Receiver-Selectable Loss Priorities (RSLP)*. This schema belongs to the multi-rate layered category. It works as follows. The source sends data in layers as different multicast groups. Receivers subscribe to as many layers of data as their bandwidth allows them. Also, the receiver has option to subscribe to a layer at higher priority or lower priority. During congestion, the router attached to the congested link drops packets associated with groups mapped as higher priority at this router.

- *Router-Assisted Layered Multicast (RALM)*. In this schema, the router monitors the queue status of each outgoing link. If congestion is detected on the links, the router immediately suspends some of the current transmitted groups on that link temporarily. Routers will try reactivating a suspended group on an outgoing link when congestion is relieved on this link.

3.6 Active Queue Management

The traditional queue management technique DropTail (Drop from Tail) manages the length of queues in routers by dropping the incoming packets from the tail of queue only when the queue overflows. For each arriving packet, the DropTail checks the recent queue length. If it is less than the maximal queue size, the packet is buffered into the queue based on FCFS (first come first serve), otherwise the subsequent packets are dropped (Figure 3-31).

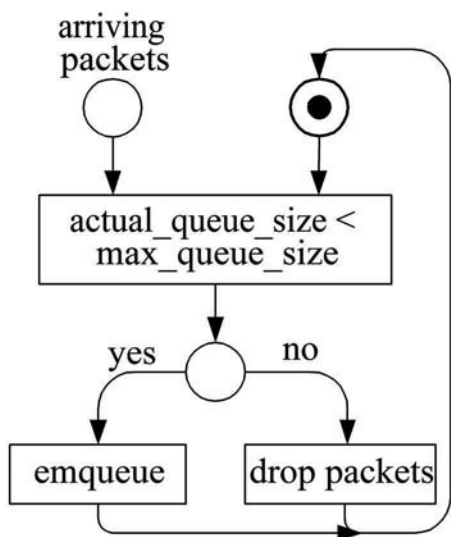


Fig. 3-1

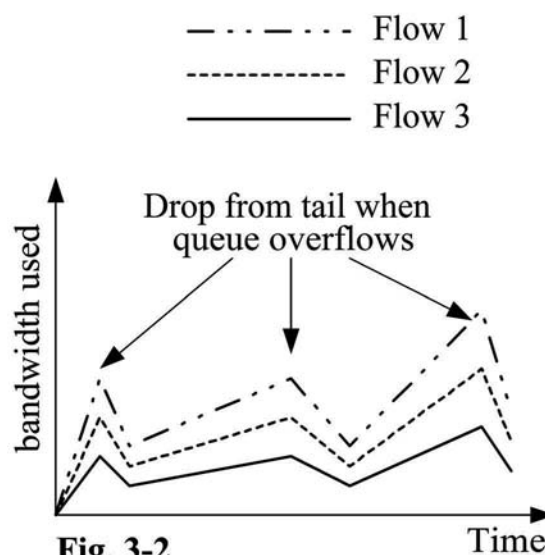


Fig. 3-2

Figure 3-31: DropTail principle and Figure 3-32: Synchronized TCP flows

DropTail was the standard queue management in Internet for years, but it has several fundamental drawbacks. Firstly, transport protocols, such as TCP, still suffer enough loss to shut down. When majority of traffic on a congested link consists of TCP traffic from various sources, DropTail drops packets from all connections when queue overflows, causing all the TCP sources to slow down their sending rates at the same time. This causes under utilization of the link until sources increase their transmission rates again. Over a period time t , TCP sources ramp up the sending rate, and when the link is congested again, all TCP senders back off at the same time. This problem is called global

synchronization (Figure 3-32). Furthermore, DropTail drops the subsequent packets with the same way without considering of packet types and of applications to which these packets belong to. This has a negative effect to the drop rate of multimedia applications that use the UDP as their transport protocol. Moreover, in some situation DropTail allows a single connection or a few connections to monopolize the queue space, preventing other connections from getting room in the queue. This effect is called lock-out phenomenon which is often the result of synchronization or timeout effects.

A solution for problems of the conventional queue management technique is the active queue management (AQM). AQM is a technique that explicitly signals the congestion to the senders and actively manages the queues at network elements. Its aim is to prevent congestion in packet switched networks. The Active Queue Management monitors the queue size and starts dropping and marking packets before congestion occurs. Thus, the problem to be solved by each AQM is the packet drop strategy that makes decision of

- When should the routers drop/re-mark the packets in order to signal the end systems above the congestion?
- Which packets should be dropped (or remarked) when queue size exceeds a given threshold?

3.6.1 Packet Drop Policies

The packet drop strategy includes four fundamental policies [SH-2002, Kes-1997]: (1) degree of aggregation, (2) choice of drop priorities, (3) early or overloaded drop, and (4) drop position. These policies will be discussed in the followings.

3.6.1.1 Degree of aggregation

A drop algorithm may be applied for a per-connection state, or for aggregate connections. With a per-connection state, the drop policy provides more protection among several connections, at expense of management of more connection states. Aggregating several connections into one flow class will reduce connection states, but during queue overload, packets from all connections within this class will be dropped by the same way, thus the connections are not protected from each other.

If packets are buffered per-connection queue and share buffer from a common pool, a drop algorithm, which always drops packets from longest queue, can provide good condition to a scheduler to achieve max-min fair share. To see it, note that when the per-connection queues are not full, connections get the buffer whatever they need, and thus they get the service from the scheduler.

However, when packets arrive to a full buffer, the AQM drops one or more packets from the longest queue, creating the space for incoming packets. This drop algorithm together with a scheduling discipline (e.g. WRR or WFQ) can ensure that backlogged connections get equal shares, while non-backlogged connections are fully satisfied, which is a criterion for max-min fair share.

3.6.1.2 Drop position

An AQM can drop packets from the head or tail of a per-connection (or aggregated connections) queue, from a random position or from an entire longest per-connection queue.

- *Dropping packets from the tail of the queue:* Arriving packets are dropped when queue is full. The approach is easy to implement, because the scheduler simply drops the incoming packets and does not need to modify the queue points. However this approach has a negative consequence. Consider a source (such as TCP) that detects lost packets using duplicate acknowledgements via the fast-retransmit algorithm. If the packets are dropped from tail, the sender recognizes that the packets have been lost via receiving the duplicate acknowledgements from receiver only after the entire queue has been served, because the receiver discovers the packet loss via the missing sequence number when it receives the packets in the tail of the queue, and after that the receiver then sends the duplicate acknowledgements.
- *Dropping packets from head of the queue:* The packets, which are at the head of the queue, are dropped, when queue is full. In this approach, the queue point must be modified after dropping the packets and thus it is expensive to implement. But in comparison with drop from tail, this approach improves the performance of the sources that implement the fast-retransmit algorithm. The reason is that if packets are dropped from the head of the queue, the receiver recognizes the packet loss via the missing sequence number when it receives the packets in the head of queue and not in the tail. Thus, the receiver can send duplicate acknowledgements sooner and therefore the source can take the retransmission action earlier.
- *Dropping packets from random positions.* If the queue is full, packets with a random position in the queue are dropped. Therefore, this approach distributes packet losses fairly among connections. Connections that send data at a faster rate may have more packets in the buffer than connections that send data at a slower rate. Thus, packets from faster connections will be dropped more than slower connections. This approach is very complex,

since it not only needs to compute a random number, but also to remove a packet from a position in the queue. Thus, it is not implement-able in real systems.

3.6.1.3 Drop priorities

The first form of drop priorities is that ingress routers or end hosts can mark packets entering the network as lower-priority packets. This packet marking is based on the traffic description described in section 3.3. When the network is under loaded, these packets soak up available bandwidth, but when the network is congested, the AQM first drops the lower-priority packets. The problem of this approach is how to choose the traffic descriptor for each traffic classes.

The second form of drop priorities is called packet discard that concerns packet losses versus cell losses. The loss of a single ATM cell causes the entire IP packet to be lost. Thus the loss of a burst of cells could cause multiple IP packets to be lost, resulting in a error multiplication. To avoid this problem, when an ATM switch drops a cell, it informs the corresponding connection to be in drop state and drop all subsequence cells belonging to a packet, until it receives the last frame.

The third form of drop priorities refers to drop packets from connections that nearly originate, rather than connections that enter the network farther away. The reason is that dropping packets that have used a lot of network resources is a waste, so dropping packets that have recently entered the network is better. In practice, each IPv4 packet carries an hop count incremented at each hop toward destination. Thus, the AQM can drop packets with smaller hop count. However, this approach requires storing the packets in order of increasing hop count, which leads to increase the complexity of the drop algorithm. Moreover, the IPv4's time-to-life field decreases at every hop, instead of increasing. Thus, this form cannot be implemented in Internet.

3.6.1.4 Early or overloaded drop

The overloaded drop queue management drops incoming packets when the buffer is full. In contrast, early-drop AQMs deal with strategies to drop packets even if the buffer is not full. This approach is suitable for endpoints that recognize the lost packets as an implicit congestion signal from the network, and thus these endpoints reduce their sending rate in response to packet loss. If an endpoint does not reduce its sending rate, the router will become overload and drops packets anyway. Therefore, an early drop AQM needs cooperative sources.

There are two forms of early drop AQM – the early random drop [Has-1989] and random early drop [FJ-1993]. The early random drop AQM drops each arriving packets with a fixed drop probability, whenever the instantaneous queue length exceeds a certain threshold. Since the misbehaving sources intuitively send more packets than well-behave sources, so dropping a arriving packet as random is more like to drop a packet from a misbehaving source. Therefore, the schema can target misbehaving sources without affecting the bandwidth received by well-behave resources. However, a disadvantage of the early random drop is that this drop policy is not successfully in controlling misbehaving sources. Random early detection (RED) improves early the random drop in two ways. First, packets are dropped based on an average queue length instead of instantaneous queue length. This allows AQM to drop packets only during sustained overloads, rather than current overload. Second, the packet drop probability is a linear function of the average queue length. And, an increasing of the average queue length causes the increasing of packet losses.

3.6.2 DEC-Bit

The TCP congestion control uses the so-called implicit feedback to recognize the congestion. So, the traditional queue management drops in the wake of congestions, which results in a global synchronization, timeout and unnecessary retransmissions. DECbit is an explicit method for signalling the congestion. It was proposed by Ramakrishnan and Jain [RJ-1988, RJ-1990] and was developed for the digital network architecture at DEC. It has since been specified at the active queue management (and congestion control) mechanism for ISO transport protocol class 4 and for connection less network protocols.

The key idea of DEC bit is that a router, which is experiencing the congestion, sets a bit (called congestion indication bit, CI bit) in the header of all incoming data packets on the data path toward their destinations. When such data packets arrive at the receiver, the receiver copies the CI bit into its acknowledgements and send them back to the source (Figure 3-33). Based on the CI bits in the acknowledgements, the source adjusts its transmission rate. The important elements in the DECbit schema are how a router decides when to set the CI bit and for which connections, and how these bits are interpreted by the sources. To do it, following actions are performed at the DECbit capable routers and at the sources.

- *DECbit capable Router.* Each DECbit router monitors the arriving packets from each source and compares it with two thresholds. The first threshold is defined as one (for one packet) and the second one is set to two (for two packets). Based on the amount of the incoming packets, the router

computes the bandwidth used by each source and the mean length of the queue shared by all sources. If the measured mean queue length exceeds the first threshold, this means that the server has at least one packet waiting in the queue, so it is 100% utilized, the router sets CI bit on packets from sources whose demand is larger than max-min fair share. This causes these sources to reduce their window size, and thus their sending rate, relieving the load on the server. If the measured mean queue length exceeds the second threshold, this means that the server is not only 100% utilized, but its effort of setting bits has not decreased the queue size. The router therefore goes into the panic mode and sets the CI bit on all incoming packets

- *DECbit capable source.* A DECbit source keeps track of the CI bits it receives in the header of acknowledgements and uses them to adapt its sending rate.

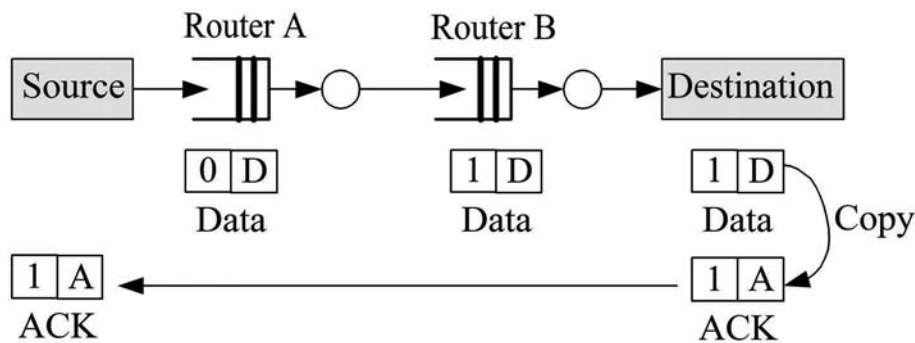


Figure 3-33: DECbit

3.6.3 Random Early Detection

The random early detection (RED) is a congestion avoidance mechanism that works on the basis of the active queue management. The RED mechanism was proposed by Sally Floyd and Van Jacobson [FJ-1993] in the early 1990s to address the network congestion in a responsive rather than relative manner.

An assumption of the RED is that the transport protocols at the end system are sensitive to the packet loss and will temporarily slow down their sending rates by packet losses. In contrast to the traditional queue management DropTail, which drops packets only when queue is full, the RED tries to anticipate the congestion, and as the queue grows, RED begins signalling the congestion by discarding subsequent packets probabilistically before the queue runs out of the buffer space. In response to each packet discarded, one source at most will react and slow down its sending rate, resulting in a reduction of traffic transmitted to routers. If the reduction is insufficient, RED will react by

increasing the frequency of packet dropped. If the reduction was sufficient, to ease the congestion, RED will reduce the frequency of drop. The drop probability is dependent on a running average queue length to avoid any bias against bursty traffic.

The RED maintains three variables used for calculating the average queue length and the packet drop priority: maximum threshold (`max_threshold`), minimum threshold (`min_threshold`), and average queue length at time t (`queue_length(t)`) (figure 3-34).

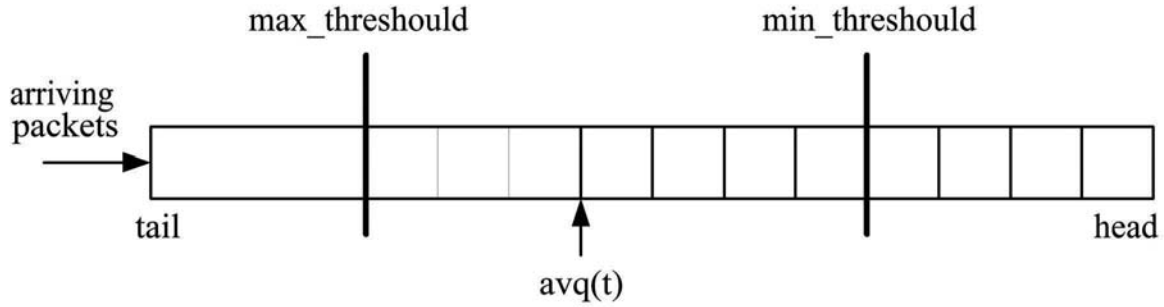


Figure 3-34: RED's variables within a RED queue

The RED mechanism itself consists of two main parts: (1) estimation of the average queue length and calculation of the packet drop probability, (2) Packet drop decision. These parts are described in the following paragraphs.

3.6.3.1 Estimating average queue length and packet drop probability

The RED controls the average queue size by using the equation (3.17) to compute the average queue occupancy based on the instantaneous queue occupancy. When a packet arrives, the RED updates the average occupancy $avq(t)$ via the following equation:

$$avq(t) = (1-w)*avq(t-1) + w*q(t) \quad (3.17)$$

Where w is the queue weight with $0 \leq w \leq 1$, $q(t)$ is the instantaneous queue occupancy, $avq(t-1)$ is the average queue length at time $(t-1)$ which is the time last packet arrived. Based on the average queue occupancy $avq(t)$, the per-packet drop probability p for this arriving packet is calculated via the following equations:

$$pb = \max_p \frac{avq(t) - \min\_threshold}{\max\_threshold - \min\_threshold} \quad (3.18)$$

$$p = \frac{pb}{1 - \text{count} * pb} \quad (3.19)$$

Where, *count* indicates the number of packets entering buffer since last dropped packets, *max_p* is the maximum drop probability if the average queue length is between *min_threshold* and *max_threshold* (Figure 3-36). The drop probability is used to determine whether to discard an incoming packet.

3.6.3.2 Packet Drop Decision

The RED algorithm compares the average queue length with a minimum threshold, *min_threshold*, and with a maximum threshold, *max_threshold*. If the average queue length exceeds the maximum threshold, RED drops all subsequent incoming packets – the packet drop probability is equal 1. Packets are not dropped so long as the average queue length remains below the minimum threshold. When the average queue length is between the thresholds, each arriving packet is dropped with a per-packet dropping probability *p* described in (3.18, 3.19).

```

For each arriving packet at time t:
IF  $avg(t) < min\_threshold$  THEN packet is accepted
    (No congestion, dropping probability is equal zero)
IF  $avg(t) > max\_threshold$  THEN packet is discarded
    (high congestion, dropping probability is equal 1)
IF  $max\_threshold \geq avg(t) \geq min\_threshold$  THEN packet is
discarded with a probability p

```

Figure 3-35: RED's drop decision

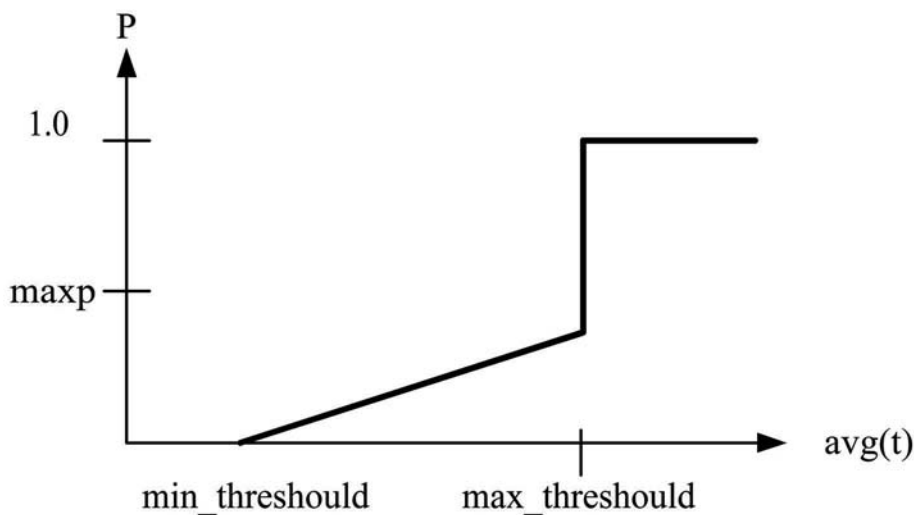


Figure 3-36: Packet drop probabilities in RED

The algorithm for packet drop decisions is described in figure 3-35. The packet drop probability depends on the average queue length and on the minimum and maximum threshold. The dropping probability is shown in figure

3-36. The packet drop rate increases linearly as the average queue length increases until it reaches the maximum threshold.

Figure 3-37 describes the RED algorithm in a simple pseudo code.

```

1 /*initialization */
2 avq := 0; /* actual average queue length */
3 count:=0; /*packets entering the buffer since last
   dropped packet*/
4 for each arriving packet i
5 begin
6   avq:= calculating the actual average queue length;
7   if (min_threshold  $\leq$  avq < max_threshold) then
8     begin
9       count:= count +1;
10      p:=calculating the dropping probability for packet I;
11      double u:=random::uniform();/*random number
   generation*/
12      if (u $\leq$ p) {dropping the arriving packet; count:=0;}
13      break;
14    end
15    else if (max_threshold $\leq$ avq) then
16      begin
17        dropping the arriving packet;
18        count:=0;
19      end
20    else count:=-1;
21  end

```

Figure 3-37: The RED algorithm

In comparison with the drop from tail, RED's intermittent discards can reduce the packet losses of each individual connection and thus it prevents the global synchronisation of the sources discussed in 3.5. While RED has certain advantages over the DropTail, it nevertheless has disadvantages. First, RED fails to employ per-connection (or per aggregated connections) information, and thus, discards may be inconsistent and lack uniformity. Second, RED relies on a discard probability that entails a random decision to discard packets from all connections in the same way.

3.6.4 Weighted Random Early Detection

Weighted random early detection (WRED) combines the capabilities of the RED mechanisms with the IP precedence to provide the preferential traffic handling of higher priority packets. WRED weights the drop probability for each packet

based on the precedence bit in the IPv4 header or on traffic class field in the Ipv6 header, which allows for service differentiation of different traffic classes. Packets with a higher priority are less to be dropped than packets with lower priority.

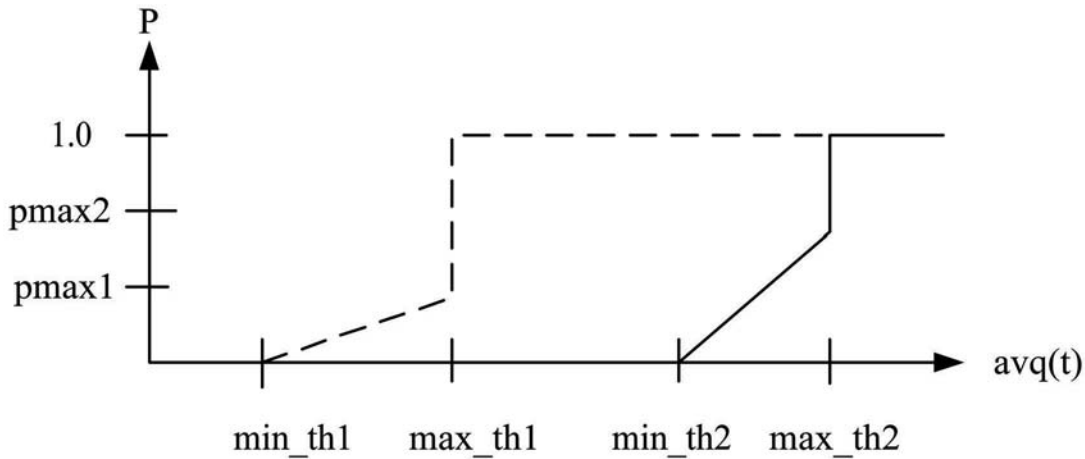


Figure 3-38: WRED with two dropping precedence

WRED is useful on any output interface where congestion is expected to occur. However WRED is usually used in the core routers, rather than edge routers. Edge routers assign drop precedence to packets as they enter the network. In core routers, WRED uses this precedence to determine how to treat different types of traffic. In order to drop packets differently, WRED allows assigning different RED dropping profiles to different classes of traffic. For each of these traffic classes, the dropping profile is a tuple of {minimum threshold, maximum threshold, maximum dropping probability}. Figure 3-38 shows an example of WRED with two RED dropping profiles: {min_th1, max_th1, pmax1} and {min_th2, max_th2, pmax2}. The profile {min_th2, max_th2, pmax2} is less aggressive than {min_th1, max_th1, pmax1}. WRED can assign a less aggressive RED profile to a certain types of packets and a more aggressive dropping profile to other types of packet given the same level of congestion.

The basic principle of WRED mechanisms is described in the following. For each incoming packet at time i , WRED first estimates its average queue length $avq(t)$ by using the equation (3.17). After that, WRED checks the IP precedence field in the packet header and assign this packet to a drop profile. The WRED then compares the average queue length with the minimum threshold and maximum threshold of this traffic class to decide whether to drop packets or not. The packet drop decision is performed with the same way as by RED. The basis principle of WRED is illustrated in figure 3-39.

1 For each arriving packet at time t :

- 2 Calculating the average queue length $avq(t)$ based on (3.17)
- 3 Checking the IP precedence field to find out, to which dropping profile the packet belongs to
- 4 Packet belongs to dropping profile k with $\{max\_threshold\_k, min\_threshold\_k, mayp\_k\}$
- 5 Calculating the dropping probability p_k for this packet based on (3.18, 3.19) and dropping profile k
- 6 IF $avq(t) \leq min\_threshold\_k$ THEN packet is accepted (No congestion, drop probability is equal zero)
- 7 IF $avq(t) > max\_threshold\_k$ THEN packet is discarded (high congestion, drop probability is equal 1)
- 8 IF $max\_threshold\_k \geq avq(t) \geq min\_threshold\_k$ THEN packet is discarded with a probability p_k

Figure 3-39: The basic principle of the WRED mechanism

In comparison with RED, the advantage of WRED is that it provides different dropping probabilities to different traffic classes and therefore it provides a mechanism for service differentiations in IP routers.

3.7 Routing

Routing is the process for determining a path used for delivering traffic from a source to each destination in a communication network. Routing is accomplished by means of routing protocols that create and update mutually consistent routing tables in every router in the network. In packet-switched networks, including IP networks, a router needs to be able to look at the destination address in the packet header and then determine an output port to which the packet should be forwarded. The router makes this decision by consulting a forwarding table. These logical routing components are shown in figure 3-40. The fundamental problem of routing is that how do routers acquire the information in their forwarding tables.

The terms forwarding table and routing table are sometimes used interchangeably, but there is a difference between them. When a packet arrives at a router, the router consults the forwarding table to decide to which output interface this packet should be forwarded. So the forwarding table must contain enough information to accomplish the forwarding function. This means that a row in the forwarding table contains e.g. the mapping from a subnet address to an outgoing interface and the MAC address of the next hop. The routing table, on the other hand, is created and updated by a routing protocol, and is as a precursor to building the forwarding table. A routing table contains at least three columns: the first is the IP address of destination endpoint or destination

network, the second is the address of the router that is the next hop in the path to this destination, and the third is the cost to reach this destination from this router. The cost may be for example the hop count.

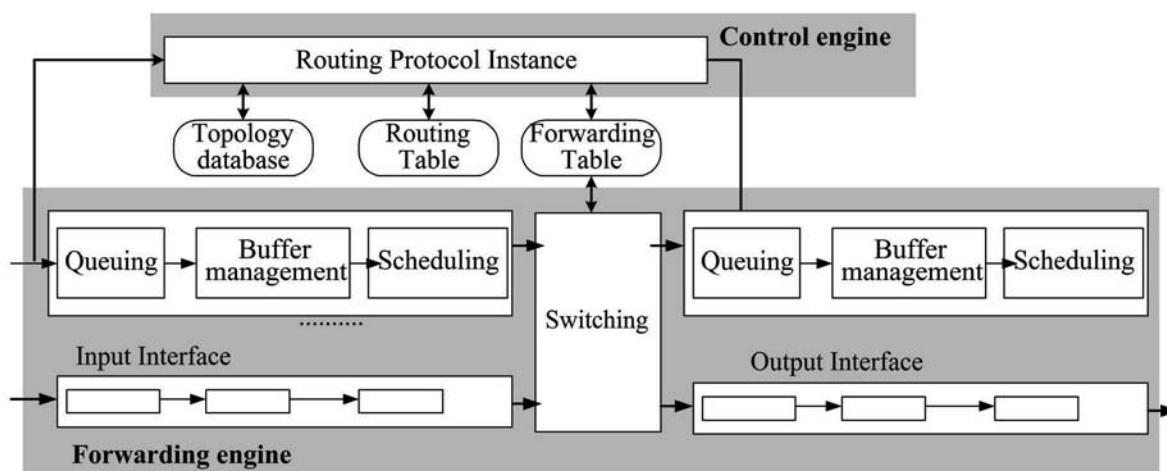


Figure 3-40: Routing protocol components within an Internet router

Since the main task of each routing protocol is to establish and update the routing tables, a routing protocol must be able to support following functions:

- *Topology discovery.* A routing protocol must be able to dynamically discover the network topology and to update the topology change. This is done via exchanging the routing protocol packets with other routers in the network.
- *Topology data summarization.* A routing protocol must be able to summarize the collected global topology information to exact only relevant portions to this router.
- *Path computation.* A routing protocol must be able to compute the paths from a router to every routers in the network
- *Routing table update.* A routing protocol must be able to asynchronously update the routing table based on the computed paths.

Depending on the communication forms and on the QoS aspects, routing can be classified into three categories (unicast routing, multicast routing and QoS routing) that will be discussed in this section.

3.7.1 Unicast Routing

Unicast routing is the process for determining the path used for delivering traffic from a source to a destination. This section starts with a classification of unicast routing protocols. Following it, the distance vector routing and link state routing algorithm will be illustrated. Finally, selected unicast routing protocols are described in detail.

3.7.1.1 Classification of Routing Protocols

Routing protocols can be categorized via different categories [Kes-2001]:

- *Operation areas.* Based on the operation areas, routing protocols can be classified as interior gateway routing protocols (IGP) and exterior gateway routing protocol (EGP). While routing protocols that operate within an autonomous system (AS) are called IGP, the routing protocols, which work between ASs, are the EGP. Examples for IGP in Internet are RIP, OSPF, EIGRP. The BGP (Border Gateway Protocol) is a EGP in Internet.
- *Centralized vs. distributed routing.* With a centralized routing, a central router collects and processes the global topology information. This central router then computes and updates the routing table for every router in the network. Distributed routing protocols are protocols that enable each router to collect the topology information and to create and to update its mutually consistent routing tables.
- *Source-based vs. hop-by-hop routing.* Source-based routing protocols are routing protocols in which a packet can carry the entire path, which defines the addresses of every router on the path from the source to destination. With a hop-by-hop routing protocol, the packet can hold just the destination address, and each router along the path can choose the next hop based on the forwarding table.
- *Stochastic vs. deterministic.* By using of a deterministic routing protocol, each router has for each incoming packet exactly one path toward a destination. In a stochastic routing, each router maintains more than one path toward destination for each packet. Thus, each router has more than one next hop for each packet. The router then randomly picks one of these hops when forwarding a packet.
- *Single vs. multiple path.* While in a single path routing a router maintains only one path to each destination, in a multiple path routing a router maintains a primary path along with alternative paths to each destination. An alternative path is used when the primary path is unavailable.
- *Static routing vs. dynamic routing.* Another category to classify the routing protocols is the way how the routing tables are built. With a static routing, routing tables are manually configured and updated from an administrator. Thus, packets are forwarded out of predetermined output interface. This routing is useable if the network is really small and the routes change infrequently. In contrast to static routing, a dynamic routing enables a router to automatically create and update its routing tables by using of routing protocols.

3.7.1.2 Distance vector routing

There are several dynamic routing protocols in the Internet. The primary difference between them lies in the way they discover the network topology information, which is done with two fundamental routing algorithms – the distance vector routing and the link state routing. Both routing algorithms allow a router to find the global routing information by exchanging topology information it knows with other routers in the network, but they differ in the manner, which defines where a router sends this topology information. The distance vector routing algorithm will be discussed in this section and the link state routing in the next section.

The main idea of a distance vector (DV) routing algorithm is that a router tells its neighbours its distance to every routers in the network. By this way, each router can discover the global network topology information.

Assumption by a DV routing algorithm is that each router knows its own address and the addresses of its neighbours. Each router maintains a distance vector consisting of a list of tuples <destination address; cost >, one tuple per destination. Whereby, the cost is the current estimation of the sum of link cost on the shortest path from this router to the destination. Based on this assumption, each DV router starts out with its initialised distance vector, consisting of zero for itself and infinite for every one else. Each router constructs its routing protocol packets containing its distance vector and periodically sends these packets to its neighbours. Upon receiving the routing protocol packets from neighbours, a router compares its current cost to reach a destination with the sum of the cost to reach its neighbours and the cost to reach the destination from its neighbours. It takes the path with smaller cost and then updates its routing table and its distance vector.

The distance vector algorithm works well if the routers and the links between them are always up, but it has the so called count-to-infinity problem when links go down or come up. This problem is illustrated with the simple network shown in figure 3-41. Initially, Router A forwards packets to the router C via the router B. Thus, A has a two-hop path to C. Now suppose the link BC goes down. B updates its cost to C to infinity and sends it within distance vector to A. B knows that A has two-hop path to C. Because B is neighbour of A, B has one-hop path to A. Therefore B updates its routing table that it has three-hop path to C and tells A about it within a distance vector. When A receives the distance vector from B, A updates its routing table to show a four-hop path to C. This process of increasing the hop count to C continues until the hop count reach infinity.

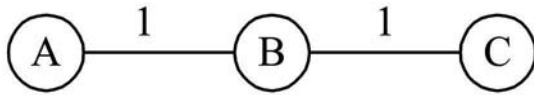


Figure 3-41: Count-to-infinity

Possible solutions for the count-to-infinity problem are described in [Spo-2002]:

- *Path vector.* The reason for count-to-infinity is that the distance vector sent from A to B did not describe that B was on the path from A to C. One possible solution is to assign in each entry of the distance vector the path used to obtain this cost. For example, A can tell B that its cost to C is 2, and the path to C is A-B-C. When B observes this, it recognizes that no path to C exists, and the count-to-infinity problem can thus not be arisen.
- *Split horizon.* The problem of the path vector discussed above is that the distance vectors require a large table size because of the path information. This can lead to overhead. Split horizon avoids this problem. The idea of the split horizon is that a router never advertises the cost of a destination to a neighbour, if this neighbour is the next hop to this destination. This means in figure 3-41 that A does not advertise a cost for C to B because it uses B as its next hop to C.
- *Triggered updates.* Most distance vector routing algorithms advertise the distance vectors in a time interval about 30 seconds. This adversely affects the time taken to recover the count-to-infinity situation. To avoid this, the solution “triggered updates” can trigger distance vector changes immediately after a link is marked down.
- *Source tracing.* The idea of source tracing is that a distance vector carries not only the cost to a destination, but also penultimate router of the destination. This method provides a source to have sufficient information to detect loop and to construct entire path to destination.

3.7.1.3 Link state routing

The second fundamental routing algorithm is the link state routing. In contrast to the DV routing algorithm where a router tells its neighbours its distance to every routers in the network, the main idea of a link state (LS) routing algorithm is that a router sends the network topology information with the cost of each link to all routers in the network. Each router processes the network topology information it received und update this topology information in its local link state advertisement database (LSA database). The router then uses the topology information in its LSA database to independently compute the cost-effective paths to every destination. Finally each router updates its routing table (figure 3-42). Thus, the key issues in a link state routing algorithm are how a router

distributes the network topology knowledge to every router in the network, and how a router computes shortest path from itself to every router. These issues will be discussed in this section.

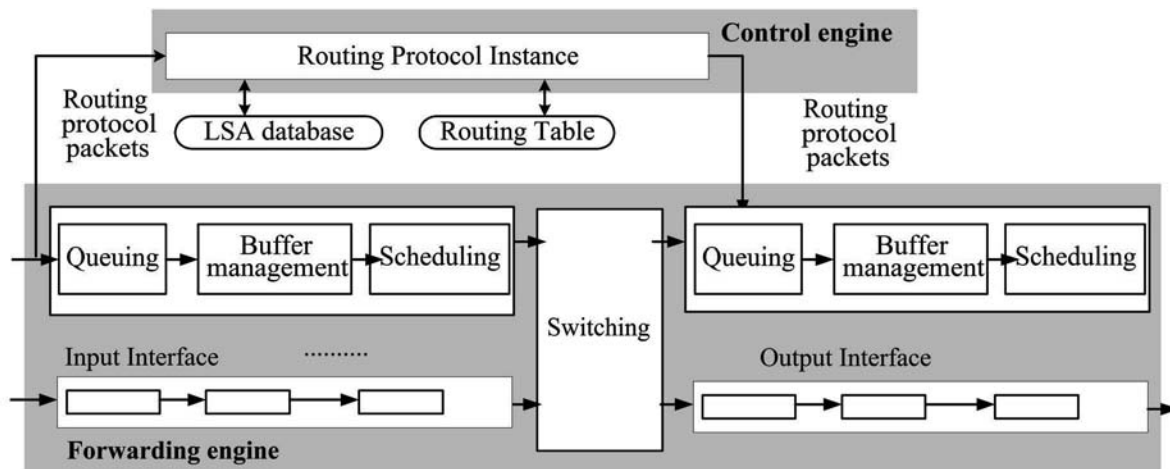


Figure 3-42: Basic components of a link state routing protocol

3.7.1.3.1 Network topology distribution and recovery

In order to distribute and to recovery the network topology, five fundamental routing protocol packets are generally used in each link state routing protocol.

- **Hello.** Hello packets are used to dynamically establish and maintain the neighbourhood relation ships.
- **Link State Advertisement (LSA).** Each piece of the link state database corresponds to a particular router's local state called link state advertisement. Each LSA packet describes this local state of a router. This includes the state of a router's interface and adjacencies. Each router in an autonomous system originates one or more LSA packets. These packets are sent via flooding to every router in the network.
- **Link State Update (LSU).** For reducing the overhead and improving the performance, each router can construct several incoming LSA packets into one LSU packet and sends this LSU packet to every router via flooding. Thus, a LSU packet can carries a set of LSA packets.
- **Data Base Description (DD).** For each router, the collection of all link state packets forms the link state database. A router can send it link state database to a router via several DD packets
- **Link state acknowledgement (LSACK).** Each LSU packet is acknowledged via a LSACK.

In order to discover the network topology, the link state routing algorithm performs following tasks: (a) sending hello packets and processing the incoming

hello packets; (b) sending LSA packets and processing the incoming LSA packets. These tasks are described as follows.

(a) Sending and processing the hello packets

In order to establish and to maintain the neighbourhood, each router supporting LS routing dynamically discovers its neighbour via periodically broadcast the so-called hello packets to all of its interfaces. Each hello packet generally contains following fields

- *Network mask*: used for identifying the network associated with this router,
- *Hello interval*: specifies the maximum time interval between the transmission of a hello packets,
- *Router dead interval*: defines the time a router declares a neighbour router is down if the router does not received the hello packet from this neighbour as a respond to a hello packet it had sent.
- *Neighbour list*: a list of the neighbours in the last router dead interval.

Acknowledgements to a hello packet are hello packets from neighbour routers. Based on the information in these hello packets, each router updates its LSA database. If a router does not receive hello packets from a neighbour within a router dead interval, the router removes this neighbour from its neighbour list and broadcast its new neighbour list (via hello packets) to all of its interfaces.

(b) Sending and processing the LSA packets

Each router participating in a link state routing creates a set of LSA packets describing the local state of routers it just learn. An LSA packet contains the router's ID, the neighbour's ID, the cost to the neighbour and a sequence number for this LSA packet. The router then constructs these LSA packets into one or more LSU packets, which are flooded to every router in the network. The basic principle of the flooding is that a router copies the incoming packet and transmits it on all outgoing interfaces except the one the packet came into. Moreover, the routers keep a list of packet sequence numbers, and if a packet with the same sequence number has already been seen, the routers drop this packet without sending it to the other links.

When a router receives a LSU packet from other routers, the router processes LSA packets in this LSU packet, in order to decide whether the router updates these packets into its local link state database or ignores them. The router then constructs new LSU packets from the received LSU packets and floods these new LSU packets to its outgoing links.

The processing of a LSU packet at a router is performed as follows. Each LSU packet contains a set of LSA packets. Each LSA packet contains a

sequence number that is incremented for each new packet created by a source router. Each router keeps track of all pair (source router, sequence number) it sees. When a new LSU packet comes, the router checks against the list of LSA packets it has already seen. If this LSA packet is new, the router updates this packet into its LSA database, constructs this LSA packet with other new LSA packets into a LSU packet, and floods it on all lines except the one it arrived on. If this LSA packet is a duplicate one, it is discarded.

3.7.1.3.2 Shortest path computation

We have seen how every router in the network obtains a consistent copy of the LSA database. Each router uses this database to compute optimal paths in the network. The shortest path computation is performed typically by using Dijkstra's shortest path algorithm. This algorithm computes the shortest path from a root node, which corresponds to the router where the algorithm is being run, to every router in the network. The main idea of this algorithm is to maintain a set of routers, *R*, for which shortest path has already been found. Every router not belongs to *R* must be reached by a path from a router that is already in *R*. The path to an outside router *R1* is the shortest path to *R1* if *R1* can be reached by a one-hop path from a router already in *R*. Detail about Dijkstra's shortest path algorithm is described in [Tan-2004].

3.7.1.3.3 Routing table update

The result of the Dijkstra's algorithm at a router is a shortest path tree describing shortest paths from this router to all routers in the network. Using the shortest path tree, each router updates its own routing table. For each shortest path to a destination, the router only takes the hop, which is next to it and writes it as the next hop to this destination.

Figure 3-43 shows the network topology of a simple autonomous system described in RFC 2328. The number on a link defines the cost of this link, and, a node defines a network or a router. The shortest path tree for the router RT6 and the RT6's routing table are shown in figures 3-44 and 3-45. Here we see that the router RT6 only takes three routers (RT5, RT6, RT10) in this shortest path tree as its next hop in its routing table.

Destination	Next Hop	Distance
N1	RT3	10
N2	RT4	10
N3	RT3	7
N4	RT3	8
N6	RT10	8
N7	RT10	12
N8	RT10	10
N9	RT10	11
N10	RT10	13
N11	RT10	14
H1	RT10	21
RT5	RT5	6
RT7	RT10	8

Figure 3-45: The RT6's routing table

3.7.2 IP Multicast Routing

There are three ways to develop multicast network applications – using the unicast, using the broadcast and using the multicast.

- *Using the unicast.* With a unicast design, applications must send one copy of each packet to each receiver of the multicast group. This technique is simple to implement, and, the intermediate systems (e.g. routers and switches) do not need special multicast functions and do not require copying or replicating the data. In addition, it requires extra bandwidth, because the same data has to be carried multiple times, even on a shared link.
- *Using the broadcast.* With a broadcast design, applications can send one copy of each packet and address it to a broadcast address. This technique is also simpler than using the unicast. However, if this technique is used, the network must either stop broadcasts at the LAN boundary or send broadcast every where, which is a significant usage of network resources even if only a small group actually need to receive the packets.
- *Using Multicast.* By using of the multicast paradigm, applications can send one copy of each packet and address it to the group of recipients that want to receive the packet. The multicast technique addresses packets to a group of receivers rather than to a single receiver and forwards the packets only to the networks that have receivers for this group.

Multicast can be implemented in four layers of the TCP/IP protocol stack – data link layer, network layer, transport layer and application layer. This section focuses only on multicast at the network layer – the IP multicast. The IP multicast provides explicit multicast support at the network layer. It enables the transmitting of a single packet from the sending host and replicating this packet at the router whenever it must be forwarded on multiple outgoing links in order to reach their receivers.

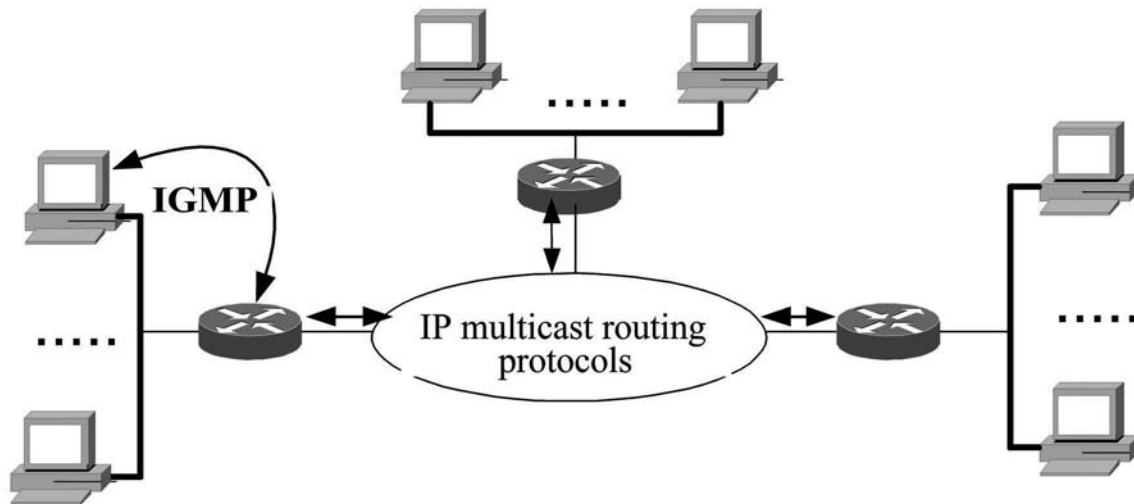


Figure 3-46: Two components of the network layer multicast

In order to support the IP multicast communications, three fundamental aspects must be addressed at the network layer (Figure 3-46):

- *Multicast addressing:* Dealing with protocol mechanisms that define how to address a datagram sent to a group of receivers.
- *Group maintenance:* Determining how to identify the receivers of a multicast datagram. The Internet Group Management Protocol (IGMP) for Ipv4 and the multicast listener discovery (MLD) for Ipv6 address this issue. These protocols operate between hosts and their immediately attached multicast routers. they enable routers to manage and maintain the group member ships. In particular, these protocols allow a host to inform its local multicast router that it wishes to receive data addressed to a specific multicast group. They also allow multicast routers to periodically query the LAN to determine if known group member are still active.
- *Multicast routing:* Defining how to route the multicast datagrams to their destinations. In comparison with IGMP, the multicast routing protocols operate between multicast routers. These protocols are used at the routers to determine the multicast spanning tree used for delivery the multicast packets to their multicast receivers.

These three aspects of the IP multicast will be illustrated in the next sections.

3.7.2.1 Multicast Addressing

A multicast address is an IP address assigned to a set of receivers that belongs to a multicast group. Senders use the multicast address as the destination IP address of a packet, which is to be transmitted to all group members. The source address in a multicast packet is still the unicast IP address.

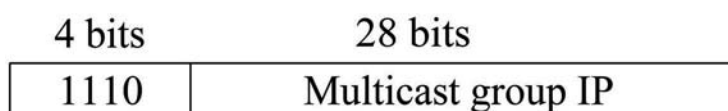


Figure 3-47: Structure of an IPv4 multicast address

An IPv4 multicast group is identified by a class D address (figure 3-47). Class D addresses have their high-order four bits set to “1110” followed by a 28-bit unicast group ID. Thus, the IPv4 multicast group addresses range from 224.0.0.0 to 239.255.255.255. The base address 224.0.0.0 is reserved and cannot be assigned to any group. The multicast addresses ranging from 224.0.0.1 to 224.0.0.225 are reserved for the use of routing protocols and maintenance protocols. Other multicast addresses are assigned to various multicast applications or remain unassigned. From this range, addresses from 239.0.0.0 to 239.255.255.255 are reserved for administratively scoped applications, and not for Internet-wide applications.

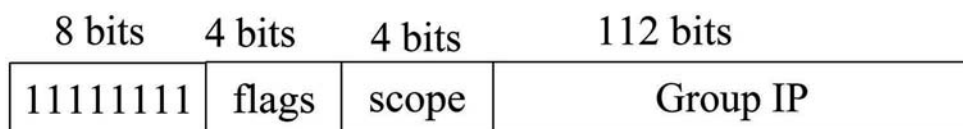


Figure 3-48: Structure of an IPv6 multicast address

The format of an IPv6 multicast address is described in figure 3-48. The 8 bits “11111111” at the start of the address identifies that the address is a multicast address. The field “flags” is a set of 4 flags “000T”. The high-order 3 flags are initialized to 0. The address is a permanently assigned as multicast address if T is set equal to 0, otherwise the address is a non-permanently assigned. The 4-bit values of the scop field are used to limit the scope of the multicast group [HD-2003]. The group ID field is used to identify the multicast group.

3.7.2.2 Internet Group Management Protocol

The existing IGMP versions for IPv4 are IGMPv1 [Dee-1989], IGMPv2 [Fen-1997] and IGMPv3 [HD-2003]. The IGMP protocol is built on top of IPv4, and the IGMP messages are specified in the IP datagram with a protocol value 2. These protocols are short summarized in this paragraph.

3.7.2.2.1 IGMPv1

There are two types of IGMPv1 messages used for host-router communication – the membership query and the membership report. The format of these two messages is shown in figure 3-49. The version field is set equal to 1. The type field is used to identify whether the message is a membership query or membership report. The checksum field is used for the whole IGMP packet. It is based on the Internet checksum method. The group address field determines the multicast group address, to which the multicast sources send their data, and from which the multicast receivers receive the data.

IGMPv1 functions as follows. When a host wants to register to a multicast group, the host sends IGMP membership reports to the group address to which the host subscribes with a TTL of 1. The multicast routers receive these IGMP reports and therefore it is informed of a new multicast group member. On each interface, a multicast router periodically sends membership query messages with TTL of 1 to all hosts that subscribe as members of multicast groups. On receiving of these query messages, each host on direct connected subnets is supposed to response with a membership report sent to each group address to which it belongs.

4 bits	4 bits	8 bits	16 bits
Version	Type	Unused	Checksum
Group address			

Figure 3-49: IGMPv1 message format

The disadvantage of IGMPv1 is that it does not provide an election mechanism for defining which router can send query messages. In IGMPv1, the design router is set to be the querier. Furthermore, there is no leave membership report in IGMPv1. The router will consider that there is no member left for this group, if the router does not receive membership reports for a group after three query messages. Thus, in the meantime, the IGMPv1 router will keep forwarding useless and bandwidth-consuming membership query datagrams.

3.7.2.2.2 IGMPv2

In comparison with IGMPv1, the IGMPv2 additionally supports a leave function that enables a host to send leave group messages as a reply to a membership query when it leaves a multicast group. This function improves the leave latency in the IGMPv1. All IGMPv2 messages have the format shown in figure 3-50. There are four types of IGMPv2 messages – membership query, membership report, version 1 membership report, and leave group. Two sub-types of version 2 membership query are the General Query and the Group-specific Query. While the first one is a query message used to learn which groups have members on an attached network, the second one is used to learn if a particular group has any members on an attached network. The max response time field defines the maximum time allowed before sending a responding report in 1/10 second. The checksum field is the same as by IGMPv1. The group address indicates the group being either queried, or reported or left.

8 bits	8 bits	16 bits
Type	Max. Resp. Time	Checksum
Group address		

Figure 3-50: IGMPv2 message format [Fen-1997]

An IGMPv2 multicast router maintains a list of multicast group memberships for each attached network and a timer for each membership. A multicast router may take one of two roles: querier or non-querier router. The IGMPv2 selection mechanism is based on these functional roles. In the beginning, all multicast routers send queries. If a router hears a query message from a router with a lower IP address, it must become a non-querier on that network. And if a router has not heard a query message from other routers, it resumes the role of the querier. Routers with the querier role periodically sends a General Query addressed to the all-systems multicast group (224.0.0.1) on each attached network to discover the membership information. When a host receives a General Query, it sets delay timers for each group that has member on the interface from which the host received the query message.

When a group's delay timer expires, the host multicasts a membership report to the group with IP TTL of 1. When a router received a membership report, it adds the group to the list of multicast group membership. When a host joins a multicast group, it sends an unsolicited membership report for that group. The leave message for a group is sent only by a host if it is the last one to reply to a query with a membership report message for this group.

3.7.2.2.3 IGMPv3

IGMPv3 [RFC 3376] additionally supports source filtering that enables a system to report its interest in receiving packets only from specific source addresses. IGMPv3 is designed to be interoperable with version 1 and 2.

In order to support source filtering, IGMPv3 adds two new message types: the membership query and version 3 membership report. To keep compatible with version 1 and 2, IGMPv3 also supports three following message types: membership report version 1, membership report version 2 and leave group report version 2. The protocol operation of these three messages is described in the previous section of IGMPv1 and IGMPv2. In this section we only focus on the membership query, membership report and the protocol IGMPv3 actions on the group members and on the multicast routers.

Membership query message

The multicast routers send query messages to request the state of the neighboring interfaces. The format of query message is shown in figure 3-51.

- The first 4 fields (type, max response time, checksum and group address) remain unchanged from IGMPv2.
- Resv (Reserved): is set to zero on transmission, and ignored on reception.
- S flag (suppress router-side processing): is used to suppress the router-side processing.
- GRV (Querier's robustness variable): contains the value used by the querier.
- QQIC (Querier's Query Interval Code): specifies the query interval used by querier
- Number of sources (N): specifies how many sources addresses are present in the query message. The number is zero in a general query or a group-specific query message, and non-zero in a group-and-source-specific query message.
- Source address [i]: a vector of the IP unicast addresses of the sources in this query message.
- Query variants. There are three variants of the query message [HD-2003]: (1) general query, (2) group-specific query and (3) group-and-source-specific query. The first one is sent by a multicast router to discover the multicast reception state of the neighboring interfaces. The second one is sent by a multicast router to learn the reception state with respect to a single multicast address. Finally, the third one is sent by a router to learn neighbouring interfaces that desire to receipt packets sent to a particular multicast address, from any of specified list of sources.

Type			Max. Resp. Time	Checksum
Group address				
	S	GRV	QQIC	
Source address[1]				
Source address[2]				
Source address[N]				

Figure 3-51: Format of the membership query message [HD-2003]

Version 3 membership report message

The version 3 membership report is sent by hosts to report to their neighboring routers about the current state or the changes in the multicast reception state of their interfaces. The report messages have the format described in figure 3-52. The new fields in the report message are [HD-2003]:

- *Number of group report*: specifies the number of group reports presented in this report.
- *Group report*: includes a block of field (figure 3-52 b) that contains information pertaining to the sender's membership in a single multicast group on the interface from which the report is sent.

Type	Reserved	Checksum	
Reserved		Number of group records(N)	
group records[1]			
group records[2]			
.....			
group records[N]			
Record Type	aux. data len	Number of sources(N)	
Group address			
Source addressn[1]			
Source addressn[2]			
.....			
Source addressn[N]			
Auxiliar data			

Figure 3-52: Format of the version 3 membership report message (a), and the format of the group record (b)

IGMPv3 functions on group members

There are two events that trigger IGMPv3 protocol actions on an interface: a change of the interface reception state and a reception of a query message:

- *A change of the interface reception state*: A change of a interface state causes multicast members to immediately transmit a state-change report message from that interface. In this case, each member determines the

contents of the group record (s) in a report message by comparing the filter mode with the source list for the affected multicast address before and after the change. The method defining how to determine the new content of a report message is described in [HD-2003].

- *Reception of a query message:* If a multicast member receives a query message, it delays its response by a random amount of time derived from the max Resp time in the received query message. For scheduling a response to a query, several states must be maintained by each member, such as a timer per interface for scheduling responses to general queries, a per-group and interface timer for scheduling responses to group specific and group-and-source-specific queries. On receiving a query message, a multicast member uses a set of rule defined in [RFC 3376] to determine if a report message needs to be scheduled and thus the type of the report message to schedule. Depending on the type and content of the received query message, the decision for issuing a new report can be maken. Furthermore, the type of report message and the content of its group record can be determined. Rules for scheduling the report messages are defined in [HD-2003].

IGMPv3 functions on multicast routers

As mentioned above, the IGMP enables the multicast routers to learn, which multicast groups are of interest to the systems attached to its neighboring networks. The IGMPv3 additionally facilitates multicast routers to find out which sources are of interest to neighboring systems. Following main tasks are performed by an IGMPv3 multicast router over each of its directly attached networks:

- *Conditioning and sending group membership queries.* A multicast router can send general queries, group-specific queries and group-and-source specific queries. General queries are sent periodically and used to build and update the group membership state of systems on attached networks. To enable all systems on a network to respond to change in a group membership, group-specific queries or group-and-source specific queries are sent. While a group-specific query is sent to make sure there are no systems that desire to receipt the traffic from a multicast group, a group-and-source specific query is sent to verify that there are no systems on a network which desire to receive traffic from a set of sources.
- *Maintaining the IGMP state.* IGMPv3 multicast routers keep state per group and per attached network. This state contains a set of records with a form {multicast address, group timer, filter-mode, a list of {source

address, source timer}}. These records are used for constructing and for conditioning the membership queries and reports.

- *Providing forwarding suggestions to the multicast routing protocols.* When a multicast datagram arrives at a router, this router has to decide whether to forward the datagram onto attached networks or not. In order to make this decision, the multicast routing protocol may use the IGMPv3 information to ensure that all source traffic requested from a sub network is forwarded to this sub network.
- *Performing actions on reception of a group membership report.* An arriving membership report message can contain the current-state records, the filter-mode-change records or the source-list-change records. When a router receives current-state records, it updates its group and source timers. When a system learns a change in the global state of a group, it sends the filter-mode-change records or the source-list-change records. By receiving these records, routers must possible change their own state to reflect the new desired membership state of the network.
- *Performing actions on reception of a group membership query messages.* By receiving a query message, a router must update the timer to reflect the correct timeout value for the queried group. Furthermore, within a subnet, routers must define a single querier that is responsible to send the queries. This is done by using the election mechanisms discussed by IGMPv2. Moreover, each router must construct specific query messages and send them. Decision for sending a specific query depends on the values of group timer, the last member query interval and on the last member query time.

3.7.2.3 Building the Multicast Distribution Trees

The multicast routers are responsible for duplicating the incoming multicast packets and sending them to the appropriate interfaces. In order to define interfaces to which a multicast packets should be sent to, each router needs to build a multicast distribution tree (MDT) connecting all routers that have attached hosts belonging to the multicast group. A MDT defines the interfaces to which the multicast packets should be sent when they arrive at a multicast router. Thus, this tree determines the path an IP multicast packet will take through the network from the sources to all receivers. Techniques for building the multicast distribution tree are for example flooding, shared spanning tree, source-based spanning tree, reverse path forwarding and prune reverse path forwarding. These techniques are described in the following graphs.

3.7.2.3.1 Flooding

The simplest method to send a packet to all members of a multicast group is to flood this packet to these routers. If a router has not seen the packet before, this router forwards this packet on all interfaces except the incoming one. Thus, flooding is very simple, but its major problem is that routers will receive duplicate packets. In order to identify duplicate packets, every router has to store an identifier for each packet it received in the past. But, this leads to overhead in a large multicast session and thus it is unacceptable.

3.7.2.3.2 Shared trees

Shared tree techniques define one multicast delivery tree for all sources sending data to a multicast group. All multicast packets sent to a multicast group are routed along the shared tree, regardless of the sources. By receiving a multicast packet, the router replicates this packet to the interfaces belonging to the shared tree except the incoming interface.

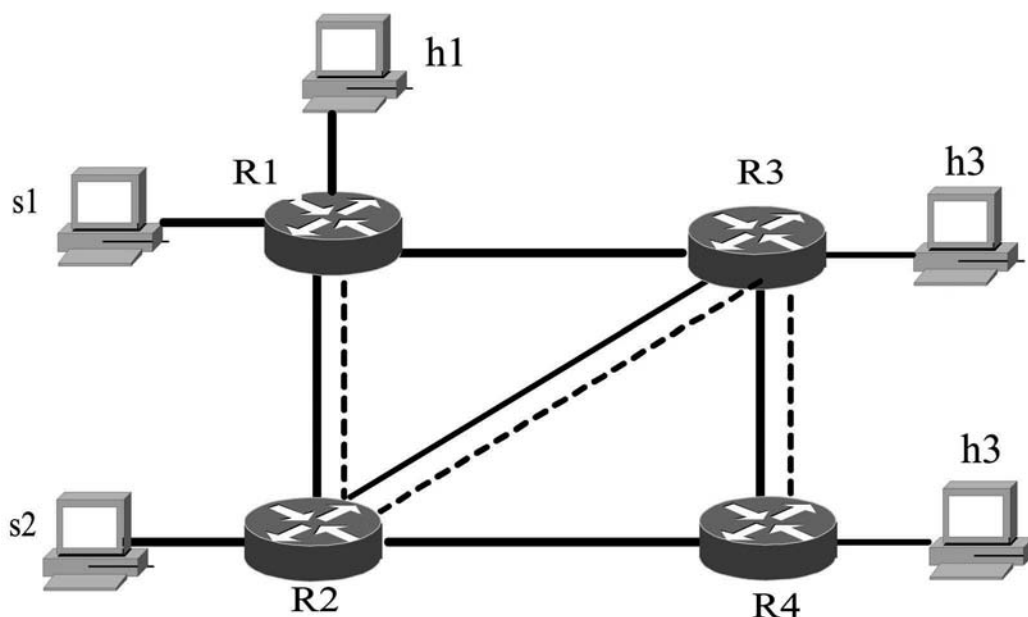


Figure 3-53: Shared tree for a multicast group with 3 receivers and two senders

Figure 3-53 shows an example of sending the multicast packets along the shared tree R1-R2-R3-R4 to a multicast group with three members {h1, h2, h3} and two senders. The multicast senders are s1 and s2. The multicast packets sent from s1 are forwarded along the path R1-R2-R3-R4 toward their receivers. The multicast packets sent from s2 are forwarded along the path R2-R1 and R2-R3-R4. These packets arrive at the receivers with no duplicate. Moreover, shared trees can centralize the multicast traffic on a smaller number of links, so

that less bandwidth will be used. The problem of this technique is that the network needs to explicitly construct the shared tree and this shared tree path may become bottlenecks.

A simple way to build a shared tree is to select a router as a rendezvous point (RP). Using RP, all sources first forward multicast packets to a directly connected router (the designated router, DR). The DR encapsulates the packets and sends them per unicast to RP, and each multicast router seeing this traffic on its way marks the link from which it arrived and the outgoing link. After that, any multicast packet received on an outgoing interface will be copied to other marked interfaces.

3.7.2.3.3 Source-based trees

Instead of defining one shared tree for all sources, source-based tree techniques build a separate multicast distribution tree for each source router. Each source-based tree is explicitly constructed using the least cost path tree from a source to all its receivers. Figure 3-54 shows an example of sending multicast packets from the sources s1 and s2 through source-based tree. Whereby, the multicast packets sent from s1 are forwarded along the source-based tree marked as long dash line, and square dot line source-based tree is used for forwarding the multicast packets sent from s2.

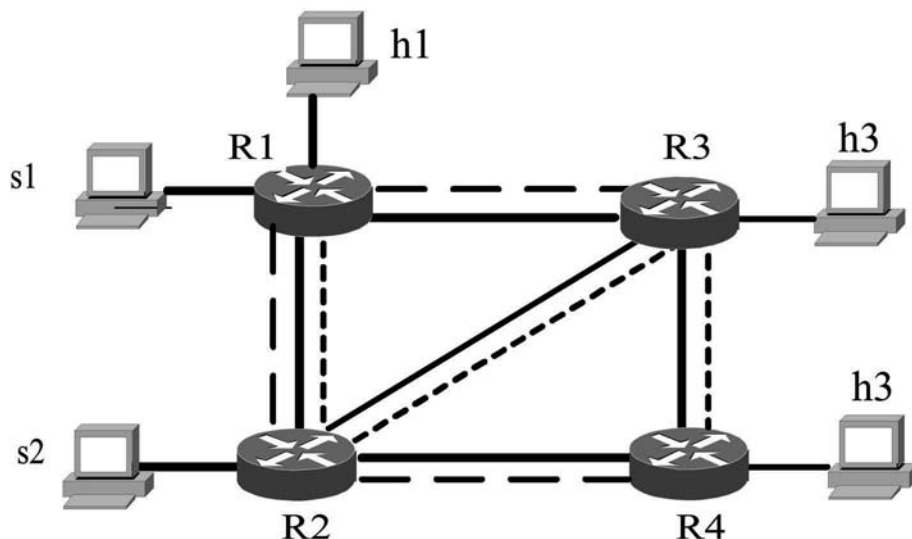


Figure 3-54: Source-based tree for a multicast group with 3 receivers and two senders

The advantage of this technique is that the multicast packets will follow the least-cost path to all receivers and there are no duplicate packets. When a host sends a packet to the group, the packet will be duplicated according to the delivery tree rooted at the host's router. This leads to smaller delivery delays. Nevertheless, this technique has a main disadvantage that the source-based tree

for each multicast sender must be explicitly set up. Therefore, the multicast routing table must carry separate entries for each source and thus the multicast routing tables can grow very large.

3.7.2.3.4 Reverse path forwarding

The reverse path forwarding (RPF) is a simple technique that avoids the overhead of storing the packet identifiers by the flooding technique. Its key idea is that a router forwards a packet from a source to all outgoing shortest path links (except the incoming one) if and only if this packet arrived on the link that is on its shortest path back to the sender. Otherwise, the router simply discards the incoming packets without forwarding them to any of its outgoing links. For example, in figure 3-55, if the router C receives a multicast packet from A, it sends this packet to F and E. But if C receives a multicast packet from B, C will drop this packet, since this packet does not arrived on a link belonging to the shortest path from the source.

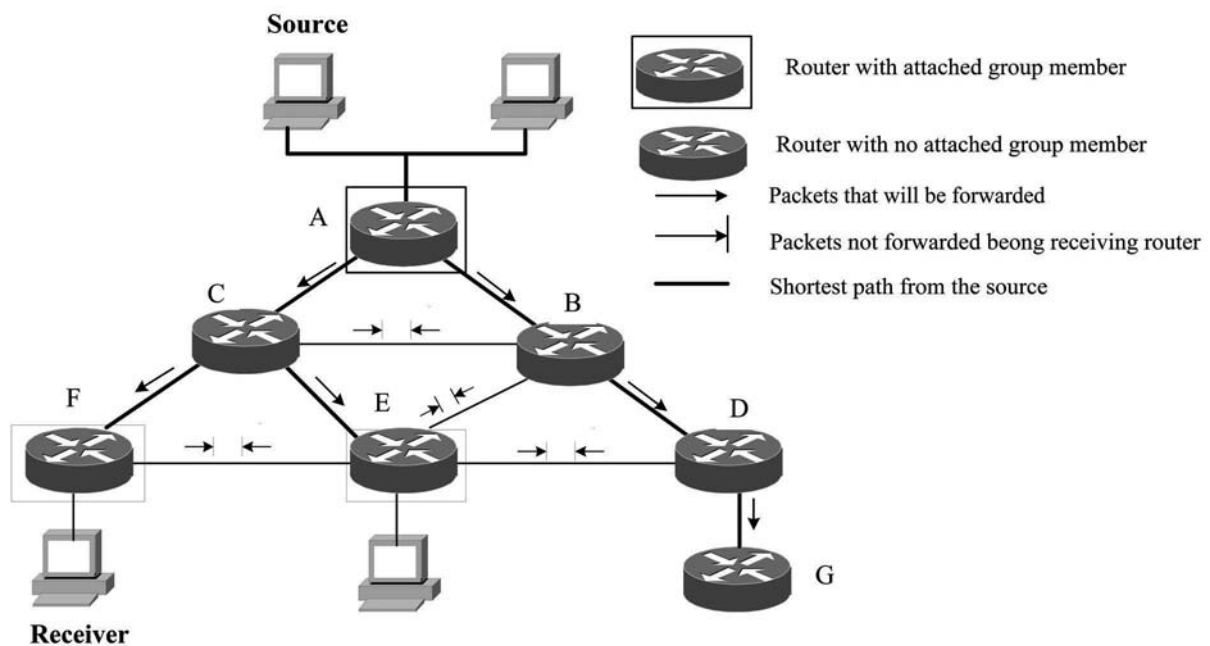


Figure 3-55: Example of the reverse path forwarding

This RPF technique is easy to implement and no packet identifier tracking is required. Moreover, a router does not require knowing the complete shortest path from itself to the source. It only needs to know the next hop on its unicast shortest path to the sender. Although the RPF technique saves storage at a router, it does not eliminate the duplicates, since source packets still go where they aren't wanted, even the subnet having no receivers.

3.7.2.3.5 *Pruning and Grafting*

The pruning technique is introduced to deal with the RPF problem, in which the multicast packets are received by every router in the network. The basis idea of pruning is to allow a router, which has no attached hosts joining to a multicast group, to inform its upstream routers in the shortest path tree that it is no longer interested in receiving multicast packets from a particular source of a particular group. If a router receives a prune message from its downstream routers, it forwards the message upstream. Prune messages allow the parent routers to stop forwarding the multicast packets down unnecessary branches for a given prune interval.

A router also has the option of sending graft messages on the parent links when its directly connected hosts join a pruned group.

3.7.3 QoS Routing

The Internet routing is primarily concerned with connectivity. Internet routing protocols (such as OSPF, RIP, BGP) mainly advertise the connectivity information, which are used by routers for calculating the shortest paths to all reachable destinations without consideration of the QoS requirements. Although when the shortest path is congested, all traffic destined to the same destination is still intended to follow this path and thus they may be delayed or even dropped by routers belong to this path. Even so, this traditional routing paradigm has been adequate for a single class of elastic applications, because they can tolerate heavy delay and losses. But for multimedia applications such as video-conferencing or telemedicine transmission, the traditional Internet routing paradigm can cause a lot of problems arising because of the delay and losses.

Real-time multimedia applications are typical less elastic and less tolerant of delay variation than elastic applications so that if the shortest path does not have enough resources to meet these requirements they must be transmitted over paths satisfying their QoS requirements (bandwidth, delay, cost, hop counts, etc.). The computation of such paths is the job of a QoS routing. Thus, to provide QoS guarantees for particular packet flows, routing should be able to determine paths which could be met the QoS requirements and at the same time maximizing the utilization of the network resources [CAN-1998, AWK-1999, ITU-2002].

The IETF QoS routing working group was established in June 1996 to discuss issues in QoS routing. This working group has been stopped at the end of 1999, because of the comprehensive understanding of the problem was still lacking. Nevertheless, QoS routing is a required functionality, because most current IETF standards rely on the traditional QoS-unaware routing. From this

perspective, QoS routing is the missing piece on QoS architecture for the Internet.

Like other Internet routing protocols, a QoS routing protocol mainly consists of two major components: QoS routing algorithm (dynamic) and path selection (static).

- *QoS routing algorithm* deals with methods for discovering the information needed to compute QoS paths. This information includes the network topology information and the information about the resource available in the network.
- *A path selection method* is an algorithm for selecting the QoS paths for all destinations that are capable of meeting the QoS requirement, and for updating and maintaining the routing tables used for selecting the QoS path for each requested flow.

These two components and the software architecture of the QoS routing within an Internet router will be discussed in this section.

3.7.3.1 QoS Routing Algorithms

Each router obtains the information about the network topology and the network resource available via exchanging the routing protocol packets with other routers in the network. Each router then maintains and updates this information in its local topology database (TDB) describing the state of all routers in a network. This state information includes the network connectivity and several metrics on which the path selection process is based. These metrics may include:

- *Input/output interface queue length* is used as a measure of the packet loss rate and the queue delay
- *Link propagation delay* is used to identify high latency links. The link propagation delay can be used while selecting a path with a delay sensitive request.
- *Link available bandwidth* is used as a measure of the bandwidth available.
- *Neighbour list* specifies a list of neighbours for each router.

To discover these metrics, the link state routing (LS) algorithm can be used. Since the traditional LS only enables the routers to exchange the neighbour list and the hop count, this routing algorithm needs to be extended for discovering additionally parameters such as link available bandwidth, queue length and link propagation delay. To exchange this additionally information with the other routers, a QoS routing algorithm must extend the link state advertisement packet (LSA) in order to advertise the metrics described above. For example, changes in the link available bandwidth metric need to be advertised as a part of the extended LSA, or changes in link propagation delay also need to be advertised

through extended LSA packets. To discover the neighbours with the hello protocol, each router needs a measurement component to monitor the queue size, the propagation delay and the available bandwidth on each link connecting to its neighbour. These parameters are sent in a hello packet together with the neighbour list.

But, the disadvantage of the link state routing and distance vector routing algorithm is that they can not guarantee the timely propagation of significance changes, and therefore they can not ensure providing accurate information for the path computation subcomponent. Updating state information whenever it changes provides the most accurate information for computing the path. But if state information changes very quickly from time to time, updating state information for each change will cause a great burden for the network links and routers – consuming much network bandwidth and router's CPU cycles. One way to solve this problem is to set a threshold to distinguish significant changes from minor changes. And the state information updating is triggered when a significant change occurs [AWK-1999].

3.7.3.2 Path Selection

Under QoS routing, the path selection deals with the routing table computation that determines a path for each flow based on some knowledges about resource availability in the network and the QoS requirement for that flow. This information is then maintained in the QoS routing table.

At each router, the path selection component uses the network topology and resource available information discovered via QoS routing algorithm for determining the paths, which will be used for forwarding packets arriving at this router to a given destination or to every other destinations. There exist several path selection algorithms. In this section, we describe the Bellman-Ford algorithm [AWK-1999] that pre-computes the QoS path for each flow, the Dijkstra-based algorithm that computes the QoS path on demand, and the standard ITU E.360.2 [ITU-2002] path selection mechanisms.

3.7.3.2.1 Bellman-Ford Algorithm

For a given source and a network topology with link metrics (link available bandwidth), the Bellman-Ford (BF) algorithm pre-computes paths with maximum available bandwidth for all hop-counts from this source to all possible destinations. The property of BF is that at each h -th iteration, the algorithm identifies the best path between the source and each destination among paths of at most h hops that have the maximal bandwidth. This best path must have a maximum available bandwidth. Specially, at the k th (hop count) iteration of the

algorithm, the maximal available bandwidth to all destinations in a path (if no more than k hops) is then recorded together with the corresponding routing information. After the algorithm terminates, this information enables the routing process to identify, for all destinations and bandwidth requirements, the path with the smallest possible number of hops with sufficient bandwidth to service the new request. This path is also the path with maximal available bandwidth, because for any hop count, the algorithm always selects one path with maximum available bandwidth.

Each router has a BF routing table that consists of a $K \times H$ matrix, where K is the number of destinations and H is the maximal allowed number of hops for a path. The $(n;h)$ entry in this routing table is determined during h th iteration of the algorithm. This entry consists of two fields (**bw** and **neighbour**):

- *bw* indicates the maximal available bandwidth on a path with at most h hops between this router and destination node h .
- *neighbour* specifies the node adjacent to this router on the path (at most h hops) to destination node n .

Based on this data structure the BF algorithm works as follows. The routing table is first initialized with all **bw** fields set to zero and neighbor fields set to empty. For each iteration h and each destination n , the **bw** and neighbor fields are copied from row $(h-1)$ into row h . The algorithm keeps a list of nodes that changed their **bw** during the $(h-1)$ iteration. The BF algorithm then looks at each link $(n; m)$ where n is a node whose **bw** value changed in the previous iteration, and checks the maximal available bandwidth on an (at most) h -hop path to the node m . This leads to select the minimum between the **bw** field in the entry $(n; h-1)$ and the link metric value $b(n;m)$ kept in the topology database. If this minimum value is higher than the present value of the **bw** field in entry $(m;h)$, then the BF has found a better path for destination m and with at most h hops. The BF algorithm then updates the **bw** field of entry $(m;h)$ to reflect this value.

3.7.3.2.2 Dijkstra Algorithm for QoS Paths

The BF algorithm described above allows a pre-computation of QoS paths. However, in many situations, such as on receiving of a request for a QoS path, selection a QoS path should be performed on demand. The Dijkstra algorithm for QoS paths can be used for the path computation on demand. For a network modelled as a graph G , the algorithm first performs a minimum hop count path computation for each requested flow and removes all edges whose available bandwidth is less than requested for the flow. After that the algorithm computes the optimal path on the remained links from the given graph.

To record the routing information, the algorithm maintains a vector t with dimension K equal to the number of destination nodes. Each entry n of this vector t consists of three fields:

- bw (*bandwidth*) indicates the maximum available bandwidth on a path between the source node s and destination node n .
- hc (*hop count*) describes the minimal number of hops on a path between the source node s and destination node n .
- nb (*neighbor*) specifies node adjacent to the source node s on that path.

Let $b(n,m)$ denotes the available bandwidth on the edge between vertices n and m , and f the bandwidth requirement for the flow. The pseudo code of the Dijkstra algorithm for QoS path computation is shown in figure 3-56.

```

Dijkstra_QoSpath( $G, t, b, f, s$ )
for (each destination  $n$  in  $t$ )do /*initialization*/
begin
     $hc(n) := \text{infinity}$ ;
     $bw[n] := \text{undefined}$ ;
     $nb[n] := \text{undefined}$ ;
end
 $h[s] := 0$ ;
 $bw[s] := \text{infinitely}$ ;
/*Compute QoS paths*/
 $S := \text{the set that contains all vertices in the graph } G$  ;
while( $S$  is not empty) do
begin
     $U = \text{vertex in } S \text{ whose value in the field } hc \text{ is minimum}$ ;
     $S := S - \{u\}$ ;
    for (each vertex  $v$  adjacent to  $u$ ) do
begin
        if( $b(u,v) \geq f$  and  $hc[v] > u+1$ ) then
begin
             $hc[v] := hc[v] + 1$ ;
             $bw[v] := \min\{bw[u], b(u,v)\}$ ;
            if( $u$  is the source node  $s$ ) then  $nb[v] := v$ ;
            else
                 $nb[v] := nb[u]$ ;
            end
        end
    end
end
end

```

Figure 3-56: Dijkstra algorithm for computing the QoS paths

3.7.3.2.3 Path Selection Algorithms in ITU-T E.360.2

The ITU-T E.360.2 [ITU-2002] recommendation describes a set of path selection algorithms used for computing the routing tables in IP-, ATM and TDM-based networks. Some of algorithms are summarized in the following:

- *Time-Dependent Routing (TDR) path selection.* The routing tables of TDR are altered at a fixed point in time during the day or week. Thus the TDR method determines the routing table based on an off-line, pre-planned basis and implements these routing tables consistently over a time period. The off-line computation determines the optimal path sets from a very large number of possible alternatives in order to minimize the network cost. Selecting a path between a source and a destination should be performed before a connection is actually attempted on that path. If a connection on one link in a path is blocked, the connection request then attempts another complete path
- *State-Dependent Routing (SDR) path selection.* In SDR, the routing tables are altered automatically according to the state of the network. For each SDR algorithm, the routing table rules are used to determine the path selections in response to changing network status (such as link bandwidth available), and are used over a relative short period. This network status information may be monitored at a central bandwidth broker processor, which then distributes this collected information to the nodes on a periodic or on-demand basis. Thus the routing tables are computed on-line or by a central bandwidth broker processor through using of the obtained network status information.
- *Event-Dependent Routing (EDR) path selection.* In EDR, the routing tables are computed locally on the basis of whether connection succeed or fail on given path choice. Its main idea is that, the path last tried, which is also successful, is tried again until blocked. If the path is blocked, another path is selected randomly and tried on the next connection request.

3.7.3.3 Software Architecture of a QoS Routing Protocol

A proposal for OSPF extension to support QoS routing is described in RFC 2676 [AWK-1999]. The software architecture of this OSPF extension is shown in figure 3-57 below. The components of this architecture are:

- *QoS routing table computation* pre-computes the QoS path for each flow and updates it in the QoS routing tables that are used by routers for selecting the QoS path for each flow.
- *QoS routing table:* contains information for finding QoS paths.

- *Core OSPF functions and topology database* is used for obtaining the network topology information including the bandwidth available and link propagation delay. Examples for such functions are the hello protocol for discovering the neighbors, the flooding protocol for sending the LSA packets.
- *Pre-computation trigger* decides whether to trigger an update or not.
- *Receive and update the QoS link state advertisement (QoS-LSA) packets*: On receiving a QoS-LSA packet, the router processes it and updates its local topology database.
- *Build and send QoS-LSA*: To inform other routers about the topology a router just know, each router builds the LSA packets and floods them to other routers in the domain.

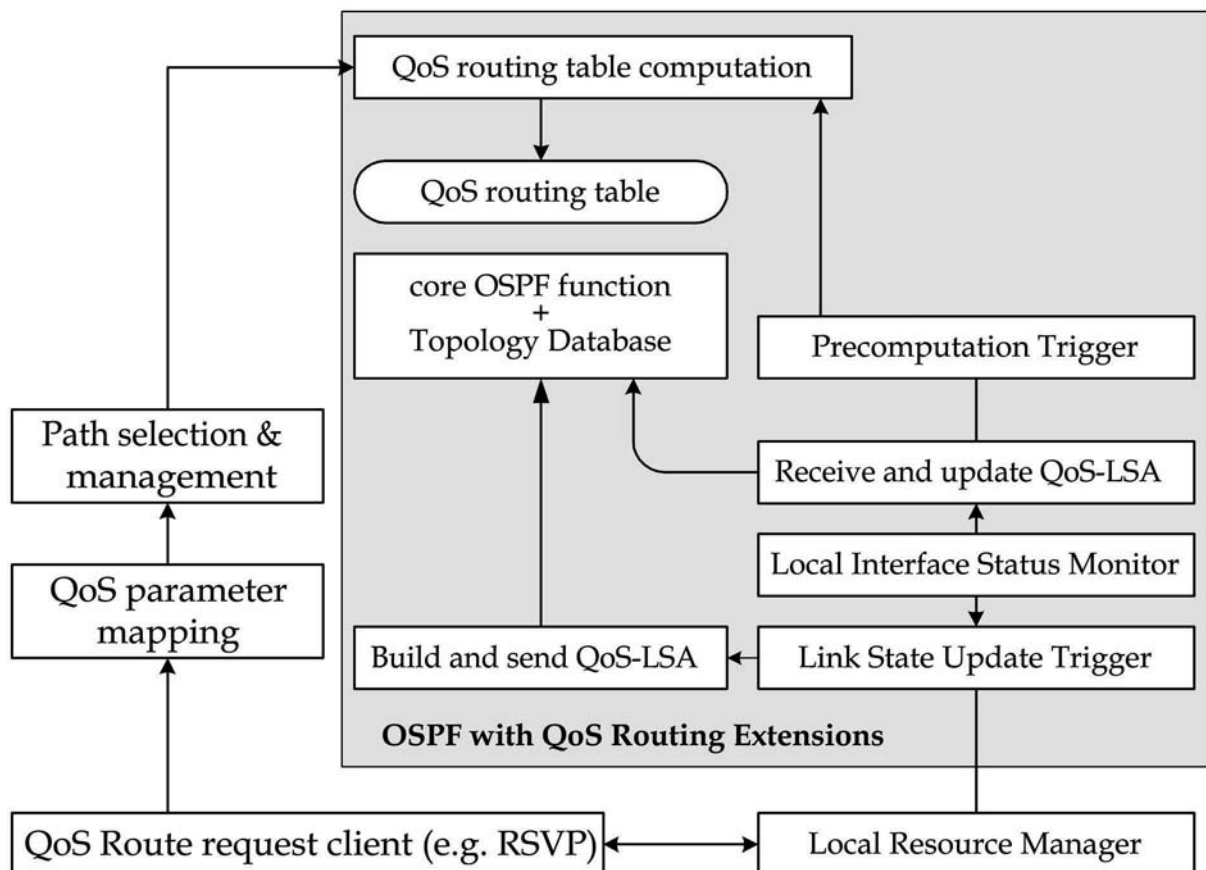


Figure 3-57: The software architecture for QoS routing by extension of OSPF [AWK-1999]

Figure 3-57 shows that a QoS routing protocol needs to work with other components, such as a local resource management to control the QoS request from client, a QoS parameter mapping to translate the client QoS parameters into the path and network QoS parameters that will be used by the QoS routing table computation.

3.8 Admission Control

As the IP technology becomes more and more basis of the Next Generation Networks, QoS is required to support real-time multimedia applications. To guarantee such QoS, capacity planning and admission control can be used. With capacity planning, the network resources (such as buffer space or bandwidth) need to be determined to carry current volumes of traffic and to meet the QoS requirements even in the busy hour. The capacity planning is done on a medium or long time scale. In contrast, admission control works on a smaller time scale. It deals with algorithms that check whether admitting a new connection would reduce the QoS of existing connections, or whether the incoming connection's QoS requirements can not be met. If either of these conditions holds, the connection is either delayed until the requested resources are available or rejected.

This section describes the existing admission controls and discusses their advantages and disadvantages. It starts with the basic architecture of an admission control. Following that, section 3.8.2 discusses parameter-based admission control. Section 3.8.3 explains the measurement-based admission control. The experience-based admission control is illustrated in section 3.8.4. Finally, section 3.8.5 presents the probe-based admission control.

3.8.1 Basic Architecture of an Admission Control

An admission control basically consists of three components: admission control algorithm, measurement process and traffic descriptors. Figure 3-58 illustrates the relationship among these three components.

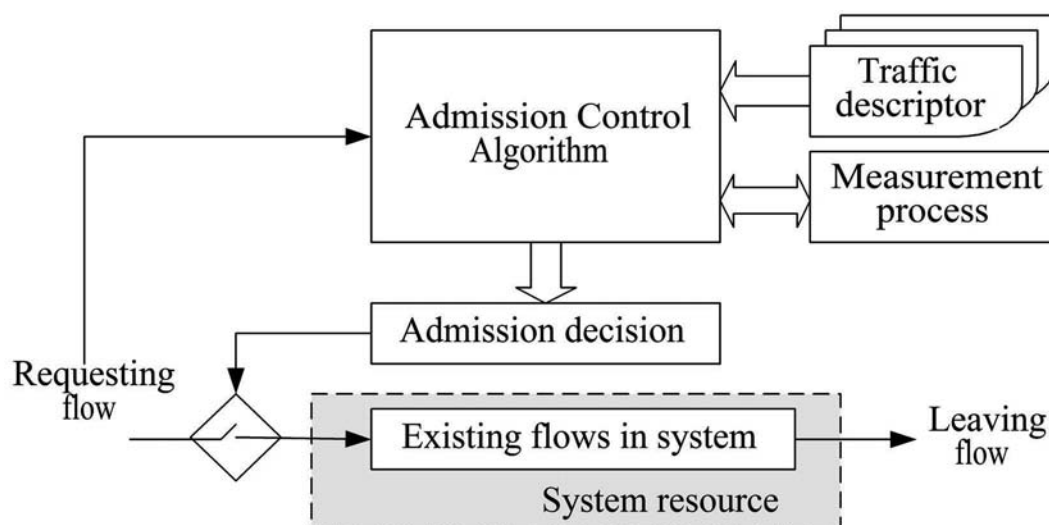


Figure 3-58: Basic components of an admission control

- *Traffic descriptor.* A traffic descriptor is a set of parameters that describe the expected characteristics of a traffic source. A typical traffic descriptor is a token bucket, which is comprised of a token fill rate r and a token bucket size b . A source described by a token bucket will send at most $r \cdot t + b$ amount of traffic over any period of t larger than packet transmission time. Sometimes a token bucket also contains a peak rate p , which constrains the smallest packet inter-arrival time to be $1/p$.
- *Measurement process.* This component can be used to estimate the traffic amount and the resource available in the system.
- *Admission control algorithms.* These algorithms use the input from the traffic descriptors and/or measurement process for making admission control decisions. Since the network resources allocated to a traffic class are shared by all flows of this class, the decision to accept a new flow may affect the QoS commitment made to the admitted flows of the particular class. A new flow can also affect the QoS of existing flows in lower priority classes, therefore, an admission control decision is usually made based on estimation of the impact that the new flow will have on other existing flows and on the utilization target of the network.

3.8.2 Parameter-based Admission Control

A simplest admission control approach is the parameter-based admission control (PBAC) [FS-2004]. It is solely based on information known before flow setup. This approach derives the worst-case bounds (e.g. packet loss, delay and jitter) from traffic descriptor (e.g. peak rate, token bucket size, maximal packet size). Therefore, an application needs to describe the traffic it is going to send. This traffic description is hold in the traffic descriptor component. There are several PBAC algorithms that will be described in the following subsections.

3.8.2.1 Constant Bit Rate Admission Control

A constant bit rate (CBR) connection i can be described by its rate $r(i)$. If a network link has a capacity C and has a load L , then a new connection i can be admitted if and only if $L + r(i) \leq C$. A CBR connection can also provide connections with delay requirement. In this case, a connection may fail the CBR admission control test if the best delay bound found is worse than the connection's delay requirement.

3.8.2.2 VBR Admission Control

In contrast to CBR connections, a variable bit rate (VBR) connection sends data in burst, so that its peak rate differs from its average rate. Admission

controls for VBR connections are difficult because VBR connections are inherently bursty. That means, VBR connections have periods where they send data at a rate which can be much greater than average rate. The basis principle of a VBR admission control is that as a link's capacity increases and it carries more and more connections, the probability that all sources simultaneously send a burst into this link becomes small. Therefore, if the number of sources is large, a burst from one source is likely to coincide with an idle period from another, so that the admission control can admit a call as if was sending a CBR stream with a rate close to its long-term average. This assumption simplifies the admission control algorithm, but it can result in delay bound violations.

3.8.2.3 VBR Admission Control with Peak Rate

The simple method for VBR admission control is to treat connections with a rate which is set equal to their peak rate. The router controller then reserves enough resources to deal with a connection. This method is correct, though potentially conservative. The other problem with peak rate allocation is that the peak rate of a connection may increase after it passes through a few schedulers because of the scheduling jitter. This problem is hard to capture analytically, unless to carefully add a fudge factor to allow for variations in the connection's peak rate.

3.8.2.4 VBR Admission Control with Worst Case

Another method, which is less conservative than peak rate allocation, is to use a scheduling to allocate resources so that a connection will meet its performance guarantees even in the worst case. For example, by using the WFQ, we can allocate sufficient bandwidth at each router so that the worst case delay along the path is bounded so that no packets are lost. This would simultaneously meet the bandwidth, delay and loss bounds. As with the peak rate allocation, worst case admission control has the potential to underutilize the network.

3.8.2.5 VBR Admission Control with Statistical Guarantee

Statistical or probabilistic performance bounds imply that the admission control method knows some thing about the statistical behaviour of the sources. It is also assumed that the sources, which share the resources, are independent. That means, if a source is sending a burst of data, it does not affect the likelihood that another source is also sending a burst. This independent assumption directly leads to a consequence that the likelihood that n sources are simultaneously bursting drop as n grows large. Thus, if the link has capacity for n burst, it can

choose to admit $N > n$ connections while keeping the probability that the link is overloaded sufficiently small.

The equivalent bandwidth is a fundamental concept of the admission control that provides connections with statistical performance guarantees. Considering is a connection that sends data into a buffer of size B with the arrival rate e . Assume that the packets on the connection are infinitely small, so that packet boundaries can be ignored and packet stream resembles a fluid. The fluid approximation is valid when the link capacity is large and packet size is small. The worse case delay for given fluid arrivals at the buffer is B/e . The equivalent bandwidth of this connection is the value e such that the probability of buffer overflow is smaller than the packet loss bound ϵ . By appropriately choosing e , a set of QoS requirements (such as connection's bandwidth, delay and loss bounds) can be met.

There are three representative approaches for equivalent bandwidths. The first approach assumes fluid sources and zero buffering. If the call loss ratio is smaller than 10^{-9} , each source has a peak rate P and a mean rate m , and sources are to be multiplexed on a link with the capacity C , then the equivalent bandwidth e of a source is determined in [Rob-1992] as follows:

$$e = 1.2m + 60m(P-m)/c \quad (3.20)$$

The second approach [GH-1991] considers the switch buffer so that the computation is more complicated. This approach assumes that sources are either on for an exponential distributed length with mean length $1/\mu$, when their rate is the peak rate p , or off for an exponential distributed interval of length $1/\lambda$, when the rate is 0. If the leaky bucket parameters (the token bucket rate ρ and the token bucket size σ) for the source are known, the parameters λ and μ are given by

$$\mu = (p - \rho) / \sigma \quad (3.21)$$

$$\lambda = p / \sigma \quad (3.22)$$

Given are sources, which share a single buffer of size B and require an acceptable packet loss ratio of ϵ . The equivalent bandwidth of the source e is given by the following equation

$$e(\xi) = \frac{\xi p + \lambda + \mu - \sqrt{(\xi p + \mu - \lambda)^2 + 4\lambda\mu}}{2\xi} \quad (3.23)$$

where the parameter ξ is defined as $\xi = (\log \epsilon) / B$.

This approach is only pessimistic when the buffer size is small. Moreover, it is valid only for asymptotically large link capacities,

The third approach proposed in [GG-1992] determines equivalent bandwidth in three steps. This approach first computes the equivalent bandwidth for an ensemble of N connections by a given peak rate, mean rate and average burst time. In the second step, the approach then computes the leaky-bucket parameters to describe an on-off source. The key idea is to choose leaky bucket parameters that minimize delay at the regulator or policer, without violating the loss probabilities at the links. In the third step, heuristic is used to model an arbitrary source with an equivalent on-off source by measuring its actual behavior at a leaky bucket regulator. The formulas for computing the peak rate, mean rate and the burst size are given in [EMW-1995].

PBAC algorithms described in this section are appropriate for providing hard QoS for real-time services. These algorithms are typically exercised over a resource reservation request for securing necessary resource for an ensuing traffic flow.

3.8.3 Measurement-based Admission Control

Measurement-based admission control (MBAC) [JEN-2004] uses measurements of existing traffic for marking an admission decision. It does not warrant throughputs or hard bounds on packet loss, delay or jitter and is appropriate for providing soft or relative QoS. MBAC basically includes two components: (1) measurement process that estimates how much traffic is in the system and how much resource remains in the system; (2) admission control algorithm that uses the inputs from measurement process for making admission control decisions. The basis principle of MBAC is described as follows: The routers monitor the current average load due to an ensemble of active connections by measuring the number of packets arrival over a fixed time interval; when a new connection appears, the router admits this connection if the sum of the measured load over the past t seconds and the load of the new connection is less than the available link bandwidth.

There are several MBAC approaches proposed in the literatures. These approaches mainly differ in their measurement and admission control decision algorithms, which will be discussed in the following sub paragraphs.

3.8.3.1 Admission Control Decision Algorithms

A number of different admission control decision algorithms was studied. These algorithms are:

- *Simple Sum (SS)*. This algorithm ensures that the sum of requested resources does not exceed link capacity. Let χ is the sum of reserved rates of the existing flows, c is the capacity of outgoing link, $r(i)$ is the rate requested by the flow i . The Simple Sum method accepts the new flow if the check (3.24) below succeeds. This is the simplest admission control algorithm and hence is being widely implemented by switch and router vendors.

$$\chi + r(i) < c \quad (3.24)$$

- *Measured Sum (MS)*. Whereas the previous algorithm ensures that the sum of existing rates plus the rate of a newly incoming connection does not exceed the link capacity, the Measured Sum algorithm [BJS-2000] uses the measurement to estimate the load of existing traffic. This algorithm admits a new flow if the test in (3.25) succeeds. Where v is the user-defined link utilization target and χ is the measured load of the existing traffic. A measured-based approach is doomed to fail when delay variations are exceedingly large, which will occur at very high utilization. Thus, identification of a utilization target is necessary needed. Moreover, the admission control algorithm should strive to keep the link utilization below this level.

$$\chi + r(i) < vc \quad (3.25)$$

- *Acceptance Region*. The algorithm derives an acceptance region from admission control proposed in [TG-1997]. Essentially, the algorithm decides whether to admit a new flow based on the current state of the system and whether the state lies within the “acceptance” region or “rejection” region. For a specific set of flows with a given bandwidth, buffer space, traffic description and flow burstiness, this admission control computes an acceptance region, beyond which no more flow of these parameter types should be accepted. This computation is based on the assumption that the call arrival process is Poisson and independent, and the call holding times are exponentially distributed. The measurement-based method of this algorithm ensures that the sum of the measured instantaneous load and of the peak rate of a new flow is below the acceptance region.
- *Hoeffding Bound (HB)*. This algorithm computes the equivalent bandwidth for a set of flows using the so called Hoeffding bound. The equivalent bandwidth of a set of flows is defined in [Flo-1996] as the bandwidth $C(\epsilon)$ such that the stationary bandwidth requirement of the set of flows exceeds this value with probability at most ϵ . In an environment

where a large portion of traffic is elastic traffic, real-time traffic rate exceeding its equivalent bandwidth is not lost but simply encroaches upon elastic traffic. The equivalent bandwidth C_H based on Hoeffding bounds for n flows is given by (3.26). Where χ is the measured average arrival rate of existing traffic and ε is the probability that arrival rate exceeds the link capacity. The admission control checks the condition (3.27) when a new flow i requests a rate $r(i)$.

$$C_H = \chi + \sqrt{\frac{\ln(1/\varepsilon) \sum_{i=1}^n (p_i)^2}{2}} \quad (3.26)$$

$$C_H + r(i) < c \quad (3.27)$$

- *Tangent at Peak (TP)*. TP computes the equivalent bandwidth from the Chernoff Bounds. Using this algorithm, a new flow is admitted if the condition (3.28) is met. Where n is the number of admitted flows, p is the peak rate of the flows, s is the space parameter of the Chernoff Bound, χ is the estimated current load, and c is the link bandwidth.

$$np(1-e^{-sp}) + e^{-sp}\chi \leq c \quad (3.28)$$

3.8.3.2 Measurement Process

In order to make an intelligent admission decision, a measurement-based admission control must provide an accurate measure of the rate of congestion and the amount of resources used in the network. There are a number of algorithms to obtain these measurements. In this section, three measurement algorithms: time window, point samples, and exponential averaging will be discussed. Note that these algorithms can be used to measure the average load of a link, average delay of packets, and other statistics needed as input for the admission control.

- *Time Window*. The time window algorithm measures the network load over a period of time. The output of this measurement algorithm is a current estimate of the network load for use by an admission control algorithm. The network load is sampled every average period (S) and the result is stored. After a window of a number of samples (T), the estimated load is updated to reflect the maximal average load seen in the previous window. Whenever a new flow is admitted to the system, the estimated load is increased according to the advertised flow information, and the window is restarted. The estimate is also increased immediately if a

measured sample is ever higher than the current estimate. Figure 3-59 graphically shows an example of the mechanism in action [BJS-2000].

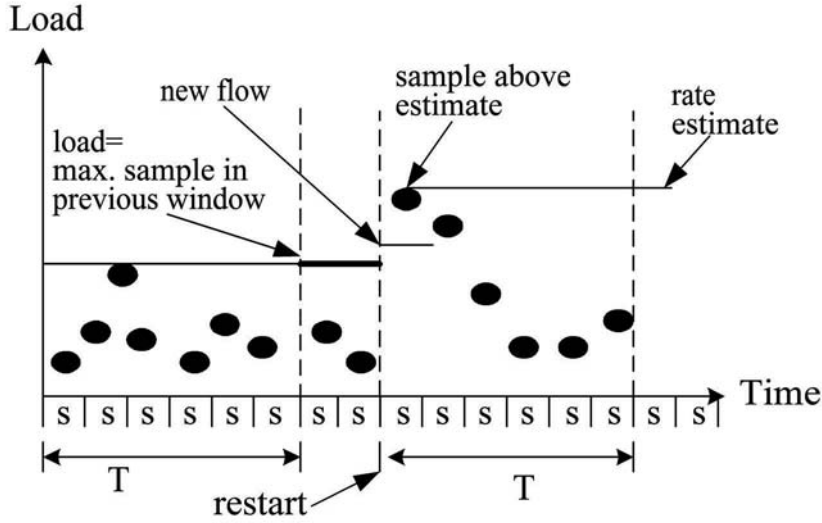


Figure 3-59: Time window measurement of network load

- *Point Samples.* This measurement algorithm is usually used with the acceptance region algorithm. It simply takes a sample of the instantaneous load every S interval and treats this measurement as the average load.
- *Exponential Averaging.* This algorithm [Flo-1996, JSD-1997] takes a sample of traffic load every S interval. The average load v' is then updated as a function of the past measurement v and the instantaneous load measurement $v(i)$ given in (3.29). Where w is an averaging weight determining how fast the estimated average adapts to the new measurements. A larger w results in a fast reaction to the network dynamics.

$$v' = (1-w)*v + w*v(i) \quad (3.29)$$

Another important factor in this algorithm is the time constant t . Given w and S , t is given by

$$t = \frac{-1}{\ln(1-w)} * S \quad (3.30)$$

The time constant t reflects the time taken for the estimated average to reach 63% of the new measurements, assuming the traffic changes from 0 to 1 abruptly. It can affect how long the measurement process will remember the past. If t is too long, the measurements will remember the flows that have already terminated long ago. On the other hand, t should not be shorter than the

interval between the time when a new flow is admitted and when the new flow's traffic is reflected in the measurements.

3.8.4 Experience-Based Admission Control

Experienced-based Admission Control (EBAC) [MMJ-2007] can be understood as a hybrid approach combining functional elements of PBAC and MBAC. EBAC uses measurements to make admission control decisions like MBAC. In contrast to MBAC, it uses information from the past instead of real-time measurements. Like PBAC, it also takes a priori flow specification into account. The goal is to achieve even higher resource utilization than MBAC by using experience rather than just momentary information. The concept of EBAC is summarized in [MMJ-2007] as follows: EBAC admits a new flow if the sum of the peak rates of all admitted flows and the new flow is not greater than the link capacity multiplied by an overbooking factor. The difference to PBAC lies in the overbooking factor, which is calculated based on the reservation utilization of the admitted flows in the past. EBAC also requires the measurement process (shown in figure 3-59) to compute the reservation utilization, but these measurements do not have real-time requirement and thus they only indirectly influence the admission control.

3.8.5 Probe-based Admission Control

All admission control approaches discussed above require some or even essential support in routers. While these approaches provide excellent QoS, they have limited scalability because they require routers to keep per-flow state and to process per-flow reservation messages. A form of host-based admission control that aims to reduce or eliminate necessary support of admission control in routers is the Probe-based Admission Control (PrBAC) [BKS-2000, CB-2001, GKP-2006, IK-2001], which makes the admission decision based on the packet-loss ratio of the probe stream. Thus, flows are accepted if the estimated packet loss probability is below the acceptance threshold P_{target} , which is fixed for a service class and is the same for all flows. The aim of this admission control is to provide a reliable upper bound on packet loss probability for accepted flows.

Figure 3-60 shows how a sender host and a receiver host cooperate to perform PrBAC. When a new flow needs to be transmitted, the sending application passes the flow request to admission control with a traffic descriptor. When receiving this flow request, the admission control starts sending probe packets to the destination host at the maximum bit rate the flow will require. The probe packets contain information about the peak bit rate and the length of the

probe, as well as the sequence number. As soon as the first probe packet arrives, the admission control at the receiving side starts measuring the packet loss. Based on the information contained in the probe packet and the measured packet loss rate P_{loss} , the receiving host can perform acceptance or rejection of the admission. In particular, when a probe period finishes and the host receives the last probe packet, it uses the the packet loss rate and the acceptance threshold to make the admission decision. For example, the flow is accepted if following condition holds [IK-2001]:

$$P_{loss} + Z_R + \sqrt{\frac{P_{loss}(1-P_{target})}{s}} \quad (3.31)$$

Where s is the number of probe packets, R is the confidence level and Z_R is the value depending on R . In order to identify different flows at the end host, the probe packet also needs to contain a flow identifier, since one sender could send more than one flow simultaneously.

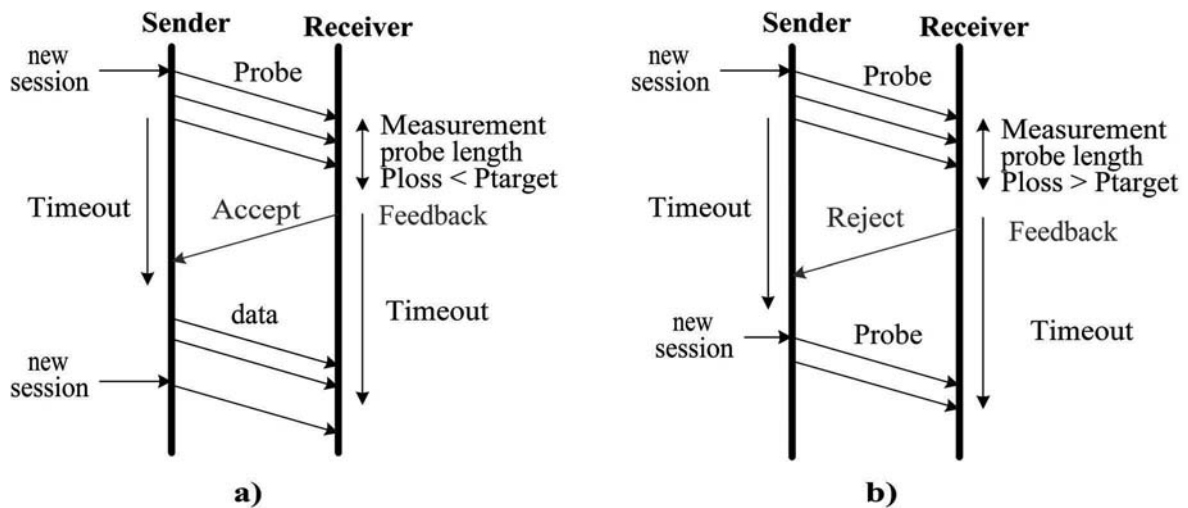


Figure 3-60: The probing procedure: a) accept and b) reject

PrBAC can be used to achieve QoS only by means of a admission control. This requires all endpoints to perform admission control, since admission control implements the QoS, specifically by limiting the packet loss of accepted flows. Also, the probe packets should be treated with low priority so that QoS of admitted flows is not disrupted. Moreover, the Accept/Reject packets should have high priority. These last requirements assume that routers do not need to perform admission control, but need to use some mechanisms that allow treating low and high priority packets.

A disadvantage of PrBAC is that many applications are not suitable for it. The probe packets need to be sent for a minimum time interval in order to estimate reliable packet loss probabilities. A voice over IP call for which the

user has to wait several seconds longer than before to get a ring tone is not desirable.

3.9 Internet Signaling

The network layer protocols especially the IP protocol only provide the best-effort service to the Internet applications. But, multimedia application traffic must be transmitted continuously or streamed. In particular, different stream classes have different QoS requirements. QoS for each of these stream classes can be guaranteed via using a set of mechanisms discussed in the previous sections, such as traffic policing, shaping, queuing and scheduling, active queue management, congestion control and admission control. Nevertheless, on the way toward destinations, packets also can take different paths, which may be congested. A solution for this problem is the QoS signaling, which refers to mechanisms for automatically negotiating the QoS with the network before transmitting the user data as well as automatically setting up and maintaining per-flow state in the routers along the path used for delivering the data. Also, a QoS signalling approach includes three main steps: (1) establishing a path through the network; (2) allocating required resource along this path and (3) maintaining the state of this path in the network nodes while delivering the user data.

There have been a number of historic attempts to develop Internet QoS signalling, primarily for the multicast, because it was believed in the past that the multicast would be popular for the multimedia communications. Several existing Internet signalling protocols and their analysis are discussed in [MF-2005]. These protocols are as follows:

- *ST-II*. The early Internet signalling protocol ST-II was developed as a reservation protocol for point-to-multipoint communication. However, being sender-initiated, it does not scale with the number of receivers in a multicast group. Moreover, the processing and maintaining of reservation state are fairly complex, since every sender needs to setup its own reservation.
- *RSVP*. Resource Reservation Protocol (RSVP) was then designed to support multipoint-to-multipoint reservation in a more efficient way. However, its scalability, complexity and ability to meet new requirement have been criticized.
- *NSIS*. The Next Steps in Signalling (NSIS) IETF working group focuses on a signalling protocol suite. In contrast to other protocols, NSIS is not a single protocol but a complete architecture of signalling protocols with well defined inter-layer APIs.

- *YESSIR*. YESSIR (Yet another Sender Session Internet Reservations) was designed after RSVP that seeks to simplify the process of establishing reserved flows while preserving many unique features introduced in RSVP. In order to reduce the processing overhead, YESSIR's proposed a mechanism, which generates reservation requests by senders. This mechanism is built as an extension to the RTCP (Real Time Transport Control Protocol). Unfortunately, this signalling protocol requires a support of applications since it is an integral part of RTCP. In particular, it requires routers to inspect RTCP packets to identify reservation requests and refreshes.
- *SIGTRAN*. The Signalling Transport IETF working group was designed to specify a family of protocols that provide the transport of packet-based PSTN signalling over IP networks, taking into account functional and performance requirements of the PSTN signalling.

This section shows the research and development in the Internet QoS signalling. The section first presents an analysis of the standard signalling protocol RSVP. After that, the most recent work on the Next Step in Internet Signalling (NSIS) protocol suite will be outlined. Finally, approaches for voice over IP signalling will be drawn.

3.9.1 Resource Reservation Protocol

RSVP (Resource Reservation Protocol) [RFC2205] is a signalling protocol used by hosts to request specific QoS from the network for particular flows. Routers also use RSVP to deliver QoS request to all nodes along the paths of flows. Moreover, RSVP is used to establish and maintaining flow-specific reservation states in routers and hosts. RSVP is a component of the QoS extensions to the Internet architecture known as integrated services (IntServ) [RFC1633].

3.9.1.1 Integrated Services

The best-effort service provided by the traditional Internet causes a large end-to-end delay and doesn't have any guarantee on QoS. To provide better services to real-time multimedia applications, the integrated services architecture (IntServ) has been developed. With integrated services, an end system can request a specific QoS, e.g. end-to-end delay, for a particular data flow. Providing this QoS by IntServ generally requires reservation of the network resources in routers along the data path (s) and in the end hosts. In order to allow QoS supporting, each node of an IntServ network must allow following main functions:

- *Reservation setup.* A reservation setup protocol is used to deliver QoS request originating in an end-system to each router along the data path. For an IntServ network, the RSVP was designed to be the reservation setup protocol.
- *Admission control.* The RSVP is used as admission control in IntServ nodes. At each node along the path the RSVP process passes a QoS request (flowspecs) to the admission control component to allocate the resources on nodes and links to satisfy the requested QoS.
- *Policy control.* Before a reservation can be established, the RSVP process must also consult policy control to ensure that the reservation is administratively permissible.
- *Packet scheduler.* If the admission control and policy control are both succeed, the RSVP process installs the flow state (flow specs) in the local packet scheduler. The packet scheduler at each router uses this state information for allocating the bandwidth needed for each flow so that the requested QoS will be met. The packet scheduler multiplexes packets from different reserved flows into the outgoing links, together with best-effort packets.
- *Packet classifier.* The RSVP process also installs the flow state (Filter specs) in the packet classifier component, which sorts data packets, forming the new flows into appropriate scheduling classes according to the QoS reservation. The state information required for selecting packets for a QoS reservation is specified by the filter specs.

These components and their relations to the RSVP are shown in figure 3-61 and 3-62.

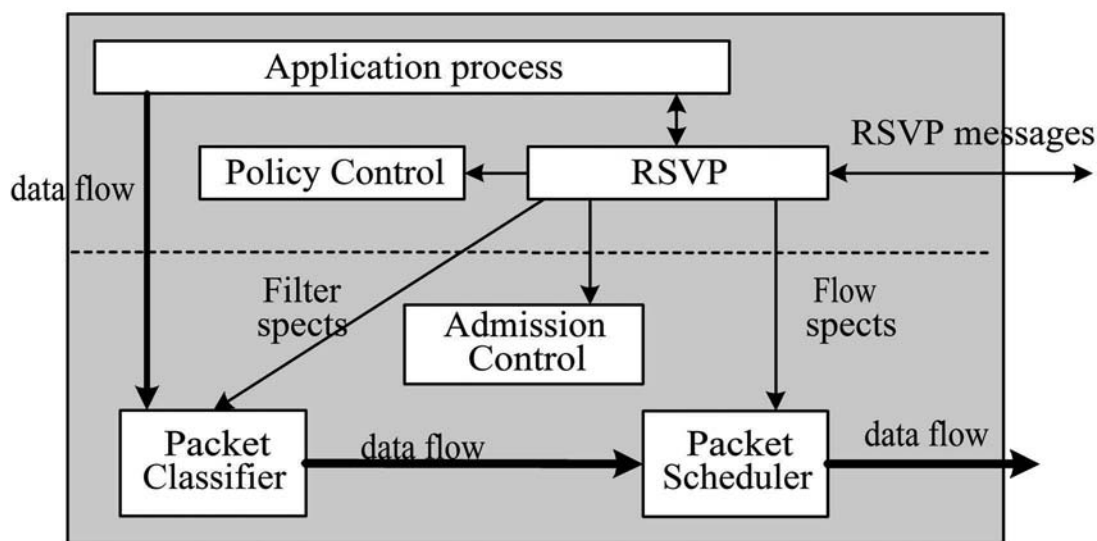


Figure 3-61: RSVP and the logical architecture of an IntServ host

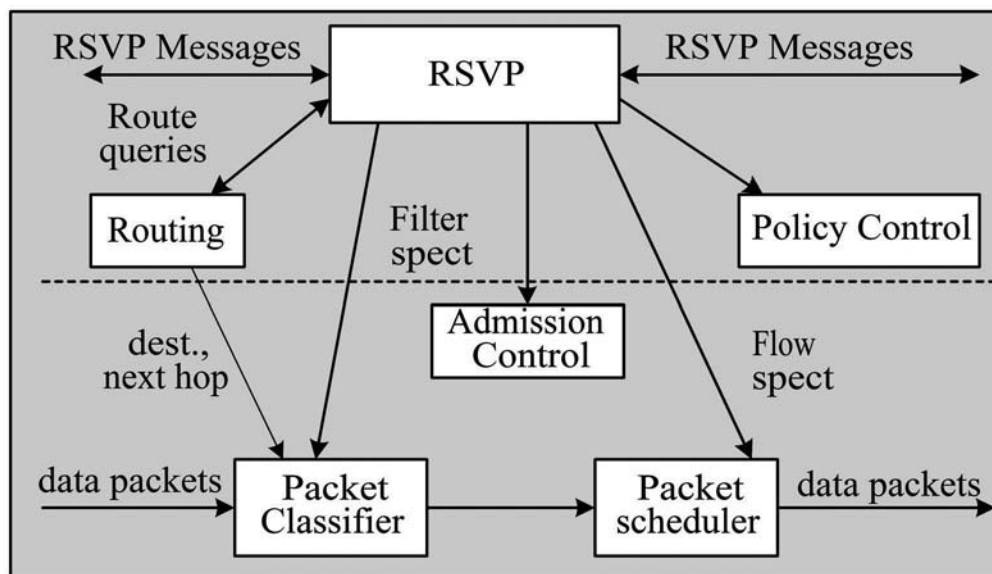


Figure 3-62: RSVP and the logical architecture of an IntServ router

3.9.1.2 RSVP Architecture

RSVP was designed to provide robust, efficient, flexible and extensible reservation services for both unicast and multicast data flows. These design requirements led to a number of basis architectural features [MF-2005]:

1. Multipoint to multipoint communication model. RSVP was designed from the beginning to support multicast as well as unicast data delivery. The RSVP communication model enables a unidirectional distribution of data from 1 sender to m receivers. Figure 3-63(a) illustrates the delivery of data packets in a multicast session from two sender hosts S1 and S2 to three receiver hosts R1, R2 and R3. This communication model allows data flows from different senders for the same session to arrive at a common node (e.g. router A). To keep a variety of applications, RSVP provides several reservation “styles” for sharing reservation among such flows.

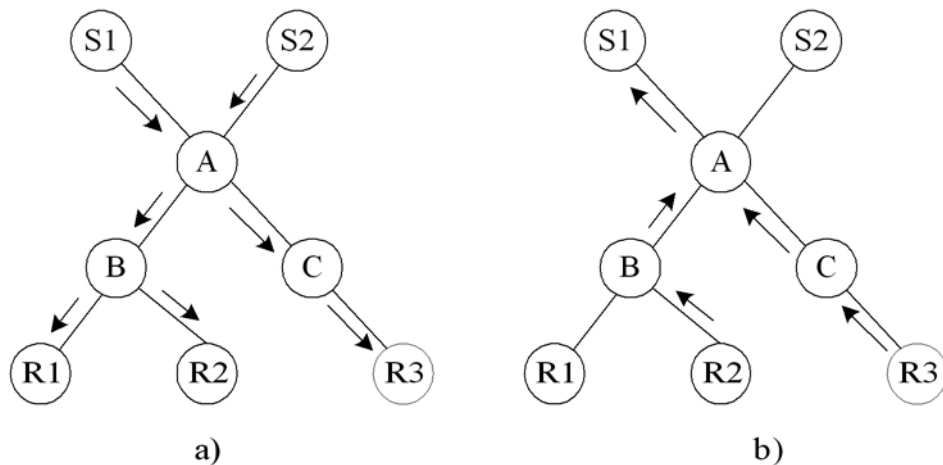


Figure 3-63: Delivery of data packets and reservation request

2. *Receiver-initiated reservation.* A major concern of RSVP design is to scale well for a large number of receivers. To achieve this requirement, receiver-initiated reservation is used. A receiver initiates an RSVP reservation request at a leaf of the multicast spanning tree; this request then travels towards the sender, as shown in figure 3-64 b). If another receiver's reservation for the same session already exists at a multicast router, this new request is merged with the previous reservation, and only a single request travels upstream.
3. *Soft state management in routers.* To achieve robustness and simplicity, RSVP takes soft state approach in its design that enables RSVP to create and remove the protocol state (Path and Resv states) in routers and hosts incrementally over time. Soft state times out without being periodically refreshed. The endpoint hosts must then periodically re-initiate the same RSVP control messages. This makes RSVP possible to automatically adapt to routing changes, link failures, and multicast group membership changes.
4. *Separation of reservation from routing.* Since multicast forwarding must function whether or not there is a reservation, the natural modularity is to separate reservation from routing, making RSVP a pure reservation setup protocol. The minimum functionality required by routing to support reservations is the answering of RSVP queries for the next hop for a given destination address. This allows RSVP to make its reservations along the data path, while route computation and installation are left to the routing protocol itself.

3.9.1.3 RSVP Signaling Model

The RSVP signalling model is based on a special handling of multicast. The sender of multicast flows periodically advertises the traffic characteristic to the receivers via “Path” messages. Upon receipt of an advertisement, a receiver may generate a “Resv” message to reserve resources along the flow path from the sender. Figure 3-64 illustrates this reservation setup process. There are two basis reservation setup models in RSVP:

- *One Pass*: a sender sends its traffic specification (Tspec) to the destination. In this model, there is no support for path-characteristic indication to the sender
- *One Pass with Adspec*: in this model, a sender sends its Tspec together with AdSpec to the routers along the path toward destination in a “Path” message. Routers look at Tspec and forward it further along with AdSpec, which advertises the capacities and available resources of routers along the path. Based on Tspec and AdSpec received along with the receiver’s own requirements, a QoS reservation request message (“Resv”) is generated by the receiver.

RSVP also periodically sends refresh messages (“Path” and “Resv”) to maintain the protocol states in the routers and hosts, and to recover from occasional lost messages. In the absence of refresh messages, the RSVP states automatically time out. These states are then deleted. States may be explicitly deleted by using “pathTear”, “PathErr” and “ResvTear” messages.

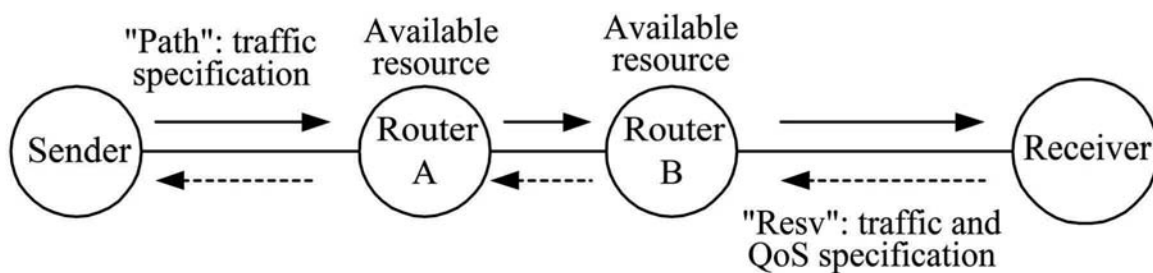


Figure 3-64: RSVP signalling setup

3.9.1.4 RSVP Messages

RSVP has seven messages: Path, Resv, Resv confirmation, Path error, Resv error, Path tear and Resv tear. These RSVP messages travel hop-by-hop. The next hop is defined in the routing table. The routers remember where the messages came from and thus they can maintain the state. Also the RSVP messages are sent via a raw IP datagram (port 46). The IP packets have the

router alert option set in the IP header. This option signals the routers that this message needs a special processing. The RSVP messages are shortly described in the following. More details about these messages is found in RFC 2205 [BZB-1997].

- *Path*: The source transmits “Path” messages every 30 seconds hop-by-hop toward destination. The forwarding decision is based on local routing tables built by routing protocols such as OSPF. At least, each “Path” message contains the IP address of each previous hop (PHOP) that is used for subsequent “Resv” messages. “Path” messages also carry the sender template, sender Tspec and AdSpec. The “sender template” field contains the data format, source address, and the port number that uniquely identifies the source’s flow from other RSVP flows. The “Sender Tspec” field describes traffic characteristics of data flows that the sender will generate. The “AdSpec” field is used to save the cumulative summary of QoS parameters, such as property of the path or availability of QoS. The “AdSpec” field is modified by a router only if the available resource or capacity to provide a service is less than what is specified in incoming “Path” message’s AdSpec field.
- *Resv*: Receivers must join to a multicast group to receive the path messages. To do this, receivers generate reservation request (“Resv” message) based on Tspec and AdSpec received together with the receiver’s own requirements, and sends this back to the previous hop to actually request the resource. A “Resv” message may include the reservation style and the flow specification. The reservation style is used to identify individual senders, group senders or all senders of a session. The flow specification field carries information necessary to make the reservation request from the receivers into the network. Attributes of the flow specification may be token bucket parameters, peak rate and maximum packet size. The “Resv” messages carry reservation requests hop-by-hop from receivers to the sender, along the reserved paths of data flow for a RSVP session.
- *Resv confirmation*: is used by the sender to inform the receiver that its reservation request has been satisfactorily installed. The “Resv confirmation” messages are directly sent to the receiver.
- *Path error*: is used to indicate an error in the processing of “Path” messages. The “Path error” message is sent hop-by-hop to the sender.
- *Resv error*: is used to indicate an error in processing of “Resv” messages. The “Resv error” message is sent hop-by-hop to the receivers.
- *Path tear*: is explicitly generated by the senders or by the routers after timeout of the path state in the node along the path. The “Path tear”

message is sent to all receivers and will immediately remove the RSVP path state.

- *Resv tear*: is explicitly generated by the receiver or any node in which the reservation state has time out. The message is sent to all pertinent senders to notify them to free up resources to be used by other flows.

3.9.1.5 RSVP Transport Mechanism Issues

Following issues in relation to transport mechanisms of RSVP are discussed in RFC 4094 [MF-2005]:

5. *Messaging Reliability*. RSVP messages are defined as a new IP packet type. For intercepting the “Path” messages, a new IP router alert option was introduced in RFC 2113 [Kat-1997]. This design is simple to implement and efficient to run. However, RSVP does not have a good message delivery mechanism. If a message is lost, it will be retransmitted only one soft-state refresh interval later, which is 30 seconds by default. To overcome this problem, a staged refresh timer mechanism was introduced in RFC 2961 [BGS-2001] as a RSVP extension. This mechanism retransmits RSVP messages until the receiver acknowledges. Thus it can address the reliability problem in RSVP.
6. *Message Packing*. Each RSVP message can only contain information for one session. In a network with a large number of RSVP sessions, this limitation poses a heavy processing burden on routers. Processing too many individual messages can easily cause congestion at socket I/O interfaces. To handle this problem, the message bundling mechanisms was introduced in RFC 2961. Since processing small packets takes almost as much CPU overhead as processing large ones, the bundling mechanisms packs multiple RSVP messages between two adjacent nodes into a single packet.
7. *MTU Problem*. Message fragmentation and reassembly are not supported by RSVP. If the size of a RSVP message is large than the link MTU, the message will be fragmented. Since the routers simply cannot detect and process fragments of RSVP messages, these fragments will be lost. There is no solution for this problem.

3.9.1.6 RSVP Performance

Performance of RSVP can be characterized as the processing overhead and the bandwidth consumption that will be described in this section.

- *Processing Overhead.* Processing overhead is the amount of processing required to handle messages belonging to a reservation session on a specific network node. A main factor that has an impact on the RSVP performance is the complexity of the protocol. Firstly, RSVP itself is per-flow based. Thus the number of states is proportional to the number of RSVP sessions, where “Path” and “Resv” states have to be maintained in each RSVP router for each session. Secondly, RSVP optimizes various merging operations for receiver-initiated multicast reservations and adds other mechanisms (such as reservation styles, scope object) to handle the multicast. These features not only cause sources of failures and error, but also complicate the state machine. Third, possible variations of the order and existence of the objects inside of the RSVP messages increase the complexity of the message parsing. It is obvious that the design of RSVP imposes limitation on its performance.
- *Bandwidth Consumption.* Bandwidth consumption indicates the amount of bandwidth used during the lifetime of a session. In particular, it defines the bandwidth needed to set up a reservation session, to keep the session alive and finally to close this session. The following formula [MF-2005] is used to calculate the bandwidth consumption in bytes for RSVP session lasting n seconds.

$$F(n) = (bP + bR) + ((n/R_i) * (bP + bR)) + bPt \quad (3.32)$$

Where bP is the IP payload size of the “Path” message, bR is the IP payload size of the “Resv” message, bPt is the IP payload size of the “Path tear” message and R_i is the refresh interval.

3.9.1.7 RSVP Security

The basis security issue in RSVP is the identification of the application and the message owner of the communication process. Furthermore, this information should securely be stored in the RSVP messages. RFC 3182 [YYP-2001] proposes mechanisms to store such information in the POLICY_DATA objects and specifies the encoding scheme.

To improve the security aspect, the authentication should be held in cryptographic manner. This is achieved by using of existing authentication and key exchange protocols. Moreover, a protection against message modification and forgery is described in RFC 2747 [BLT-2000]. However, proposed techniques do not guarantee protection from message deletion. Also, the two-way peer authentication and key management procedures are missing in

current RSVP security scheme. The security issues have been well analyzed in [TG-2005].

3.9.1.8 RSVP Mobility Support

Two issues needed to be considered when a mobile node (MN) uses RSVP: (1) flow identifier and (2) reservation refresh. The first issue relates to the location change of MN, since a MN needs to change one of its assigned IP addresses (an IP address by which MN is reachable by nodes outside the access network, and an IP address used to support local mobility management). The second issue relates to the movement of a MN. The solutions to these problems usually deal with additional RSVP extensions to allow for more seamless mobility:

- One solution proposed in [MSK-2006] is to treat the handover event as a specific case among regular RSVP operations. This extension allows a mobile node to initiate a request for a downstream reservation in the handover situation.
- Another solution is Mobile RSVP (MRSVP) [MH-2000, TBA-2001], which introduced the concept of advanced resource reservation. Instead of waiting until an MN moves to a new subnet, MRSVP makes advanced resource reservations at multiple potential locations to save time for session establishment.

3.9.2 Next Steps in Signaling

By now it should be clear that the RSVP is not dealing with many of tasks it was designed to deal with. Moreover, even with a limited set of functions, it is not able to fulfil all possible usage scenarios. The argument in defense of RSVP might be the fact that it was designed and implemented in middle 1990s. At that time the security demands were lower and there was no strong need in resource reservation over wireless and mobile networks. Therefore, QoS demands for mobile multimedia applications were negligible.

The weaknesses of the RSVP on the one hand and the growing requirements to QoS levels in different network environments on the other hand led scientists and developers to consider other approaches to support application signalling in the network. The IETF Next Step in Signaling (NSIS) working group is carrying out a research on a protocol suite of the same name. NSIS [HKL-2005] is not a simple protocol but a complete signalling protocol stack that includes several protocols for signalling information about data flow along its path in the network. By implementing such a protocol stack, it should be possible to deal with all kinds of available signalling applications.

NSIS is an ongoing research activity and at the present moment, it deals with the following basic concepts:

- *Signalling is independent from routing.* Just like RSVP, any NSIS protocol suite is not a routing protocol, but it is designed to work with any existing routing protocols to perform message forwarding tasks.
- *Path-coupled signalling.* NSIS uses “path-coupled signalling”, which involves only network elements located on the data path taken by a particular flow.
- *Unicast data flows only.* Unlike RSVP, NSIS does not support multicast. That reduces the complexity for the majority of user applications which are unicast.
- *Complete end-to-end deployment is not required.* It is not required for every node along the stream path to be NSIS enable. However, the signalling application performance highly depends on the portion of supported nodes along the stream path.
- *Signalling protocol stack.* NSIS introduces a simple protocol stack to decompose generic signalling and application specific signalling. The NSIS protocol stack is specified in RFC 4080 [HKL-2005].

3.9.2.1 Requirements for NSIS

Two general goals for the signalling solution are that it should be applicable in very wide range of scenarios, and at the same time it should be lightweight in implementation complexity and resource consumption requirements. This design goals lead to several NSIS main requirements defined in RFC 3726 [Bru-2004]:

1. *Architecture Design Goals.* NSIS needs to provide availability information on request and to separate between signalling protocol and control information being transported. Furthermore, network control mechanisms and signalling mechanisms must be independent implemented, e.g. in the case of QoS signalling the independent of the signalling protocol from the QoS provisioning allows for using NSIS together with various QoS technologies in various scenarios.
2. *Signalling Flows.* The NSIS must support path-coupled signalling and work in various scenarios such as host-to-network-to-host, edge-to-edge. Moreover this protocol should allow for hiding the internal structure of the NSIS domain (such as topology information) from end-nodes.
3. *Messaging.* Explicitly deleting the state along a path that is no longer necessary needed and automatically releasing the state after failure must be supported by each NSIS protocol. Moreover, NSIS needs to

allow sending the notification upstream, as well as notifying the state setup establishment and refusal.

4. *Control Information.* These requirements relate to the control information that needs to be exchanged. In particular, NSIS should allow adding and removing the local domain information, and addressing the state independent of the flow identifier. Also, seamless modification of already established state should be performed. Moreover, to optimize the setup delay, NSIS may group the signalling information of several micro-flows into one signalling message.
5. *Performance.* Scalability is always an important requirement for signalling protocols. In particular, NSIS must be scalable with numbers of received messages, of hand-offs in mobile environments and of interactions for setting up state. Also, scalability in amount of state per entity and in CPU usage must also be achieved. Moreover, performance requirements for NSIS also deal with the ability to keep low latency, low bandwidth consumption and highest possible network utilization.
6. *Flexibility.* Flow aggregation must be allowed by NSIS. Moreover, flexibility in the placement of NSIS initiators/responders and in initiation of state change must be supported.
7. *Mobility.* Since the handover is an essential function in wireless networks, NSIS must allow efficient service re-establishment after handover.
8. *Interworking with other protocols and techniques.* NSIS must work with other existing protocols, such as, IP tunnelling, IPv4, IPv6, seamless hand-off protocols and with traditional routing protocols.

3.9.2.2 NSIS Framework

In order to achieve a modular solution for NSIS requirements discussed in the previous section, the proposed NSIS framework [HKL-2005] consists of two layers: a “signalling transport” layer and a “signalling application” layer. The signalling transport layer is responsible for moving signalling messages around, which should be independent of any particular signalling application. The signalling application layer contains functionality such as message formats and sequences, specific to a particular signalling application. The relationships of these two layers are shown in figure 3-65 [HKL-2005].

Figure 3-65 shows the NSIS layered model overview. Here, NSIS transport layer is the generic layer that aims to support all of the common functionality for any signalling application. By succeeding in this, it is possible to achieve the desired level of abstraction on signalling applications. This will allow unifying

the treatment of all signalling applications, which reduce the architectural complexity and simplifies the configuration of signalling enable nodes. NSIS signalling layer is determined by specific signalling applications deployed on the network node. These specific applications are e.g. applications that require node configuration such as state setup. From the framework shown in figure 3-65, it is clear that both these layers interact through a well-defined API.

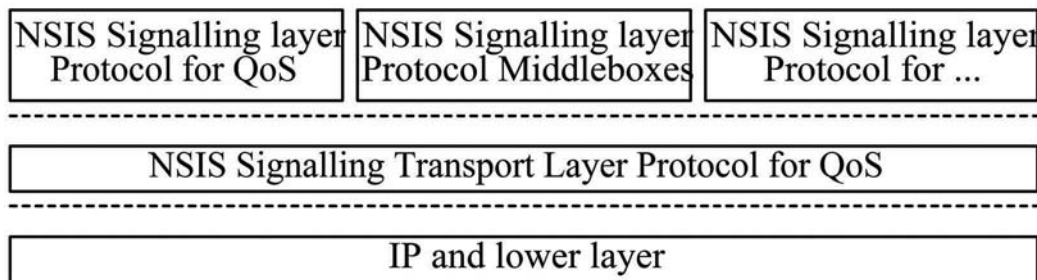


Figure 3-65: NSIS layer model overview

The basis working pattern for NSIS framework can be summarized in the following. When a signalling message must be sent by a signalling application, it is passed to the NSIS transport layer protocol (NTLP) with all necessary information included. The responsibility of the NTLP is to forward this message to the next node along the path toward the destination. In this sense, NTLP operates only between adjacent nodes and can be seen as “hop-by-hop” protocol. Respectively, when a signalling message is received, NTLP can either forward it to the recipient or pass it upwards along the protocol stack for further processing on the local node if there is an appropriate signalling application installed on this node. The signalling application can then decide to generate another message that needs to be forwarded to the next node. By this way, a larger-scope message delivery such as end-to-end delivery is achieved.

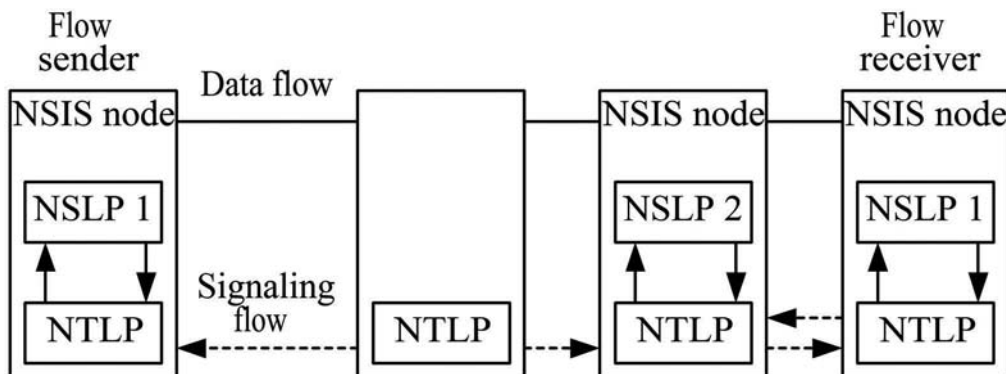


Figure 3-66: Signalling with the heterogeneous nodes

Considering the variety of possible signalling applications, it is probable that not all network nodes will support the specific NSIS signalling layer protocol

(NSLP). When signalling messages traverse such NSIS-aware intermediate nodes, they should be possibly processed at the lowest level, i.e. at the IP or at the NTLP layer. The NSIS-unaware nodes will forward messages further. The visualization of this situation is shown in figure 3-66.

In the RFC 4080, a marking at the IP layer using the router alert option is proposed to distinguish between the processing at IP and at NTLP layer. In latter case, NTLP could process the message but determines that there was no local signalling application it was relevant to. Afterwards, the message will be returned to the IP layer unchanged for further forwarding.

The complete signalling solution is the result of joint NTLP and the signalling layer protocols cooperation. In the next following sections, both NSIS layers and their mechanisms will be described.

3.9.2.3 NSIS Transport Layer Protocol

As specified by the NSIS framework, the NTLP includes all functionalities below the application signalling layer and above the IP layer. Since the overall signalling solution will always be the result of joint operations of both the NTLP and the signalling layer protocols (NSLPs), NTLP functionality is essentially just efficient for upstream and downstream peer-to-peer message delivery, which includes the ability to act for locating and/or for selecting which NTLP peer to carry out signalling exchanges for a specific data flow. It can be an active process based on specific signalling messages or a passive process operating in a particular addressing mode. This section starts with a discussion about the fundamental functionality of NTLP. Based on this, General Internet Signalling Transport (GIST) [SH-2007] will be illustrated as a concrete implementation of NTLP.

3.9.2.3.1 Fundamental Functionality of NTLP

The required functionalities of NTLP discussed in RFC 4080 [RFC4080] are:

- *State management functionality.* Internet signalling requires management maintenance of communication states within the network. While communicating with NSLP layer, NTLP passes state-management related information about up/down state of communication peers. To discover such information, it should be able to communicate with the NTLP layer of a peer node. Conceptually, NTLP provides a uniform message delivery service that is unaware of the difference in state semantics between different types of signalling application messages. An NTLP instance processes, and if necessary, forwards all signalling application messages immediately. This means that NTLP does not know explicit timer or

message sequence information for the signalling application; signalling application messages are passed immediately through an NSLP-unaware node. Specifically, it is both possible to integrate refresh messages into the signalling application protocol or to integrate it with the NTLP implementation as a generic soft-state management toolbox.

- *Addressing.* There are two ways to address a signalling message being transmitted between NTLP peers: peer-to-peer and end-to-end. The peer-to-peer addressing means that the message is addressed to a neighbouring NSIS entity that is known to be closer to the destination NE. By this way, an NE will determine the address of the next NE based on the payload of the message. This requires the derivation of the destination NE from the information present in the payload, either by using some local routing tables or through participation in active peer discovery message exchanges. In the case of end-to-end addressing, the message is addressed to the flow destination directly and intercepted by an interleaving NE. The routing of the messages should follow exactly the same path as the associated data flow.
- *Classical transport functions.* Since NSIS signalling protocols are responsible for transporting signalling data around the network, functionality such as congestion management and reliability is required. If these functions are needed, the message fragmentation should be provided by the NTLP as a service to the upper NSLP layer. To avoid the overhead of reassembly on the intermediate nodes, the fragmentation scheme used should assume the independent forwarding of separate fragments to the target node. NTLP may support message binding for short messages as an option. However message unbinding should always be performed on NTLP layer to reduce overhead caused by negotiating this feature as an option.
- *Lower layer interfaces.* The NTLP interacts with lower layer of the protocol stack for the purposes of sending and receiving signalling messages. This function places the lower boundary of NTLP at the IP layer.
- *Upper layer services.* The NTLP offers transport-layer services to higher-layer signalling applications for sending and receiving signalling messages, and for exchanging control and feedback information.
- *Identity element.* This function enables network devices for identifying flows, sessions and signalling applications. The flow identification is a mechanism for uniquely identifying a flow. Its main purpose is to provide enough information used for treatment of flows. Session identifier provides a method to correlate the signalling about the different flows with the same network control state. To be useful for mobility support, the

session identifier should be globally unique, and it should not be modified end-to-end. Signalling application identification deals with mechanisms for identifying which type a particular signalling message exchange is being used for. This identification is needed for processing of incoming messages and of general messages at an NSIS-aware intermediate node.

3.9.2.3.2 General Internet Signalling Transport

For the NTLP layer, there exists a concrete implementation – the General Internet Signalling Transport (GIST) [SH-2008]. From the protocol positioning in the stack it becomes clear that GIST does not handle signalling application state itself. In that respect it differs from application signalling protocol such as RSVP, SIP or control component of FTP. Instead, GIST manages all signalling messages on the node for upper layer signalling applications and is responsible for configuration of underlying security and transport protocols. Basically, it tries to ensure the transfer of signalling messages on behalf of signalling applications in both directions along the flow path. To perform these tasks, GIST maintains and manages its internal state.

As it was already discussed, NSIS framework does not hinder NTLP layer to be itself decomposed in functional sub-layers. GIST exploits this possibility and introduces internal layering presented in figure 3-67. Here we can see the detailed picture of NTLP protocol stack when using GIST. Basically, it shows that GIST can operate on different transport protocols and use existing security schemes like TLS or IPsec. GIST messaging layer consists of two logical components: GIST encapsulation and GIST state maintenance. GIST encapsulation deals with wrapping and unwrapping signalling messages. All the decisions done by GIST are based on its internal state, which is managed by the state maintainer and the current message content. GIST identifies routing state for upstream and downstream peers by triplet (MRI, NSLPID, SID):

- *MR1 (Message Routing Information)* describes the set of data item values used to route a signalling message according to a particular message routing method (MRM). For example, for routing along a flow path, the MRM includes flow ID, destination address, upper layer protocol, and port numbers. Path which the signalling messages should take. For the path-coupled signalling this would be the flow identifier only. Also, MRI includes a flag to distinguish between upstream and downstream data flows.
- *NSLPID (NSLD identification)* is a unique identifier associated with the NSLP, which is generating messages for this flow. This field is included to identify signalling application for which GIST preserves internal state and is used to pass messages upwards of the protocol stack.

- *SID (Session Identifier)* is an identifier for a session. GIST associates each message of signalling applications with a “signalling session”. Signalling applications provide the session identifier whenever they wish to send a message, and GIST reports the SID when a message is received. Because of several possible relationships between LSLDIP and SID, GIST does not perform any validation on flows and session mappings. Moreover, it performs no validations on property of SID itself.

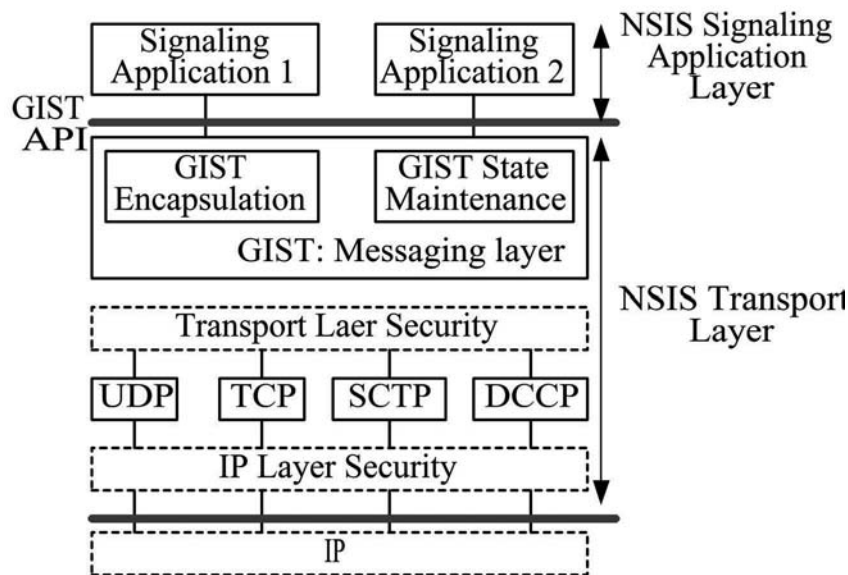


Figure 3-67: GIST's signalling transport protocol stack

The triplet mentioned above uniquely identifies a data flow within a signalling session. The information associated with a given {MRI, NSLPID, SID} triplet consists of the routing state to reach the peer in the direction given by the MRE. The routing state includes information about the peer identify, and a UDP port number (for datagram mode) or a reference to one or more messaging associations (for connection mode). This simplistic approach allows GIST to distinguish between data flows and to map them to specific signalling applications. The necessity in the GIST state maintenance has influence on the format of GIST messages, which have the following fields in the common header:

- Version defines the version number of GIST protocol
- Length describes the number of 32 bit words in the messages
- NSLPID is the signalling application identifier that needs to be included in message header to be mapped to appropriate signalling application on local nodes.
- GIST hop counter is used to prevent from endless package looping.
- Message content consists of Type-Length-Value which carries processing instruction.

To set up the necessary routing state between adjacent peers, GIST defines a three way handshake consisting of “Query”, “Response” and an optional “Confirm” message (see figure 3-68).

As mentioned above, GIST has two optional modes: datagram mode and connection mode. Datagram mode is a mode for sending GIST messages between nodes without using any transport layer reliability or security protection. This mode uses UDP encapsulation. The IP addressing is done either based on information from the flow definition or previously discovered adjacency data. Datagram mode is used for small, infrequent messages with no strict delay constraint. In contrast, the connection mode is the mode for sending GIST messages directly between nodes using point to point messaging association and is based on the TCP. This mode allows the re-use of existing security and transport protocols. In general, the connection mode is used for larger data objects where security or reliability is required. Additionally, the datagram/connection mode selection decision is made by GIST on the basis of the message characteristics and the transfer attributes stated by the applications. However it is possible to mix these two modes along the data flow path. For example, GIST can apply datagram mode at the edges of the network and connection mode in the network core.

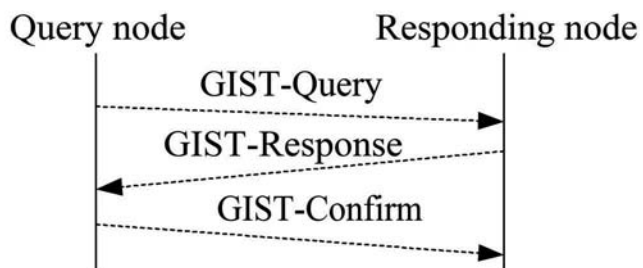


Figure 3-68: GIST three-way-handshake

In this section we have described the way GIST treats signalling messages at the NSIS transport layer. Specific signalling state setup is left to the signalling applications. They operate on NSIS signalling layer at which we will take a look in the next chapter.

3.9.2.4 NSIS Signaling Layer Protocols

NSIS signalling layer provides application specific signalling. Currently following NSIS signalling layer protocols have been defined in [MKM-2008, STA-2006]:

- *NSLP for NAT/Firewall*: this protocol allows hosts to signal along a data path for network address translators and firewalls to be configured according to the data flow needs.
- *NSLP for Quality of Service signalling*: This NSLP protocol provides signalling support for network resource reservation. It is independent of the underlying QoS specification or architecture. In the following sections, only the QoS NSLP will be considered.

3.9.2.4.1 QoS NSLP Overview

The QoS NSLP protocol establishes and maintains the flow state at nodes along the path of a data flow with the purpose of providing some forwarding resources for that flow. The QoS NSLP relies on GIST to carry out many aspects of signalling message delivery. There are three entities defined for QoS NSLP:

- *QoS NSIS Entity (QNE)*: is an NSIS entity that supports the QoS NSLP.
- *QoS NSIS Responder (QNR)*: is the last node in the sequence of QNEs that receives a reservation request.
- *QoS NSIS Initiator (QNI)*: is the first node in the sequence of QNEs that issues a reservation request for a session.

These entities within the QoS NSLP architecture are shown in figure 3-69.

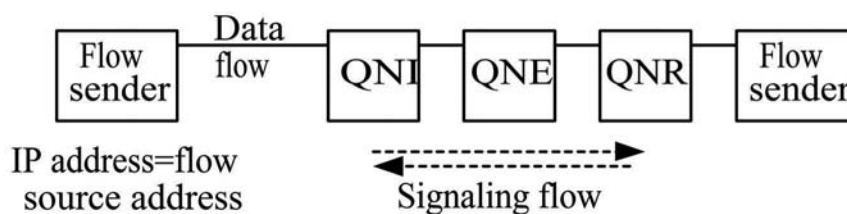


Figure 3-69: Components of QoS NSLP architecture

The logical architecture for the operation of the QoS NSLP and associated mechanisms within a node are shown in figure 3-70. This figure shows an example of a implementation scenario where QoS conditioning is performed on the output interface. For a single node, the request for QoS may result from a local application or from processing of a incoming QoS NSLP message. For a single QNR, the following schema applies:

- Incoming messages are captured during the input packet processing and handled by GIST. Only messages related to QoS are passed to QoS NSLP.
- The QoS request is then handled by a local resource management function.
- The grant processing involves two logical decision modules: policy control and admission control.

- If both checks succeed, parameters for packet classifier and for packet scheduler are set in order to obtain desired QoS.
- The final stage of the resource request processing is to notify the QoS NSLP protocol that the required resources have been configured.

The QoS NSLP may forward the resource request in one direction and may generate an acknowledgement message in the other. If reservation fails, an error notification is passed back to the request originator.

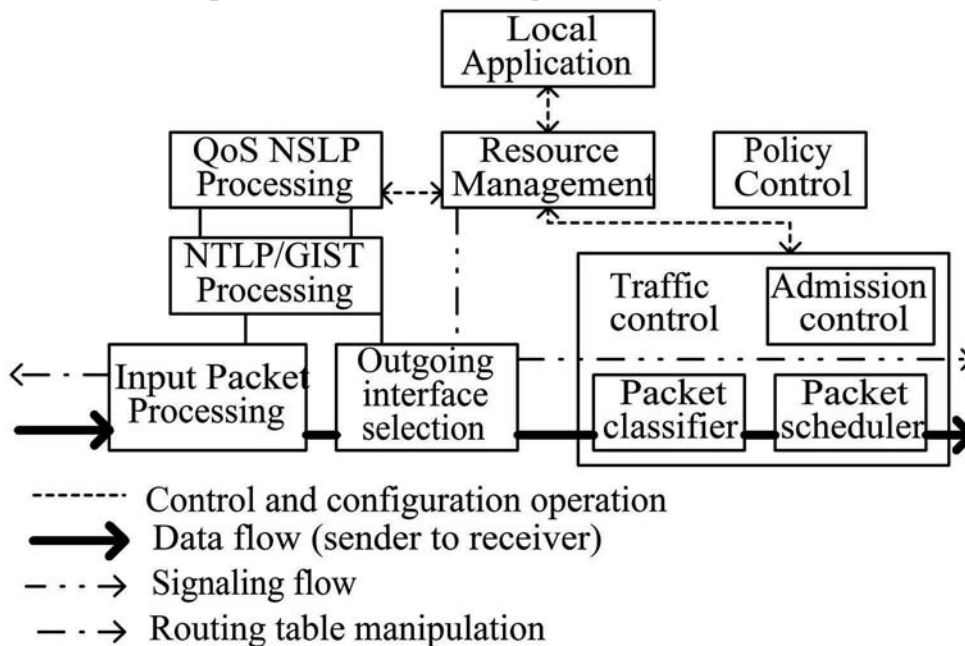


Figure 3-70: QoS NSLP in a Node [MKM-2008]

3.9.2.4.2 QoS NSLP Message

The QoS NSLP defines four following message types:

1. “Reserve”: is used to create, refresh, modify and remove QoS NSLP reservation state.
2. “Query”: is used to request information about data path without making a reservation
3. “Response”: is used to provide information about the result of a previous QoS message
4. “Notify”: is used to send notification.

As the NTLN messages, QoS NSLP messages are sent peer-to-peer, i.e. the source of each message is the adjacent downstream or upstream peer. Each protocol message has a common header which indicates the message type and contains various flag bits. Three types of objects contained in each QoS NSLP message are:

- *Control information objects* carry general information for the QoS NSLP processing, such as sequence numbers or information indicating whether a response is required.
- *QoS specification objects* (QSPECTs) describe the resources required in depending on the QoS model being used.
- *Policy objects* contain data used to authorize the reservation of resources.

3.9.2.4.3 QoS NSLP Design

Following design principles have been used as key functionality of QoS NSLP [MKM-2008]:

- *Soft States.* The reservation state in a QNE must be periodically refreshed by sending a “Reserve” message. The frequency with which state installation has to be refreshed is expressed in the “Refresh_Period” object.
- *Sender and receiver Initiation.* QoS NSLP supports both sender-initiated and receiver-initiated reservations. In the first case, “Reserve” messages travel in the same direction as data flow that is being signalling for. In the second case, “Reserve” messages travel in opposite direction; the sender of data first sends a “Query” message with the “Reserve-Init” flag set, then the receiver answers with a “Reserve” message.
- *Message sequencing.* The order in which “Reserve” messages are received influences the eventual reservation state in QNE – the most recent “Reserve” message places the current reservation. To protect against “Reserve” message re-ordering, QoS NSLP uses the “Reservation Sequence Number” (“RSN”) object.
- *Explicit confirmation and responses.* A QoS NSLP instance may request an explicit confirmation of its resource reservation actions from its peer. This is achieved by using an “Acknowledge” flag in the “Reserve” message header. QNE may also require a reply to a query along the path. To keep track of which request each response refers to, a “Request Identification information” (“RIT”) object is included in the QoS NSLP messages.
- *Reduced refreshes.* For scalability, QoS NSLP supports a reduced from refresh “Reserve” message, which references the reservation using the “RSN” and the “Session_id”, and does not include the full reservation specification (“QSPEC”).
- *Message scoping.* The QoS NSLP has an explicit mechanism to strict message propagation. A generic “Scoping” flag limits a part of the path on which state is installed or from which “Response” messages will be sent.

- *Session binding.* The concept of session binding is used in case of bidirectional and aggregate reservations. Session binding indicates a dependency relation between two or more sessions by including a “Bound_Session_Id” object. This information can be then used by a QNE for the logical resource optimization.
- *Aggregate reservation.* In some cases, it is desirable to create reservations for an aggregate, rather than on a per-flow basis, in order to reduce the amount of reservation states and the processing load for signaling messages. The QoS NSLP does not specify how reservation need to be combined in an aggregate or how end-to-end properties need to be computed but only provides signaling support for it.
- *Support for Request Priorities.* Since in some situations, some messages or some reservations may be more important than others and therefore it is necessary to give these messages or reservations priority.
- *Rerouting.* This function deals with ability to adapt to route change in the data path, e.g. detecting rerouting events, creating a QoS reservation on the new path and tearing down the reservation on the old path.

3.9.2.4.4 Examples of QoS NSLP Operations

There is a number of ways in which the QoS NSLP can be used. This paragraph illustrates some examples of the basis processing of QoS NSLP described in [MKM-2008].

- *Sender-initiated reservations.* A new reservation is initiated by the QNI, which constructs a “Reserve” message containing a QSPEC object that describes the required QoS parameters. This “Reserve” message is sent to GIST which delivers it to the next QNE. This QNE then treats the message as follows: the message is examined by the Quest NSLP processing; The policy control and admission control decisions are then made (see figure 3-70); The exact processing also takes into account the Quest model being used; Based on the QSPEC object in the message, appropriate actions are performed at the node (e.g. installing the reservation); The QoS NSLP then generates a new “Reserve” message that is passed to GIST, which forwards it to the next QNE. The same processing is performed at further QNEs along the path, up to the QNR that is the destination for the message (figure 3-71). The QNE then constructs a “Response” message which is forwarded peer-to-peer along the reverse of the path that the “Reserve” message took.

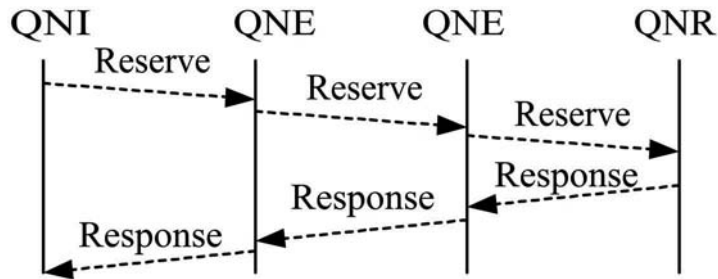


Figure 3-71: Sender initiated reservation

- *Sending a Query.* “Query” messages can be used to gather information along the path. These messages are constructed at QNEs and include a QSPEC containing the actual query to be performed at the QNEs along the path. By receiving a “Query” message, the QNE inspects it and creates a new message with the query objects modified as required. The new message is then passed to GIST for further processing. When “Query” message arrives at the QNR, it generates a “Response” message if the “Query” message includes a request identification information (RII) object. This “Response” message includes various objects from the received “Query” message. It is the passed to GIST to be forwarded peer-to-peer back along the path.
- *Receiver-initiated reservations.* To make a receiver-initiated reservation, the QNR (sender) constructs a “Query” message object with the “Reserve-Init” flag set and a QSPEC object included. While travelling to the QNI (receiver), the “Query” message records the available bandwidth on the path in the QSPEC object and causes GIST reservation path state to be installed (figure 3-72). The QNE detects that this “Query” message carries the “Reserve-Init” bit, and thus it constructs a “Reserve” message by using the information contained in the QSPEC. The “Reserve” message is then forwarded peer-to-peer using GIST reverse path state. The “Response” message is then sent back to confirm that the resources are set up.

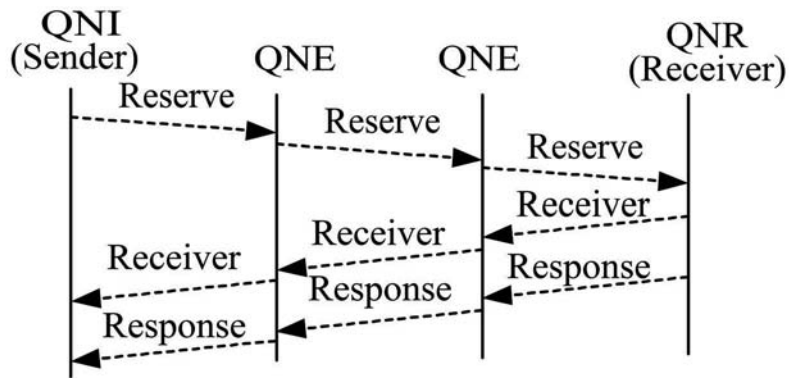


Figure 3-72: Receiver initiated reservation

- Aggregate Reservation.** In order to reduce the signalling and the per-flow state management in the network, the QoS NSLP should aggregate the reservation for a number of flows together. In this case, all end-to-end per-flow reservations are initiated as normal with “Reserve” message. A reservation for aggregated flows is initiated at the aggregator (figure 3-73) that has a flow identifier for the aggregated flow instead of for individual flow. The deaggregator stops this aggregation process by reassigning the QoS NSLPID value and becomes the next hop QNE for the end-to-end per flow reservation. The key feature of the aggregate reservation is that its flow identifier is different to that for the end-to-end reservation. This enables the aggregate reservation to be updated independently of the per-flow end-to-end reservations.

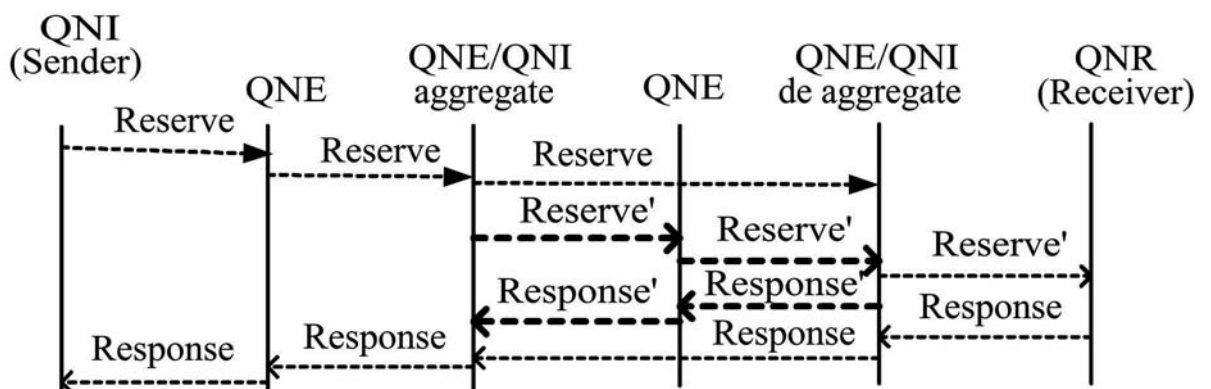


Figure 3-73: Sender initiated reservation with aggregation

3.9.3 Signaling for Voice over IP

Voice over IP (VoIP) deals with protocols, services and architectures for enabling voice conversations over the Internet or through any other IP-based network with a suitable QoS and superior cost. In order to perform voice over

IP, protocols for VoIP signalling and for end-to-end VoIP delivering are needed. A VoIP signalling protocol is used for initiating, modifying and for terminating VoIP sessions. For delivering VoIP traffic between end systems, transport protocols such as TCP, UDP, RTP and RTCP are used.

This section begins with a discussion about VoIP architectures. After that, VoIP signalling protocols (H.323 and SIP) will be described. Finally, a comparison between SIP and H.323 will be shown.

3.9.3.1 Standards and Protocols for Voice over IP

For supporting VoIP applications, several standards and protocols have been proposed that are illustrated within the TCP/IP protocol stack in figure 3-74 below.

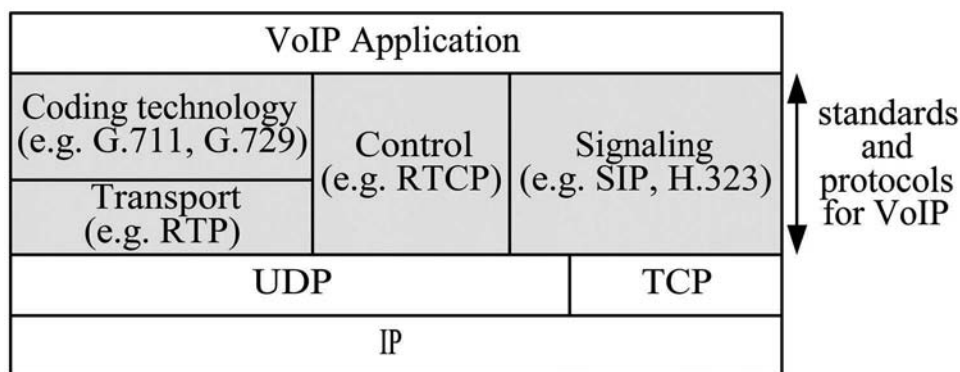


Figure 3-74: Voice over IP standards and protocols

These protocols and standards are described as follows:

- *Coding technologies.* These standards deal with most popular voice and video coding technologies that define how the analogue signals were transformed into data.
- *RTP (Real-Time Transmission Protocol).* RTP is used as the transport protocol for packetized VoIP. This protocol is a standard [RFC 3550] from IETF and usually associated with Real-time Control Protocol.
- *RTCP (Real-time Control Protocol).* RTCP is a control protocol used by multimedia applications along with RTP. RTCP enables the end systems to identify the participants and monitor the QoS parameters.
- *SIP and H323.* These VoIP signalling and controlling standards are used for establishing, manipulating, tearing down an interactive user session. While SIP is a standard from IETF, H.323 is a standard from ITU.

The RTP and RTCP will be described in section 3.11 and thus they are not subject of this section. In the next following sections, H.323 and SIP will be discussed.

3.9.3.2 H.323

The ITU-T H.323 standard specifies complete architectures and operations for multimedia communication systems in a packet-based network, such as IP, ATM or IPX/SPX. The standard includes a set of H.323 components and the protocols used between these components. The H.323 consists of a specification of the following components shown in figure 3-75:

- *H.323 terminals*. These components are endpoints that enable real-time voice or video communications with other H.323 terminals, gateways or MCUs on the network.
- *MCU/MC/MPs*. Multipoint Control Units (MCUs) include a Multipoint Controller (MC) and one or several Multipoint Processors (MPs). These components allow the management of multipoint conferences.
- *Gateways*. These devices allow intercommunication between IP networks and legacy Switched Circuit Networks, such as ISDN and PSTN. The gateways provide signalling, mapping and transcoding facilities.
- *Gatekeepers*. These devices perform the role of the central “managers” of VoIP services to the endpoints. Mandatory functionality includes address resolution, authentication, terminal registration, call admission control and more.

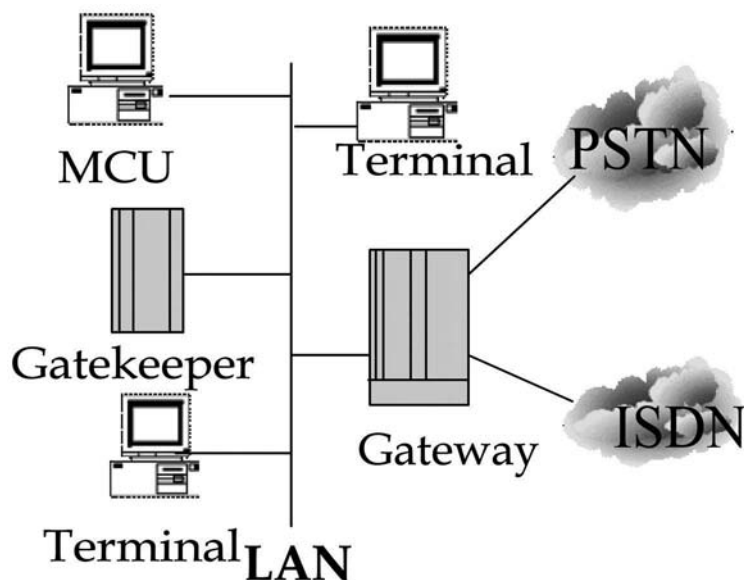


Figure 3-75: VoIP components

H.323 is an umbrella standard that includes the following protocols (see figure 3-76):

- H.225 Call signalling and RAS: are used between terminals (H.323 endpoints) and their gatekeeper and for some inter-gatekeeper communications. H.225 performs two functions. The first one is used between H.323 endpoints to signal call setup intention, success, failures, etc. as well as to carry operations for supplementary. The second one is the so called RAS (registration, admission and status) that performs registration, admission control, bandwidth changes, and disengage procedures between endpoints and their gatekeepers.
- H.245 Conference Control: is used to establish and control two party calls, allowing two endpoints to negotiate media processing capacities such as audio/video codecs for each media channel between them and to configure actual media streams. In context of H.323, H.245 is used to exchange terminal capability, determine master-slaver relationships of endpoints, and open and close logical channels between two endpoints.
- RTP and RTCP: are used for transfer the audio data.
- Q.931 is the signalling protocol for call setup and teardown between two H.323 terminals. It includes protocol discriminator defining which signalling protocol is used, the call reference value for addressing the connection, and the message types.
- Codecs. Most popular voice coding technologies are G.711, G.712, G.728 and G.729. For video coding, H.261 and H.263 are used.

H.323 is placed above the transport layer. In theory, H.323 is transport-independent but in practice RTP/RTCP runs over UDP or ATM and other protocols run over TCP and UDP.

Speech	Control			
Speech (De)coding (G.7xx)	RTCP	H.225	Q.931	H.245
RTP				
UDP	TCP			
IP				

Figure 3-76: H.323 protocol architecture

3.9.3.3 SIP

The Session Initiation Protocol (SIP) is an ASCII-based, end-to-end signalling protocol that can be used to establish, maintain, modify and terminate Internet telephone sessions between two or more endpoints. SIP can also invite

participants to already existing sessions, such as multicast conferences. In SIP, the signalling state is stored in end-devices only and not in the routers along the path to destination. Thus, there is no single point of failure and networks designed this way scale well. SIP is specified in the RFC 3621 [RSC-2002] by the IETF.

SIP is a part of the IETF multimedia architecture that includes the RTP for transporting audio and video data, the RTSP for setting up and controlling media streams, the media gateway control protocol (MGCP), H.248 for controlling media gateways, and the session description protocol (SDP) for describing multimedia sessions.

This section provides an overview of SIP. It first describes the basic architecture of SIP. It then discusses the SIP basic functions including location of an end point, signalling of a desire to communicate, negotiation of session parameters, and teardown of an established session. Finally, the SIP protocol structure will be presented.

3.9.3.3.1 SIP Architecture and Functionality

SIP basic architecture includes the specification of four logical types of entities participating in SIP - user agents, redirect servers, proxy servers, and registrars. These entities are described as follows [RSC-2002]:

- *User agents.* A user agent is a SIP endpoint that can act as both user agent client (UAC) and user agent server (UAS). The role of a user agent lasts only for duration of that transaction. A UAC is a client application that generates a SIP request and uses the client transaction to send it, as well processes a response. A UAS is a server application that is capable of receiving a request and generating a response based on user inputs, external stimulus, program execution result or on some other mechanisms. This response accepts, rejects, or redirects the request.
- *Redirect servers.* Redirect servers receive requests and then return the location of another SIP user agent or server where the user might be found. Redirection allows servers to push routing information for a request back in the response to the client
- *Proxy servers.* A proxy server is an application-layer router that forwards SIP requests to user agent servers and SIP responses to user agent clients. A request may traverse several proxy servers on its way to a UAS. Each proxy will make routing decisions, modifying the request before forwarding it to the next element. Response will route through the same set of proxy servers traversed by the requests in the reverse order.
- *Registrar servers.* These entities process the requests from UACs for registration of their current location within their assigned network domain.

From an architectural standpoint, the physical components of a SIP network can be grouped into two categories: clients (User agents) and servers (Redirect Server, Proxy Server, and Registrar Server). Figure 3-77 illustrates the architecture of a SIP network.

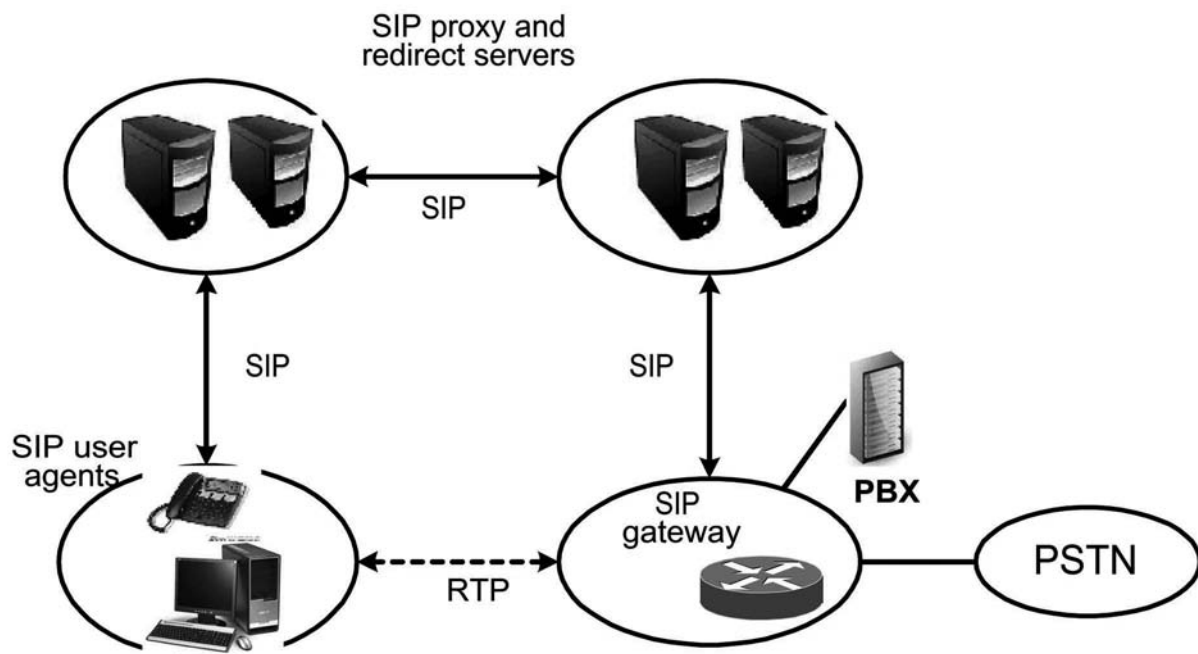


Figure 3-77: SIP Architecture

These four SIP entities described above together perform the following SIP functions:

- *Determining the location of the target endpoint:* SIP supports address resolution, name mapping and call redirection.
- *Determining the media capabilities of the target endpoint:* The lowest level of common services between endpoints can be determined by SIP through session description protocol (SDP). Thus, SIP establishes the conferences using only the media capabilities that can be supported by all endpoints.
- *Determining the availability of the target endpoint:* If a call cannot be completed because the target endpoint is unavailable, SIP determines whether the called party is connected to a call or did not answer in the allotted number of rings. SIP then returns a message indicating why the target endpoint was unavailable.
- *Establishing a session between originating and target endpoints:* if a call can be completed, SIP establishes a session between the endpoints.
- *Handling the transmission and termination of calls:* SIP supports the transmission of calls from one endpoint to another. During a call transfer,

SIP simply establishes a session between the transferee and a new endpoint and terminates the session between the transferee and the transferring party.

3.9.3.3.2 How SIP Works

SIP uses requests and responses to establish communication among the various components in the network and to setup conference between two or more endpoints. Users in a SIP network are identified by unique SIP addresses. A SIP address is similar to email addresses and is in the SIP format “userID@gateway.com”.

Users register with a registrar server using their assigned SIP addresses. The registrar server provides this information to the local server upon request. When a user initiates a call, a SIP request is sent to a SIP server (either proxy or a direct server). The request consists of the address of the caller (in the from header field) and the address of the intended called party (in the “To” header field). When a SIP end user moves between end systems, the location of the end user can be dynamically registered with the SIP server. SIP can work with a proxy server or with a redirect server depending on where the request is coming. If the request is coming through a SIP proxy server, the proxy server tries each of returned addresses until it locates the end user. If the request is coming from the SIP redirect server, the redirect server forwards all the addresses to the caller in the contact header field of the invitation response. The working principle with these servers is described in figures 3-78 and 3-79 as follows:

- *SIP session through a proxy server.* If a proxy server is used, the caller user agent (UAC) sends an INVITE request to the proxy server, which determines the path and then forwards the request to the called user agent (UAS). The called user agent responds “Response 200 OK” the proxy server that then forwards the response to the caller user agent. The proxy server then forwards the acknowledgements of the caller and called user agent. A session is established between these parties. At this point, RTP is used for data transfer between the caller and called party. The process of session establishment via a proxy server is illustrated in figure 3-78.
- *SIP session through a redirect server.* If a redirect server is used, the caller user agent sends an INVITE request to the redirect server, which then contacts the location server to define the path to the called user agent and sends that information “302 moved temporarily” back to the caller. The caller user agent then sends a INVITE request to the called party. Once the request reaches the called party, it sends back a response and caller acknowledges the response. From this time, RTP is used for delivering the

data between the caller and called user agent. The process of session establishment via a redirect server is illustrated in figure 3-79.

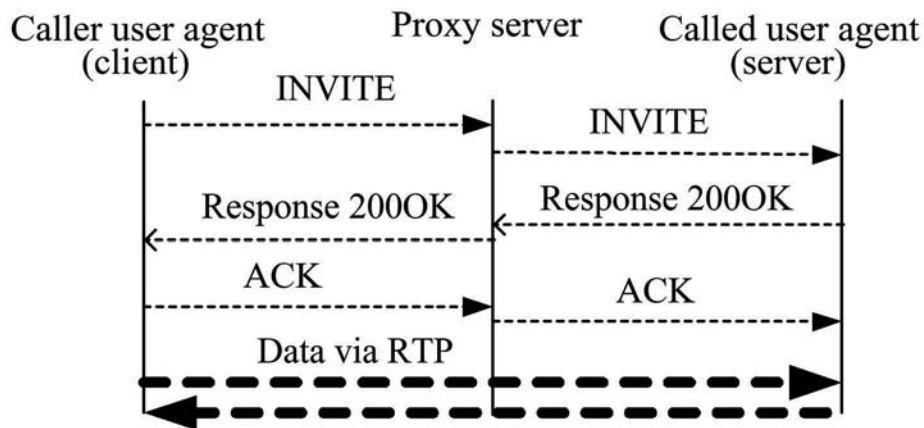


Figure 3-78: Sequence diagram of a SIP session through a proxy server

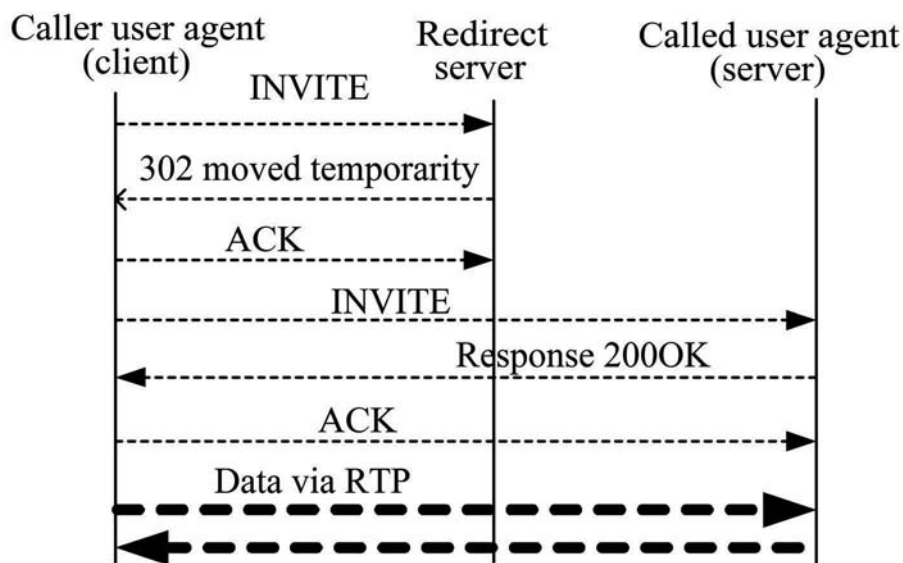


Figure 3-79: Sequence diagram of a SIP session through a redirect server

3.10 QoS Architectures

So far we have discussed the details of how individual routers and hosts can differentiate traffic, provide predictable bandwidth sharing, and support congestion control and queue management to keep senders from overrunning networks. But the information about these mechanisms do not mean too much with a architecture of how these mechanisms ought to be used together to provide end-to-end QoS. This section summarizes three architectures proposed

by IETF to provide QoS over IP networks – Integrated Services (IntServ) [RFC1633], Differentiated Services (DiffServ) [RFC2475], and Multi Protocol Label Switching (MPLS) [RFC3031].

IntServ was the first architecture that supports per-flow quality of service guarantees, requires relatively complex packet classify, admission control, signaling, queuing and scheduling within any router belonging to the end-to-end data transmission path. DiffServ can be viewed as a improvement to IntServ. In contrast to IntServ, DiffServ handles packets on the per-class basis that allows the aggregation of several flows to one class, and does not need the per-router signalling as in IntServ. In comparison with IntServ and DiffServ, MPLS additionally supports explicitly constructed non-shortest path routing of traffic. Based on the label-switching concept, MPLS can be used in the high speed backbone.

A number of following concepts are common to each approach:

- A router is characterized as edge or core router,
- Edge routers accept customer traffic into the network,
- Core routers provide packet forwarding services between other core routers and/or edge routers,
- Edge routers characterize, police, and/or remark customer traffic being admitted to the network. Edge routers may use admission control to accept or reject a flow connection.

3.10.1 Integrated Services

The goal of IntServ [RFC1633] was to augment the existing best effort Internet with a range of end-to-end services developed for real-time streaming and interactive applications. Its aim was to provide per-flow QoS guarantees to individual application sessions. IntServ defines several new classes of services along with the existing best-effort service. To receive performance assurance from the network, an application must set up the resource along its path before it can start to transmit packet.

3.10.1.1 IntServ Basic Architecture

The basic principle of the IntServ can be described in the following. The sender starts the setup of a reservation by describing the characteristics of the flow and the resource requirements to the network. Hosts and routers within an IntServ network use the RSVP to setup and maintain the resource reservation for each flow. The network accepts this new flow only if there is sufficient resources to meet the requested resources. Once the resource reservation setup is successful, the information for each reserved flow is stored in the resource reservation table,

which is used to configure the packet classification and packet scheduling in the data plane; when data packets arrive, the packet classifier module select packets that belong to the reserved flows and puts them on the appropriate queues; the packet scheduler then allocates the resources to the flows based on the reservation information.

The logical architecture of an IntServ host and an IntServ router is illustrated in figure 3-80 and 3-81. The architecture is divided into two parts: control plane and forwarding plane. The components in this architecture can be summarized as follows:

- *Resource reservation setup (RSVP).* A reservation setup protocol is used to deliver QoS requests originating in an end-system to each router along the data path, and, to install and manage the reservation states in the routers. For an IntServ network, the RSVP was designed to be the reservation setup protocol.
- *Admission control.* In order to guarantee resources for reserved flows, each router uses the admission control to monitor its resource usage. It should deny reservation requests when no sufficient resources are available. The admission control component performs this task as a part of the reservation process; before a reservation request can be accepted, it has to pass the admission control test. At each node along the path the RSVP process passes a QoS request (flowspecs) to the admission control component to allocate the resources on node and link to satisfy the requested QoS.
- *Policy control.* Before a reservation can be established, the RSVP process must also consult policy control to ensure that the reservation is administratively permissible.
- *Packet scheduler.* If admission control and policy control are both succeed, the RSVP process installs state (flow specs) in the local packet scheduler. This state information is then used by the packet scheduler for allocating the bandwidth needed for each flow so that the requested QoS will be met. The packet scheduler multiplexes packets from different reserved flows into the outgoing links, together with best-effort packets.
- *Packet classifier.* The RSVP process also installs state (Filter specs) in the packet classifier component, which sorts the data packets forming the appropriate scheduling classes. The state required for selecting packets for a QoS reservation, is specified by the filter specs.
- *Routing.* Each router must determine which part should be used to setup the resource reservation. The path must be selected so that it is likely to have sufficient resources to meet the traffic demand and QoS requirement. It is important that the selected path meets such bandwidth, but optimal

route selection is difficult with the existing IP routing. The conventional existing routing protocols typically use a simple metric such as delay, hop count, or link weight to compute the shortest paths to all destination networks. These routing protocols do not have the necessary information about the available resources to make intelligent decision. In order to determine paths that meet the QoS requirements, QoS routing discussed in section 3.7.3 should be used.

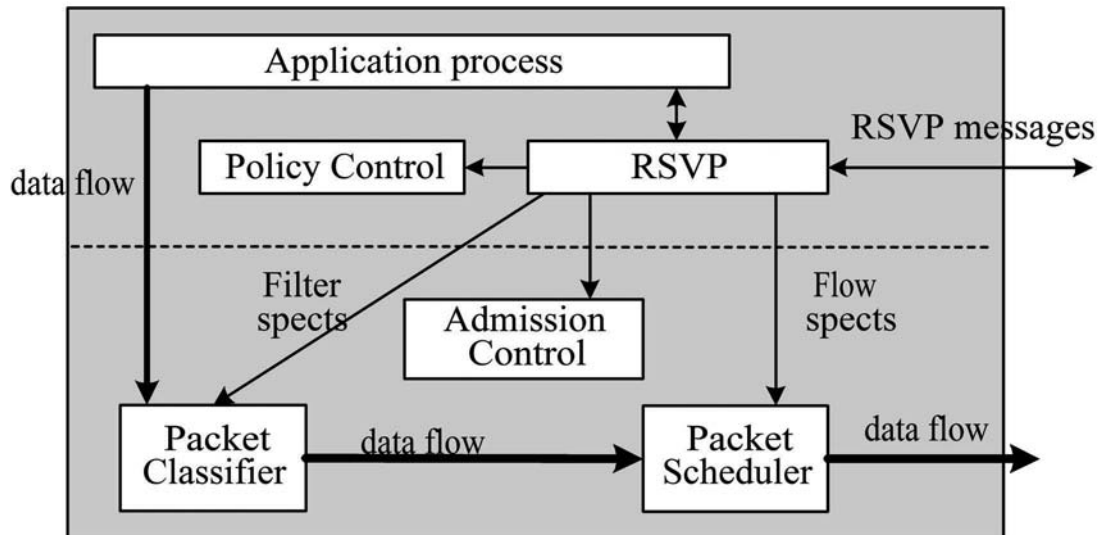


Figure 3-80: Logical architecture of an IntServ host

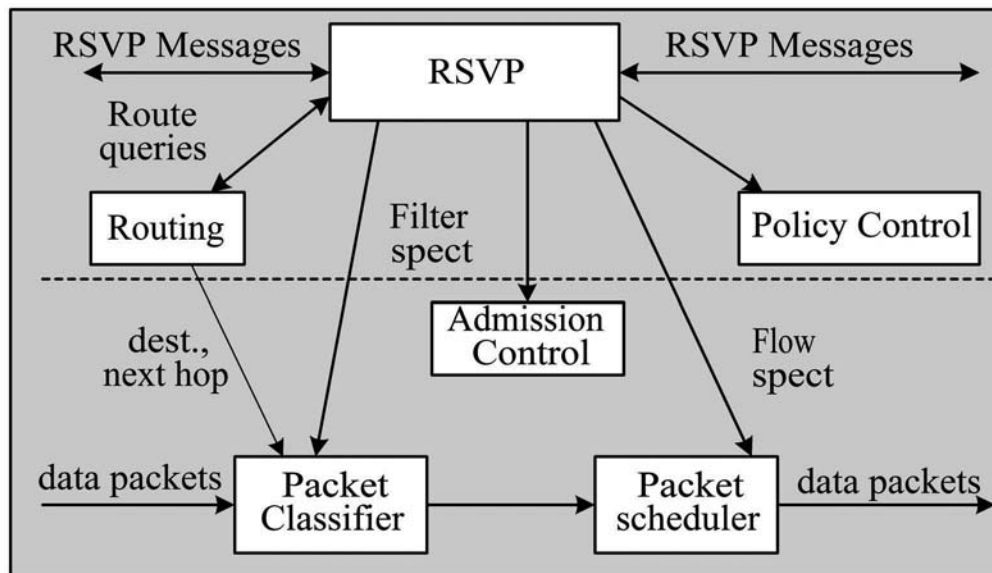


Figure 3-81: Logical architecture of an IntServ router

3.10.1.2 IntServ Service Classes

In IntServ, applications are classified into three categories: elastic applications, tolerant real-time applications, intolerant real-time applications.

- *Elastic applications* such as FTP or Telnet, are flexible in terms of their QoS requirements and can tolerate data rate, delay and packet loss. The best-effort service is acceptable for these applications as long as some resources are available for these services.
- *Tolerant real-time applications* such as audio conference or video streaming are very sensitive to delay. These applications require a sufficient amount of bandwidth and can tolerate occasional delays and losses.
- *Intolerant real-time applications* such as Internet telephony require more stringent QoS from the network. These applications require precise bandwidth, delay and jitter constraints, and they degrade severely if timing constraints are not met.

To meet the requirements for the application categories described above, IntServ implemented two additional services on top of the existing best-effort services: the controlled load service and the guaranteed service. These services will be summarized in the following:

- *Guaranteed Service*. This service provides firm bounds on bandwidth and deterministic upper bounds on end-to-end delay for conforming flows. Guaranteed service is intended for intolerant real-time applications. The guaranteed service specification is described via a token bucket (with a token rate r and a token size b) and by a transmission rate R . This service can be implemented using resource reservation protocols such as RSVP. In order to provide guaranteed service, the resources have to be reserved for the worse case. For bursty traffic sources this will lead to low network utilization and to increased cost for resource reservation. Moreover, it is often difficult to know exactly the bandwidth and delay requirements for a given application flow.
- *Controlled Load Service*. For some applications, a service model with less strict guarantees than guaranteed service and lower cost would better serve their needs. The controlled load service is developed for this purpose. This service is meant to be better than the best-effort service and is designed for tolerant real-time applications. These applications are performed well with the controlled load service when the network is only lightly loaded, but these applications degrade their performance rapidly as the network load increases. In comparison with the guaranteed service, the controlled load service model allows statistical multiplexing and so can be implemented

in more efficient way than guaranteed service, such as it can be implemented as a combination of policing, weighted random early drop and priority scheduling or weighted fair queuing.

3.10.1.3 IntServ Problems

The reason why IntServ has not been accepted in the Internet is the scalability, because the aim of IntServ was to provide per-flow QoS guarantee. Typically more than 200.000 flows may pass through an Internet core router. Maintaining the state for each of this large number of flows requires enormous resources. An empirical study on the impact of a large number of flows in an IntServ router is performed in [NCS-1999]. This study observed that real-time scheduling overhead increases with the number of real-time flows. And if the number of IntServ flows reaches around 400, the scheduling overhead increases sharply so that the router is unable to cope with the load. As a result, the packet needs to wait in the queue. This study also showed that the elastic traffic suffers a 1% packet loss when the number of real-time flows exceeds 450.

Another IntServ's problem is that the RSVP required for flow setup has limitations such as lack of negotiation and back tracking, requirement of frequent updates, and path pinning to maintain soft state in routers.

3.10.2 Differentiated Services

In order to solve IntServ problems discussed in the last section, the IETF proposed Differentiated Services architecture (DiffServ) that could support a scalable form of QoS and could provide a variety of end-to-end services across multiple, separately administered domains.

The DiffServ framework is specified in the RFCs 2474 and 2475 [NBB-1998, BBC-1998]. The basic idea of DiffServ is to classify and mark each IP packet's header with one of the standard DiffServ code point (DSCP), separating the packet processing functions between edge routers and core routers, moving per-flow data path functions to edge routers, treating and forwarding the incoming packets at the core routers based on the DSCP. Packets with the same code point receive an identical forwarding treatment by the routers and switches in the path toward the receiver. This prevents the state maintaining in the routers or complex forwarding decisions in the core routers based on per flow, as with IntServ.

The rest of this section discusses various aspects of the DiffServ architecture and services.

3.10.2.1 DiffServ Architecture

A DiffServ network domain, which consists of a set of interior core routers and boundary edge routers, is illustrated in figure 3-82. In order to prevent per-flow basic, DiffServ separates the functionalities between edge routers and core routers within a DiffServ domain. The basic principle is to move per-flow data path functions to the edge routers that classify and mark the arriving packets with the so called DiffServ code point (DSCP) – a bit combination in the ToS field in IPv4 header and in the traffic class field in the IPv6 header. The core routers then treat and forward the incoming packets based on the DSCP.

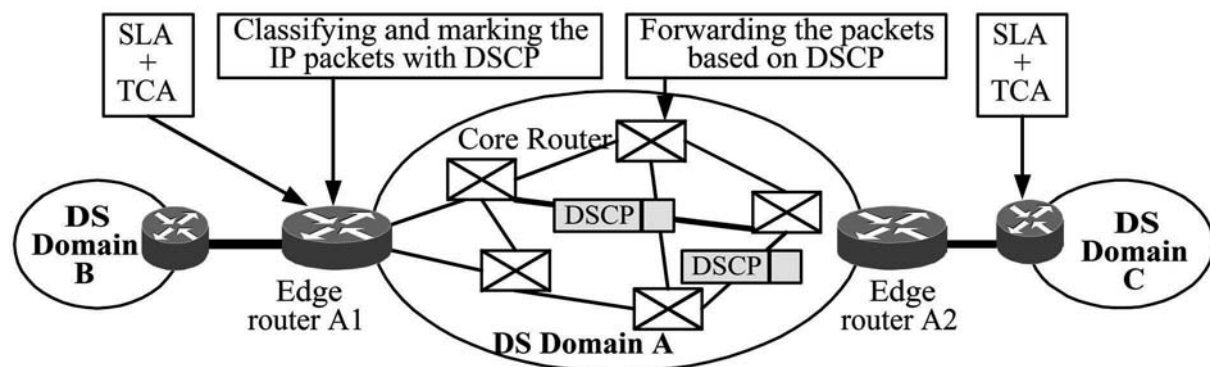


Figure 3-82: DiffServ Domain

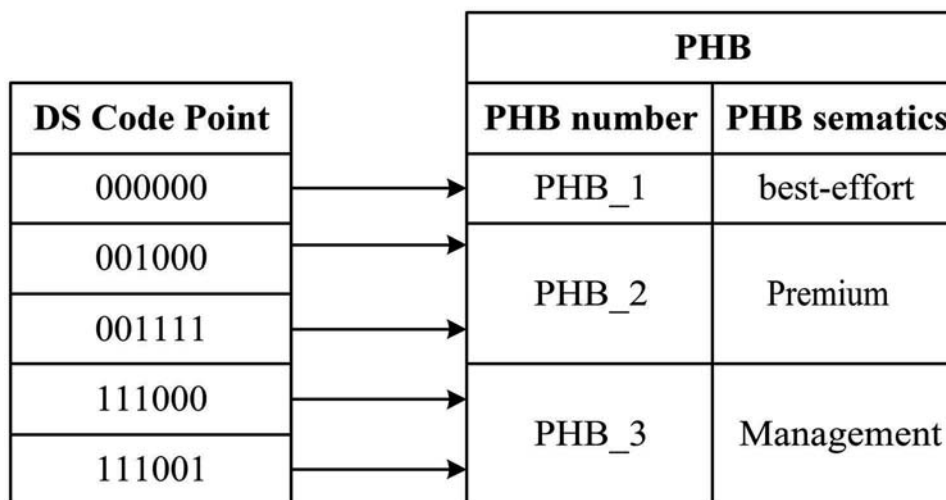


Figure 3-83: Example of mapping between DSCP and PHB semantics

To enable the service differentiation, DiffServ defines the per hop behaviour (PHB) a packet may receive at each hop. A PHB is a forwarding treatment specifying the queuing, scheduling and congestion-related actions at each node. When a PHB for a given DSCP is unknown, the concerned packet is assigned as the default PHB. Figure 3-83 illustrates an example of mapping from DSCP to PHB semantics. This example shows that a DSCP of an incoming packet is used

by a router to identify which service the router should use to treat and forward this packet. For example, packets with DSCP value equal to “000000” should be treated with the best-effort service and packets with DSCP value equal to “001000” should be treated with services used for premium traffic [BBC-2001].

3.10.2.2 DiffServ Routers and the Protocol Mechanisms

As mentioned above that the DiffServ concept separates the packet processing functions between edge routers and core routers. While edge routers perform multiple header fields (MF) classification, marking, traffic conditioning and forwarding, core routers achieve the packet classification based on the DSCP and packet forwarding based on the PHB. The protocols mechanisms implemented in a DiffServ edge router and core router are shown in figure 3-84 and 3-85.

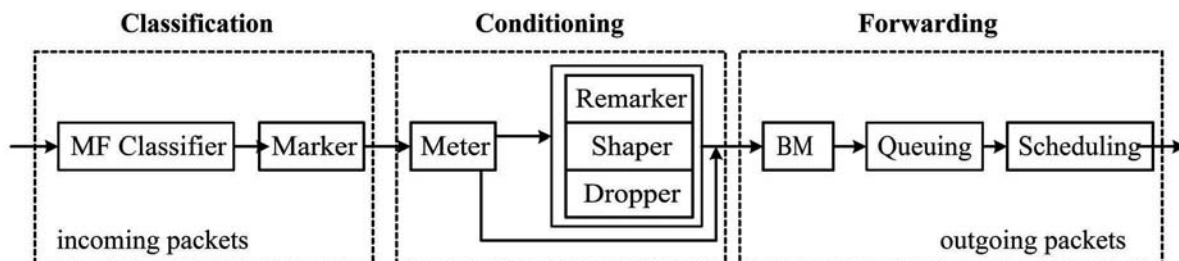


Figure 3-84: Packet forwarding mechanisms at a DiffServ edge router

These mechanisms are summarized as follows

- **Classifier.** A classifier divides incoming packets into multiple groups. In DiffServ, the MF classification is used by the edge router, and at the core router the BA classification mechanism is applied.
- **Marker.** A marker sets the DSCP of a packet to a particular DSCP, mapping DSCP to a specific forwarding treatment (PHB), and letting this packet into the network.
- **Meter.** A meter measures the traffic flow from a customer into in-of-profile and out-of-profile packets. While in-of-profile packets are allowed to enter the network, the out-of-profile packets are further conditioned based on the traffic condition agreement between customers and the service provider. Mechanisms for traffic metering are described in 3.3.
- **Shaper.** A shaper delays the non-conformant packets in order to bring the stream into compliance with the agreed-on traffic profile. The traffic shaping mechanisms are discussed in section 3.3.
- **Dropper.** Dropping is the action that may be applied to out-of-profile packets in case of congestion.

Detail description of these mechanisms is found in section 3.3.

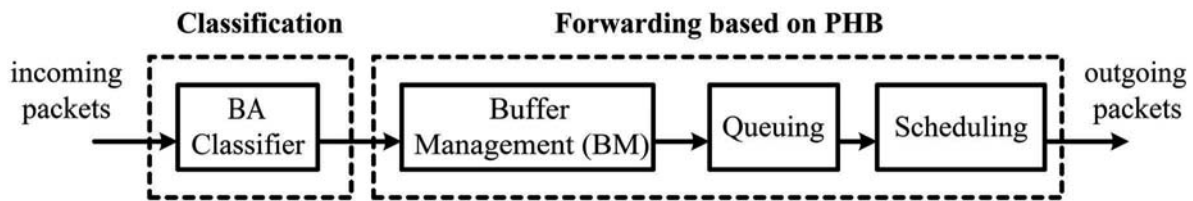


Figure 3-85: Packet forwarding mechanisms at a DiffServ core router

3.10.2.3 DiffServ Service Groups

DiffServ implements three service groups: the expedited forwarding PHB (EF PHB), the assured forwarding PHB (AF PHB) and the default PHB (best-effort PHB).

- Expedited forwarding PHB.** The objective of this PHB is to provide tools to develop a service that provides a low loss, low latency, low jitter, and bandwidth assurance through its DiffServ domain. Such a service is known as premium service, which is intended for traffic that requires a virtual leased line. The EF PHB is defined as a forwarding treatment for a particular aggregate where the departure rate of the aggregated flows from any DiffServ router must equal or exceed a configurable rate [NBB-1998]. The EF PHB can be implemented via a combination of shaping and priority scheduling with highest priority for the EF traffic queue.
- Assured forwarding PHB.** The AF PHB group defined in RFC 2597 is used to provide DiffServ domains to offer different levels of forwarding assurances for IP packets received from a customer DiffServ domain [RFC 2597]. The basic principle of AF PHB is that the DiffServ domain separates traffic into one or more of these AF classes according to the services that a customer has requested. Packets within each class are further divided into drop precedence levels. DiffServ provides four AF PHB classes. Each of these classes contains three drop precedence levels: low drop, medium drop and high drop precedence. This drop precedence is encoded in the last three bits of the DiffServ code point field, and the first three bits encode the class (figure 3-86). Also, DiffServ provides delay differentiation between four classes and drop probability differentiation within a class. Each AF PHB class in each DiffServ node gets a certain amount shared resources and a scheduler can be configured to assign bandwidth for some queues. Packets are assigned to a queue based on the AF PHB class. AF PHB can be implemented via a

combination of traffic metering, weighted random early drop (WRED) and weighted fair queuing (WFQ) scheduling.

- *Best-effort PHB.* The best-effort PHB group is used to develop best-effort service like by the traditional IP network. This best-effort PHB can be implemented via a combination of the RED and the FIFO scheduling.

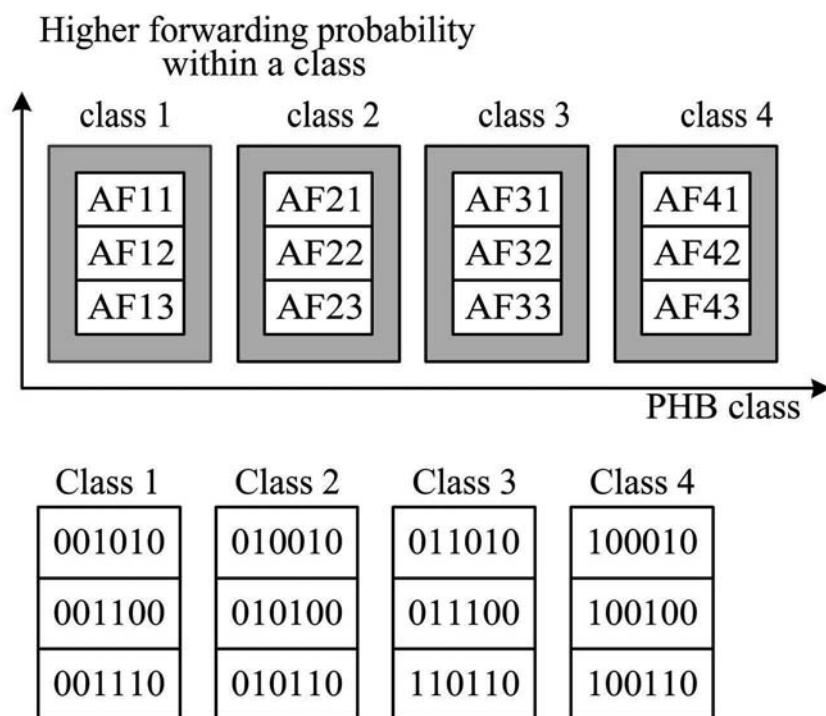


Figure 3-86: Implementation of PHB groups

3.10.3 Multi Protocol Label Switching

Internet Service Providers (ISP) constantly face the challenges of adapting their networks to support the rapid growth and customer demand for more reliable and differentiated services. In the early 1990s, the ATM allowed many service providers to offer the QoS guarantees that were not possible over the best-effort IP network. However, the ATM technology had several disadvantages, including the scalability and a high management cost for multiplexing IP traffic as one of many services carried over an ATM core. Researchers and developers worked on alternate ways to engineer the IP networks for providing the QoS, integrating IP and ATM, as well as VPN provisioning. The result of these efforts is the Multiprotocol Label Switching (MPLS) [RFC3031, RFC2702, RFC3032].

MPLS promises to be the most effective way to provide a stable packet network and integrate ATM and IP in the same backbone network. This enables the ISPs to preserve investment they made by ATM. MPLS is not an IP network,

although it utilizes IP routing protocols such as OSPF. Similarly, MPLS is not an ATM network, although it is a convergence of connection-oriented ATM forwarding techniques. Thus, MPLS has advantages of the Internet routing protocols and ATM traffic engineering so that MPLS resolves the problem of IP over ATM. Figure 3-87 depicts the conversions of the ATM and IP technologies in MPLS. MPLS reduces the processing overhead in routers, improving the packet forwarding performance. Furthermore, MPLS provides a new way to provide QoS that is complementary and in competition with DiffServ, IntServ with RSVP, and ATM.

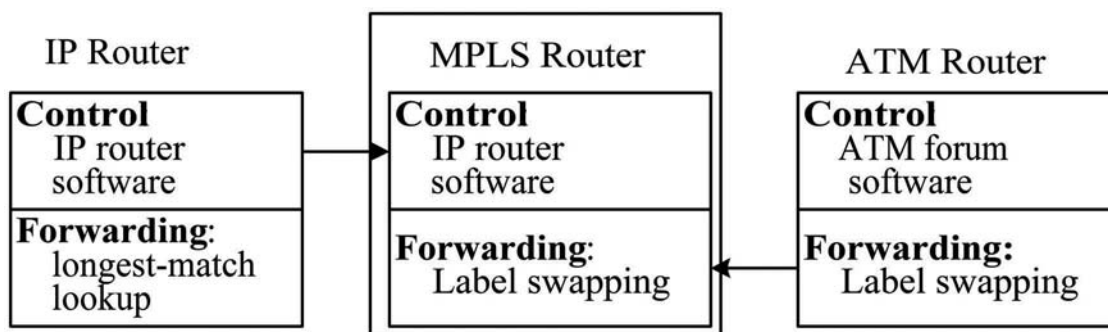


Figure 3-87: IP, MPLS and ATM

The rest of this section first describes the MPLS architectural concept. After that, the label distribution process will be discussed. Also, the MPLS routers and the protocol mechanisms will be explained. Finally, the traffic engineering and service implementation within MPLS will be summarized.

3.10.3.1 MPLS Architectural Concept

MPLS uses the so called label switching technique to forward data through the network. When an IP packet arrives into a MPLS ingress label switched router (LSR), a small fixed-format MPLS header (figure 3-88) is inserted in front of this IP packet. At each hop across the network, the packet is routed based on the value of the incoming interface and label and dispatched to an outgoing interface with a new label value. When the packet arrives at the egress LSR, the router removes the MPLS header and forwards the original IP packet into the IP network.

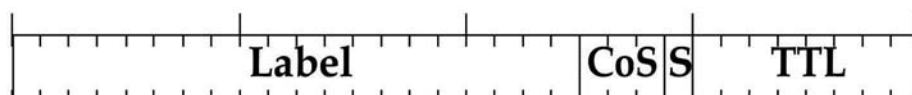


Figure 3-88: MPLS header

The MPLS header format is shown in figure 3-88. This header includes the following fields:

- *Label (20 bits)*. A label is a short fixed-length integer number used to identify a forwarding equivalent class (FEC) for this packet. A FEC is a group of IP packets, which are forwarded in the same manner (e.g. over the same path, with some forwarding treatment).
- *Exp (3 bits)*. This field is reserved for experimental use, such as for setting the drop priorities for packets in a way similar to that in the DiffServ.
- *Stack bit S (1 bit)*. The S bit is used to indicate the bottom of the label stack. The bit is set equal to 1 for the last entry in the label stack and to 0 for all other entries.
- *Time to live TTL (8 bits)*. The 8-bit field is used to encode a time-to-live value for detecting loops in LSPs.

The process of packet forwarding based on the label is illustrated in figure 3-89. The MPLS routers perform following tasks:

- *LSRs setup LSP for packets before sending them.*
- *Ingress LSRs* completely perform packet classification by using IP header fields, assigning MPLS header for each IP packet and forwarding the packet to the next core LSR.
- *Core LSR* examines the label of incoming packets for making the forwarding decisions, and performs label swapping.
- *Egress LSR* removes the MPLS header from the packet and forwards each packet on the basic of IP services assigned for this packet.

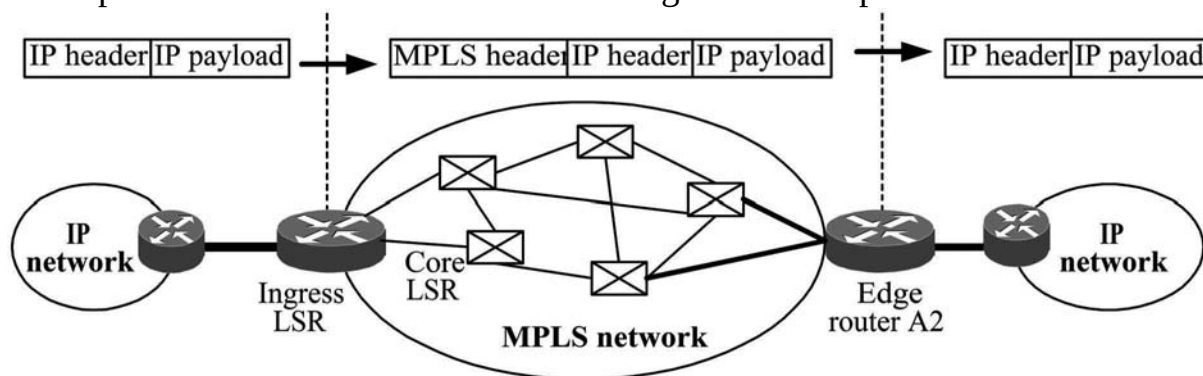


Figure 3-89: MPLS architectural component

The path that data flows through network is called label-switched path (LSP). At the ingress to an MPLS network, routers examine each IP packet to determine which LSP it should take, and, hence, which label to assign to it. This local decision is likely to be based on factors such as destination address, QoS requirements, and current network load. This dynamic flexibility is one of the key elements that makes MPLS so useful. The set of all packets that are

- *Notification messages*: Distributing advisory information and error information.

The label distribution generally consists of three mechanisms: label binding, label allocation and label switching.

- *Label binding*. Label binding deals with the algorithms for binding a label to an IP prefix address.
- *Label allocation*. Once LDP bindings are done, each LSR performs updating and modifying the label forwarding information base. In particular, the local label allocation at a LSR is a operation in which the local LSR sets up a label relation ship with the FEC.
- *Label switching*. The label switching determines how packets are forwarded within a LSR domain by using of label swapping. This process is done as follows. When a labelled packet arrives at a LSR, the forwarding component uses the input port number and the label to perform an exact match search in its forwarding table. When a match is found, the forwarding component retrieves the outgoing label, the outgoing interface, and the next hop address from the forwarding table. The forwarding component then swaps the incoming label with the outgoing label and directs the packets to the outbound interface for transmission to the next hop in the LSP.

The MPLS architecture does not mandate a single protocol for distributing the labels between LSRs. In addition to LDP, MPLS also allows the use of another label distribution protocols in different scenarios. Examples are

- *Constraint-based routing LDP (CR-LDP)*. CR-LDP is a label distribution protocol specifically designed to support traffic engineering. This protocol is based on the LDP specification with additionally extensions for supporting explicit routes and resource reservations. These extension features in CR-LDP include:
 - *Setup of explicit routes*. An explicit route is defined in a label request message as a list of nodes along the explicit route. CR-LDP supports both strict and loose modes of explicit routes. In the strict mode, each hop of explicit route is uniquely identified by an IP address. In the loose mode, the explicit route may contain some of the so called abstract nodes. Whereby, an abstract node represents a set of nodes. Abstract nodes may be defined via the IPv4 prefix, IPv6 prefix, autonomous system number, or LSP ID.
 - *Resource reservation and class*. Sources can be reserved for explicit routes. The characteristics of a explicit route can be described in terms of peak rate, committed data rate, peak burst size, committed burst size, weight and service granularity.

- *Path preemption and priorities.* If an LSP requires a certain resource reservation and sufficient resources are not available, the LSP may preempt existing LSPs based on the setup priority and holding priority parameters that are associated with each LSP. A new LSP can preempt an existing LSP if the setup priority of the new LSP is higher than the holding priority of the existing LSP
- *Resource Reservation Protocol and traffic engineering (RSVP-TE).* The RSVP-TE [RFC3209] is an extension of the original RSVP in order to perform label distribution and to support explicit routing. The new features added to the original RSVP include label distribution, explicit routing, bandwidth reservation for LSPs, rerouting of LSPs after failures, tracking of the actual route of an LSP, and pre-emption options.

3.10.3.3 MPLS Routers and Protocol Mechanisms

Simply replacing the longest prefix match forwarding with the label-switched forwarding is a major win. Moreover, an LSP needs not follow the shortest path between any edge LSRs. Since conventional IP routing protocols typically do not generate non-shortest path routes, external routing algorithms can be used to determine new routes for LSPs that will result in more optimal distribution of traffic load around the network. This traffic engineering feature is a major advantage of MPLS over IntServ or DiffServ alone.

In relation to QoS perspective, MPLS labels simplify the classification and determining the packet forwarding behaviour at the core and at the edge LSRs. The LSRs may implement the metering, policing, shaping, queuing, scheduling and buffer management techniques described in 3.4-3.6 for regular IP routers. But, instead of classifying packets based on their IP headers and payload fields, the MPLS labels itself provide all necessary contexts to determine the subsequence associated processing in routers and the next hop for this packet. This process is shown in figure 3-91 and 3-92. The LFIB contains the next hop information for labels the LSR knows about. Just like generic IP routers, an LSR includes control plane and forwarding plane. Although, forwarding engine of a core LSR includes packet classification based on the MPLS labels, label wrapping, switching, buffer management, queuing and scheduling. In comparison with the core router, the packet classification at a ingress LSR is based on the multiple fields in the IP header and IP payload of the packet (figure 3-92). Once the packet classification is done, the ingress LSR performs the label mapping, switching, buffer management, queuing and scheduling.

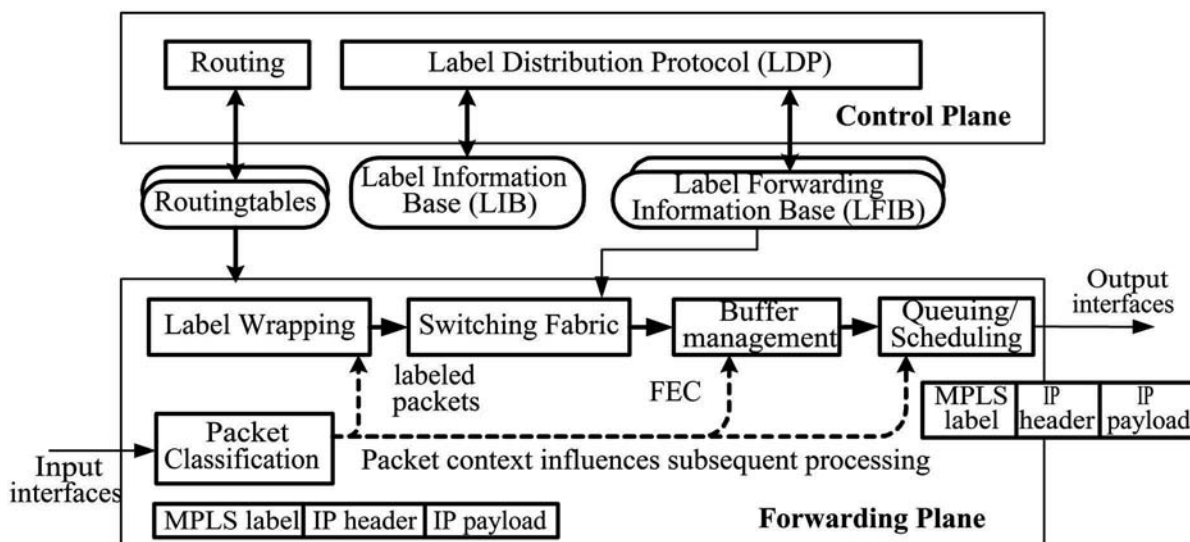


Figure 3-91: Packet processing within a core LSR

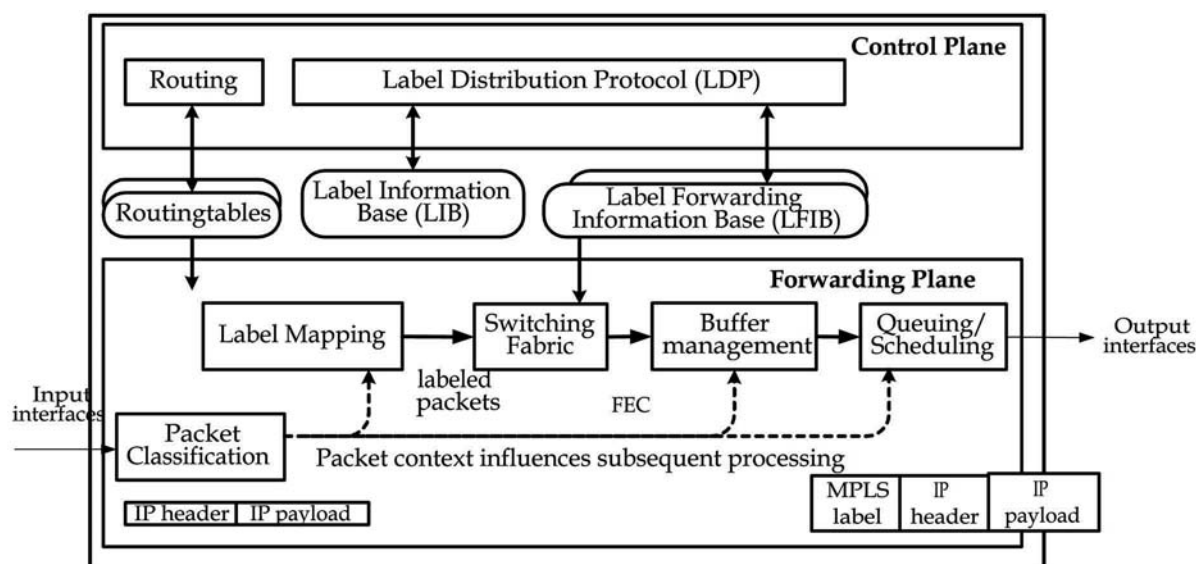


Figure 3-92: Packet processing within an edge LSR

3.11 Mobility Support

The increasing popularity of mobile devices, which demand for accessing to the Internet to get data and services at any time and any where, require the internet infrastructure to provide these mobile devices with a capability of connecting to the Internet while roaming, preferably without interruption and degradation of communication quality. While it has been possible to use a mobile device within one wireless domain for a long time, problems arise when the user wants to change location to another network, i.e. roaming. Since IP addresses have traditionally been fixed and bound to a physical location, every time a host changes its location, the address must change and TCP connections

are broken. To solve this problem, mobile IP introduces the use of two IP addresses: a fixed home address for other nodes (the correspondent nodes) to use, and a dynamic care-of-address that indicates the current location of the mobile node. Also, mobile IP defines architectures and mechanisms to allow mobile nodes to continue communicating with its corresponding nodes during its movement and to maintain the communication session during the mobility.

The section begins with the discussion about the mobile IPv4 – the standard solution for support mobility in IPv4 networks. Following this, solution for mobility support in IPv6 networks will be illustrated.

3.11.1 Mobile IPv4

Mobile IPv4 is a standard proposed by the IETF working group “Mobility for IPv4 (mip4)”. This standard is specified in RFC 3220 [Per-2002]. Together with this specification, several proposals [AL-2005, FJP-2007, KPL-2006, PCJ-2007, Per-2006, Mal-2007] for adding new functionalities into mobile IPv4 are defined by IETF.

The following sections begin with an illustration of the architectural overview of mobile IPv4. Based on this, the message format and the mobile IPv4 protocol mechanisms, such as agent discovery, registration, tunneling, will be discussed.

3.11.1.1 Architectural Overview

The mobile IP network architecture is shown in figure 3-93. Main components of this architecture are as:

- *Mobile Node (MN)*. MN is a host or router that changes its point of attachment from one network or sub-network to another. The MN’s functionality is to maintain network connectivity using its home IP address, regardless of which subnet it is connected to.
- *Home Agent (HA)*. HA is a router on a home network of mobile node. Its functionality is to maintain an association between the MN’s home IP address and its care-of-address
- *Foreign Agent (FA)*. FA is a router on a mobile node’s visited network (foreign network). FA provides an addressable point of attachment to the mobile node called care-of-address. Main functionality of FA is to maintain awareness for all visiting MNs, and to act as a relay between MN and its home agent. The FA intercepts all packets for MN from the MN’s home agent.
- *Correspondent Node (CN)*. CN is a communication partner (host) of MN.

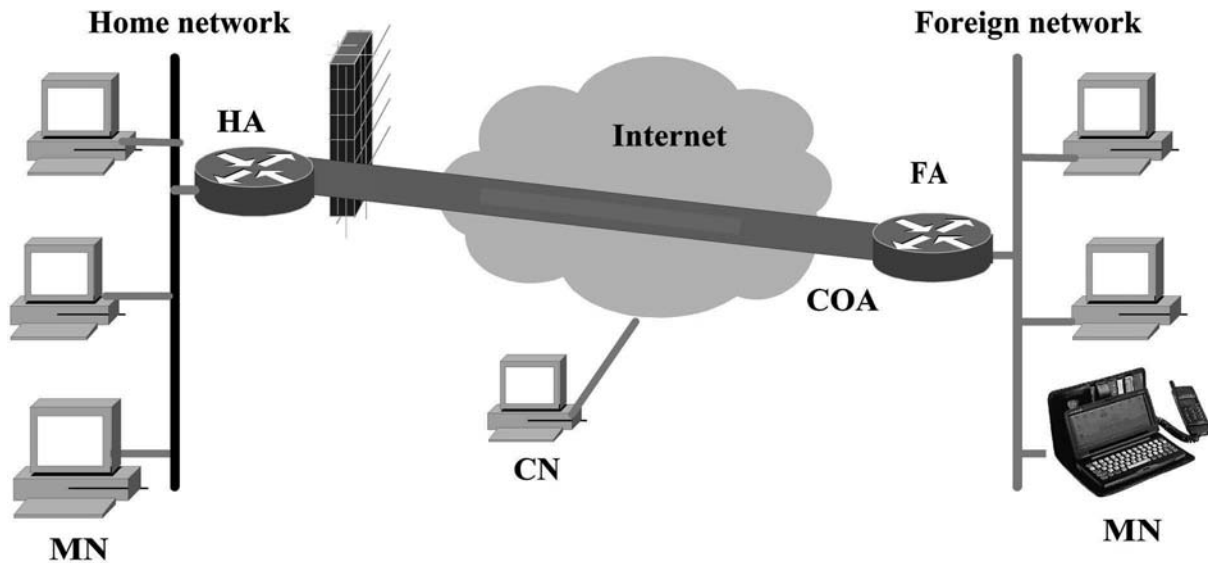


Figure 3-93: Mobile IPv4 – network architectural overview

The working principle of the mobile Ipve is described via the following steps:

1. *Discovery*: When arriving to a foreign network, a MN must firstly discover the foreign agent to obtain its care-of-address (COA).
2. *Registration*: After receiving the CoA from the foreign agent, MN must perform registration with its home agent to inform the HA of its CoA.
3. *Tunnelling*: If the registration is successful performed, the HA uses the CoA to tunnel packets intercepted from correspondent node to the foreign agent which then forwards these packets to the MN.

Since the IP address of a CN is fixed, the IP packets from MN to a CN travels directly across the Internet by using the CN's IP address.

Agent Discover	Registration	Tunneling
ICMP	UDP	
IPv4		

Figure 3-94: Mobile IPv4 protocol stack

For enable mobility in IPv4, three principle protocols (agent discovery, registration and tunnelling) are added into TCP/IP protocol stack (figure 3-94). The agent discovery enables home agents and foreign agent to advertise their availability on each link for which their provide service. This protocol is also used by a newly arrived mobile node to learn whether any prospective agents are present. The registration protocol functions between mobile node and its home

agent for performing the care-of-address registration. The tunnelling protocol operates between home agent and foreign agent to delivery packets from home network to the mobile node. Detail of these protocols will be described in the next following sections.

3.11.1.2 Agent Discovery

The agent discovery deals with methods used by a mobile node to determine whether it is currently connected to its home network or to a foreign network. The agent discovery also enables the mobile node to detect when it has moved from one network to another. When the mobile node is connected to a foreign network, the mechanism allows the mobile node to obtain the foreign agent care-of-address being offered by each foreign agent on that network.

As shown in figure 3-94, the mobile IPv4 agent discovery is based on ICMP router discovery function. To perform the agent discovery, two messages are specified in RFC 3220 [Per-2002]: the agent advertisement and the agent solicitation. The agent advertisement is an ICMP router advertisement that has been extended to carry a mobility agent advertisement extension and, optionally, a prefix-lengths extension, one-byte padding extension, or other extensions that may be defined in the future. An agent solicitation message is identical to an ICMP router solicitation with a restriction that the IP TTL field must be set to 1. The formats of these two messages are found in [Per-2002] and thus will not be discussed in this section.

The working principle of the agent discovery can be explained as follows. The foreign agent and home agent periodically issue the agent advertisement (AA) messages that carry its IP address (care-of-address) and the information about the role of the router as an agent. A mobile node listens for agent advertisement messages. By receiving the agent advertisement messages, a mobile node can determines whether it is currently connected to its home network or to a foreign network by comparing its fixed IP address with the IP address in these AA addresses. If it is connected to a foreign network, this mobile node also knows the care-of-address of the corresponding foreign agent determined in these AA messages. When a mobile node arrives to a foreign network and it did not received agent advertisement messages as well as it has no possibilities to get care-of-address, this mobile node then can issue agent solicitation messages. Any foreign agent receiving these agent solicitation messages will then issues an agent advertisement.

In order to maintain communication between mobile node and its foreign agent and home agent, following main considerations are required:

- *Registration with the foreign agent.* When the mobile node receives an agent advertisement, the MN should register through the foreign agent.
- *Move detection.* In order to detect the movement of a mobile node from one subnet to another one, two primary algorithms described in [Per-2002], can be implemented. The first method is based on the lifetime field of the agent advertisement message. Its main idea is that the mobile node records the life time received by any agent advertisements, until the life time expires. If the mobile node fails to receive another advertisement from the same agent within the specified lifetime, it assumes that it has lost contact with this agent. In that case, if the mobile node has previously received an agent advertisement from another agent for which lifetime field has not yet expired, the mobile node may immediately attempt registration with other agent. Otherwise, the mobile node should attempt to discover a new agent on which it should register. The second method uses network prefixes. By this method, the mobile node compares its prefix with the prefix of the foreign agent's care-of-address. If the prefixes differ, the mobile node may assume that it has moved.
- *Returning home.* A mobile node can detect that it has returned to its home network when it receives an advertisement from its own home agent. In that case, it should deregister with its home agent to inform it to stop tunnelling packets to the foreign network.

3.11.1.3 Registration

Registration is performed between a mobile node, a foreign agent, and the home agent of this mobile node. Registration creates or modifies a mobility binding at the home agent, associating the mobile node's home address with its care-of-address for the specified lifetime [Per-2002]. In particular, the registration procedure enables a mobile node to discover its home address and a home agent address if the mobile node is not configured with this address. Moreover the registration allows mobile node to maintain multiple simultaneous registrations. Also, it enables a mobile node to deregister specific care-of-address while retaining other mobility binding.

Two registration procedures are specified for the mobile IP, one via a foreign agent that relays the registration to the mobile node's home agent, and one directly with the mobile node's home agent. In both registration procedures, exchanging the registration request and registration reply messages is needed. The registration via foreign agent is illustrated in figure 3-95 (a), and figure 3-95 (b) shows the registration directly with the home agent. When registering via a foreign agent, four messages need to be sent via the registration procedure:

1. In order to begin the registration process, the mobile node sends a registration request to the prospective foreign agent.
2. When this registration request arrives at the foreign agent (FA), the FA processes it and then relays this registration request to the home agent.
3. The home agent then sends a registration reply to the foreign agent to accept or reject the request.
4. The foreign agent processes this registration replay and then relays it to the mobile node to inform it of the disposition of its request.

By directly registering with the home agent (figure 3-95 b)), only following two messages are required:

1. The mobile node sends a registration request directly to its home agent.
2. The home agent processes this request and sends a registration reply to the mobile node to grant or deny the request.

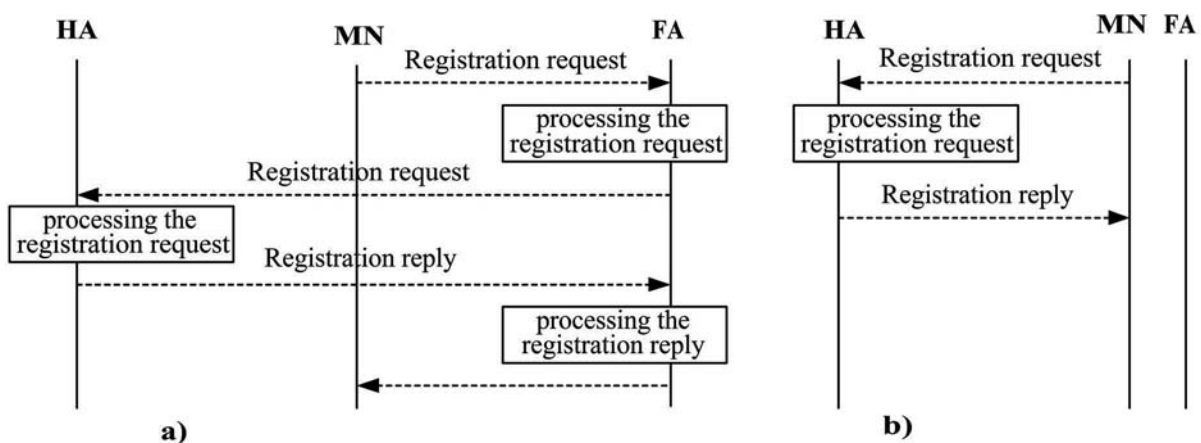


Figure 3-95: The registration procedure via (a) foreign agent and (b) via home agent

As shown in figure 3-95, registration request and registration reply messages are used for the mobile IP registration. Since the mobile IP uses UDP to perform its registration process, each registration message includes an UDP header followed by a set of mobile IP fields. The formats of the registration request and registration reply are described as follows:

- **Registration request.** In the IP header, the source IP address is typically the interface address from which the message is sent, and the destination address is the IP address of the foreign agent or the home agent. In the UDP header of this message, the source port is variable, but the destination port number is fixed defined with the value 434. The format of a registration request is illustrated in figure 3-96. The flags, which are defined from the 8th bit to the 15th bit, are used for different purposes e.g.

binding, broadcast tunnelling, decapsulation and encapsulation of datagrams. These flags are detail described in RFC 3220. The lifetime field indicates the time (in seconds) remaining before the registration is considered expired. The identification field constructed by the mobile node is used for matching registration requests with the registration replies and for protecting against reply attacks of registration messages.

Type	S	B	D	M	G	r	T	x	Lifetime
IP home address of mobile node									
IP home address of home agent									
Care-of-address									
Identification									
Extension									

Figure 3-96: Format of registration request message

- *Registration reply.* The reply message contains the necessary codes to inform the mobile node about the status of its requests, together with the lifetime grand by home agent, which may be smaller than the original request. The registration reply message consists of several fields as shown in figure 3-97. For example: the source address field is copied from destination address of registration request to which the agent is replying; the destination address is copied from the source address of the registration request to which the agent is replying; the source port field is variable and the destination port number field is copied from the source port of the corresponding registration request; The code field indicates the result of the registration request, e.g. registration timeout, invalid care-of-address. The other fields of the registration reply have the same meaning as by the registration request message.

Type	Code	Lifetime
IP home address of mobile node		
IP home address of home agent		
Identification		
Extension		

Figure 3-97: The format of registration reply message

3.11.1.4 Tunneling

Tunnelling is a mechanism that allows the mobile node to send and receive packets by using its home IP address. Even while the mobile node is roaming on foreign networks, its movements are transparent to the correspondent nodes. The data packets addressed to the mobile node are routed to its home network, where the HA intercepts and tunnels them to the care-of-address (the FA) towards the mobile node (see figure 3-98). The tunnelling has two main functions: encapsulation of data packets to reach the tunnel endpoint, and decapsulation when the packet is delivered at that endpoint.

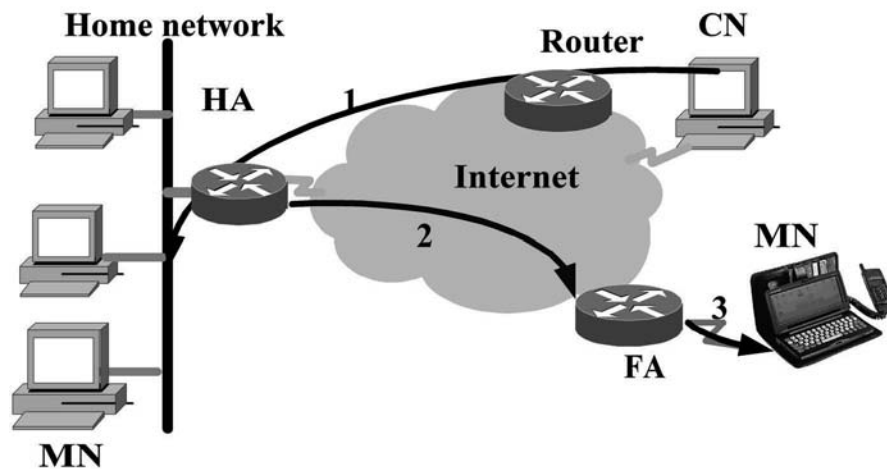


Figure 3-98: Packet forwarding by using tunnelling

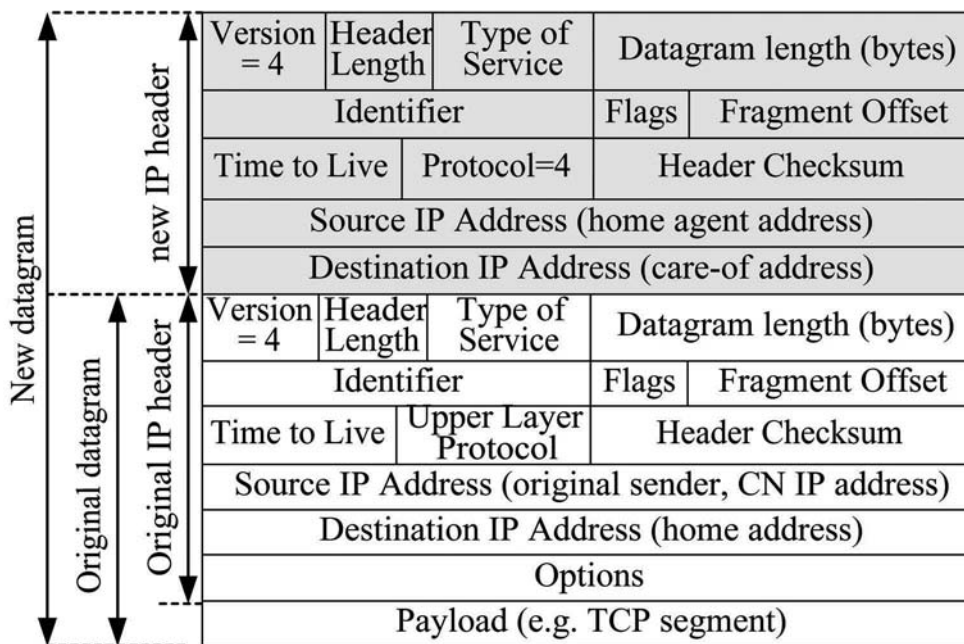


Figure 3-99: IP encapsulation within IP encapsulation

The default tunnel mechanism is IP encapsulation within IP encapsulation, by which the entire datagram becomes the payload in the new datagram shown in figure 3-99. The inner original IP header is unchanged except to decrement time to life (TTL) value by 1. The version field and ToS field are copied from the inner header.

3.11.1.5 Routing

The routing in mobile IP determines how mobile nodes, home agents and foreign agents cooperate to route datagrams to/from mobile nodes that are connected to a foreign network. In mobile IPv4, the routing is based on the so-called triangle routing shown in figure 3-100. When a correspondent node (CN) sends traffic to the mobile node, the packets are first intercepted at the home agent (HA), who encapsulates these packets and tunnels them to the foreign agent (FA). The foreign agent de-tunnels the packets and delivers them to the mobile node (MN). As shown in figure 3-100, the route taken by these packets is triangular in nature, and an extreme case of routing can be observed when communicating node and the mobile node are in the same subnet. For the datagram sent by the mobile node, standard routing is used.

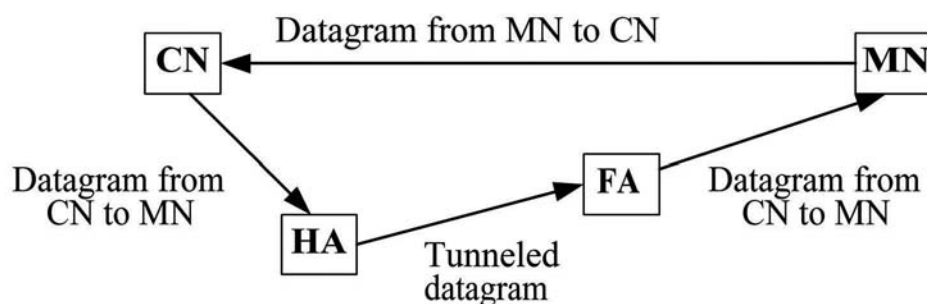


Figure 3-100: The triangle routing in mobile IPv4

3.11.2 Mobile IPv6

Mobile IPv6 is a standard specified in RFC 3775 [JPA-2004] proposed by the IETF working group “Mobility for IPv6”. Together with this specification, several proposals e.g. [Koo-2007, PG-2006, KKN-2006] for adding new functionalities into mobile IPv6 are specified by IETF. The following sections begin with an illustration of the architectural overview of the mobile IPv6. Based on this, protocol design aspects to support mobile IPv6 will be discussed. Finally, mobile IPv6 operations performed on the correspondent node, home agent and on the mobile node will be described in detail.

3.11.2.1 Architectural Overview

In a mobile IPv6 network, a mobile node (MN) is addressable at its home address and one or more care-of-addresses. While the mobile node stays at home, packets addressed to its home address are routed to mobile node's home link (home agent HA) by using conventional Internet routing mechanisms. Care-of-addresses are used for addressing the mobile node when it is attached to some foreign links away from its home. Acquiring the care-of-addresses for a mobile node can be done through conventional IPv6 stateless and stateful auto-configuration. The architectural overview of mobile IPv6 is shown in figure 3-101. In comparison with the mobile IPv4 architecture, the foreign agent (FA) is eliminated in the mobile IPv6. In particular, datagrams sent from correspondent node (CN) to the MN is intercepted by HA and directly tunneled to the MN.

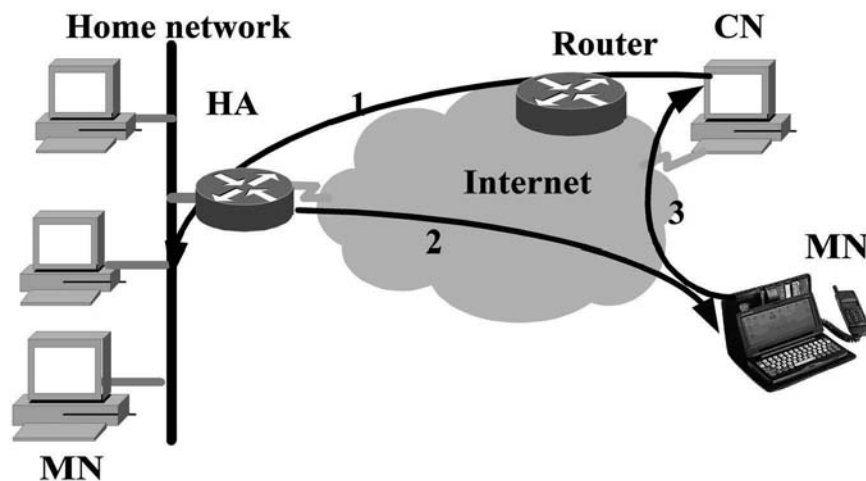


Figure 3-101: Mobile IPv6 – network architectural overview

The association between a mobile node's home address and care-of-address is known as "binding" that allows a mobile node to register its primary care-of-address with a router on its home network, requesting this router to operate as its home agent. The binding registration is performed via two messages: binding update message from MN to HA and binding acknowledgement message from HA to MN. Also mobile nodes can provide information about their current locations to the correspondent nodes through registration with the CN.

Data transmission between a mobile node and a correspondent node can be done via two possible modes: (1) bidirectional tunneling and (2) router optimization. The bidirectional tunneling mode does not require mobile IPv6 support from the correspondent node. Packets from CN are forwarded to the HA and then tunneled to the mobile node's care-of-address even if the MN has not registered its current binding with the CN. The second mode, the router optimi-

zation, requires the mobile node to register its current binding with the correspondent node. Packets from the CN can be forwarded directly to the care-of-address of the mobile node without interception by the HA and thus without tunneling. This mode allows the shortest communication path to be used and eliminates congestion at the mobile node's home agent and home link.

While going away from home network, two modes are used by mobile nodes to send packets to its correspondent node: route optimization and reverse tunneling. Using the route optimization mode, the MN sends packets directly to its CN. This manner of delivering packets does not require going through the home network, and thus will enable faster and more reliable transmission. With the reverse tunneling mechanism the MN tunnels packets to home agent, which then sends packets to the correspondent node. This mechanism is not as efficient as the route optimization mechanism, but it is needed if there is no binding with the correspondent node.

3.11.2.2 Protocol Design Aspects to Support Mobile IPv6

To support mobile IPv6, changing in IPv6, ICMPv6 and in IPv6 neighbor discovery protocol is required. This modification is described as follows:

- *IPv6 extension.* To support mobile IPv6, the mobility header as a new IPv6 extension header is defined. This new extension header is used by mobile node, correspondent node and by home agent in all messaging related to the creating and managing of bindings. Mobile IPv6 messages carried within this mobility extension header are e.g. binding update, binding acknowledgement, binding refresh request, binding error, and, messages used to perform the return route-ability procedure from a MN to a CN. Furthermore, mobile IPv6 also defines a new IPv6 destination option, the home address destination option. The mobility header is identified by a next header value of 135 in the immediately preceding header, and has the format shown in figure 3-102. The Payload Proto field is used to identify the type of header immediately following the mobility header. The MH type indicates the identifier of a particular mobility message. The checksum field contains the checksum of the mobility header; the message data is a variable length field containing the data specific to the indicated mobility header type.

Payload Proto	Header length	MH type	Reserved
Checksum			
IP home address of home agent			

Figure 3-102: The mobility header format

- ICMPv6 extension. In order to support mobile IPv6, four new ICMPv6 message types are also introduced. Two of these four messages, the home agent address discovery request and home agent address discovery reply, are used in the dynamic home agent address discovery mechanism. In particular, a home agent address discovery request is sent by MN to HA any-cast address to discover the address of a suitable HA on its home link. The response to this message is the home agent address discovery reply that gives the MN the addresses of HAs operating on its home link. The other two messages, the mobile prefix solicitation and the mobile prefix advertisement, are used for network renumbering and address configuration on the mobile node. When a MN has a home address that is about to become invalid, MN sends prefix solicitation message to request fresh prefix information. The response to this message is the mobile prefix advertisement sent by HA.
- IPv6 neighbour discovery extension. In order to indicate that the router sending the advertisement message is operating as HA, a flag bit is added to the router advertisement message. Since neighbour discovery only advertises a router's link-local address that is used as the IP source address of each router advertisement, modification of the prefix information format is required so that a list of HAs as a part of dynamic HA address discovery can be advertised.

Based on these extensions, the mobile IPv6 protocol is specified via operations at the CN, HA and at the MN. These operations will be found in the RFC 3775. In the following next sections, some operations that are not supported by mobile IPv4 will be discussed.

3.11.2.3 Movement Detection

Consider a mobile node connected to its home network. This mobile node opens a communication with a correspondent node before moving toward a foreign network. When the mobile node is connecting to a foreign network, it starts to detect its movement in a new network. The mobile node first acquires a care-of-address (CoA) for this foreign network. The CoA acquirement can be done by router advertisement messages, which are periodically sent by the foreign network routers, and, advertise a list of CoA for this mobile node. Based on these care-of-addresses, the mobile node knows whether it is connecting to its home network or to a foreign network.

3.11.2.4 Binding Update

If the mobile node is connecting to a foreign network and has obtained its care-of-address, it then registers this address with its home agent and correspondent node in order to make it as primary care-of address. This is done by sending the binding update messages as shown in figure 3-103. A binding acknowledgement may be sent by HA and CN to indicate the receipt of a binding update if the acknowledgement bit (A) is set in the binding update message, or, if the node rejects the binding update due to an expired nonce index, sequence number being out of window, or insufficiency of resources. If the node accepts the binding update and creates or updates an entry for this binding, the status field in the binding acknowledgement is set to a value less than 128. Otherwise, this field is set to a value greater than 128 [JPA-2004].

The processing of the binding update is performed by the home agent via the following sequence of tests:

- If the node implements only the CN functionality, or has not been configured to act as home agent, the node rejects the binding update and returns a binding acknowledgement to the mobile node to indicate that the registration is not supported.
- If the home address field (in the packet's home address option) is not an on-link Ipv6 address, the home agent rejects the binding update and returns a binding acknowledgement to the mobile node to indicate that the home address the MN wanted to register is not home subnet.
- If the home agent chooses to reject the binding update for other reason, it returns a binding acknowledgement to the MN, in which the status field is set to an appropriate value to describe the reason for the rejection.

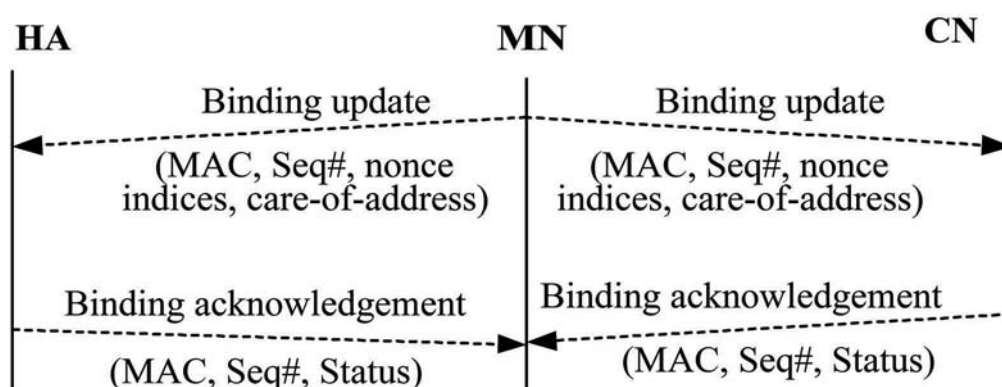


Figure 3-103: The binding update by the mobile node

3.12 Audio and Video Transport

Multimedia communication is the fastest growing emerging telecom sector in the last years. There is an explosive expansion in the development and deployment of network applications that transmit and receive audio and video contents over the Internet. New multimedia applications – IP telephony, Internet Protocol Television (IPTV), Internet radio, multimedia WWW sites, teleconferencing, interactive games, distance learning, and much more – seen to be announced daily.

The service requirements for multimedia applications differ significantly from traditional elastic applications. In particular, multimedia applications are highly sensitive to the end-to-end delay and delay variation, but can tolerate occasional losses. However, the TCP/IP network architectures together with protocols haven been designed primarily only for elastic applications. They are not designed to support multimedia applications and factors such as network delay, jitter and packet loss lead to a deterioration of perceived quality of voice and video. Therefore, new architectures and protocols, which offer services and provide QoS guarantees for multimedia applications, are developed in the last years.

In order to provide QoS for multimedia applications, two main approaches have been currently developed. The first approach relies on application-level QoS mechanisms to improve perceived QoS without making any change to the network infrastructure. New transport protocols such as RTP, RTCP and DCCP, and different compensation strategies for packet losses (e.g. Forward Error Correction (FEC)) and jitter are belonging to this approach. These new transport protocols will be discussed in this section.

The second approach relies on the network-level QoS mechanisms and the emphasis on how to guarantee IP network performance in order to achieve the required network QoS. The IntServ, DiffServ, MPLS architecture described in the previous sections and the architectures for VoIP and IPTV that will be discussed in this section are belong to this second approach.

This section provides a short survey of protocols and architectures for supporting the transport of audio and video over the Internet.

3.12.1 Transport Protocols

Most Internet applications use either TCP or UDP for data transfer. However these two general protocols do not ideally satisfy all applications, especially the multimedia applications. The main limitations of these protocols that users have wished to bypass include the following:

- UDP doesn't support any flow control and congestion control mechanisms. Therefore, streams from different servers may collide and thus it can lead to network congestion. There are no mechanisms for synchronizing between UDP sender and UDP receiver to exchange the feedback information that can be used to reduce the congestion and to improve the QoS.
- The TCP supports a reliable data transfer. Its strict order-of-transmission delivery of data generates the head-of-line blocking and thus causes unnecessary delay. Moreover, since TCP doesn't support multi-homing, its limitation complicates the task of providing robustness to failures. Furthermore, TCP is relatively vulnerable to denial of service attacks, such as SYN flooding.

Transmission of PSTN signalling and of video/audio data across the IP network requires such applications for which all of these limitations of TCP and UDP are relevant. These applications directly motivated the development of the new transport protocols, such as RTP, RTCP, SCTP and DCCP. An overview of these protocols within the protocol stack is shown in figure 3-104. These transport protocols will be described in this section.

Application layer	Data application	Audio/Video applications			
		RTP	RTCP		
Transport layer	TCP	UDP		SCTP	DCCP
Network layer	IP				
Physical layer	e.g. Ethernet/SDH				

Figure 3-104: Overview of the transport protocols for audio and video applications

3.12.1.1 Real Time Transport Protocol (RTP)

The real time transport protocol (RTP) [SCF-2003] developed within the IETF is the most widely used application layer protocol for real-time audio/video applications in the Internet. Most of the used conferencing applications such as VIC (Video Conferencing Tool) or VAT (Audio Conferencing Tool) support RTP. Moreover, the standards proposed for Internet telephony, such as H.322 or SIP, define RTP as the application level based transport protocol for the data.

RTP runs on top of the transport protocol UDP. Unlike UDP RTP provides audio/video applications with end-to-end delivery services such as payload type identification and delivery monitoring. RTP provides the transport of data with a notion of time to enable the receiver to reconstruct the timing information of the

sender. Applications using RTP will be provided sequence numbers, time stamps and QoS parameters. Nevertheless, RTP doesn't offer any mechanisms to ensure timely delivery, to promise the reliable delivery of packets or to prevent their out-of-order delivery, to provide the QoS guarantees and to control and avoid the congestion control. Thus, it is typically implemented as part of the applications or as library rather than integrated into the operating system kernel.

Each RTP session consists of two streams: a data stream for audio or video data packets and a control stream for control packets by using the sub-protocol Real Time Transport Control (RTCP). These two streams use separate ports.

The RTP basic principle for data delivering is illustrated in figure 3-105. At the application sending side, an RTP-based application collects the encoded data in chunks, encapsulates each chunk with an RTP header and sends these RTP packets into UDP socket interface. In the network layer, each UDP segment is encapsulated in an IP packet that is processed and forwarded via the Internet. At the application receiving side, RTP packets enter the application through a UDP socket interface. The application then extracts the media chunks from the RTP packets and uses header fields of RTP packets to properly decode and playback the audio or video chunks.

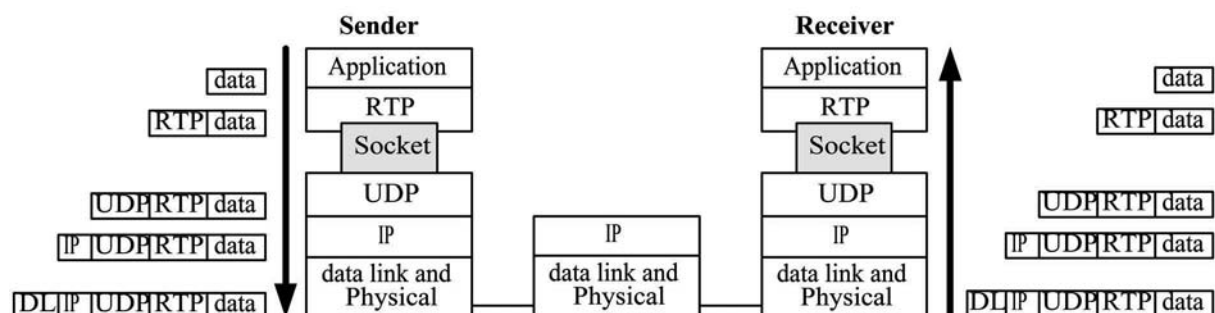


Figure 3-105: The RTP data delivery architecture

An RTP packet contains an RTP header followed by RTP payload. The format of a RTP data header is shown in figure 3-106. Some fields in the header are:

- *Version field (V)* indicates which version of RTP is being used where the version 2 is the common one.
- *Extension bit (X)* specifies whether an extension header follows the fixed header or not.
- *Sequence number* is incremented by one for each RTP data packet and can be used by the receiver for loss detection.
- *Timestamp* reflects the sampling instance of the first data sample contained in the payload of RTP data packets and is implemented by one

for each data sample, regardless of whether the data samples are transmitted onto the network or are dropped as silent. The timestamp helps the receivers to calculate the arrival jitter of RTP packets and synchronize themselves with the sender.

- *SSRC and CSRC* contain the identity of the sending source.

v	p	x	cc	M	Payload typ (7 bits)	Sequence number (16 bits)
Time stamp (32 bits)						
Synchronization source (SSRC) identifier (32 bits)						
Contributing source (CSRC) identifier (32 bits)						

Figure 3-106: RTP header

To monitor the QoS of a session or to trace the identify of a member in a session, each participant in a RTP session periodically sends RTCP control packets to all other session members using IP multicast.

RTCP packets do not encapsulate chunks of audio or video. Instead RTCP packets are sent periodically between senders and receivers in order to collect statistics on a media connection and to collect information such as packets sent, lost packets, bytes sent, inter-arrival time and round trip delay. The RTCP specification does not dictate what the applications should do with the feedback information. This is up to application developers. An application can use this information to regulate the sending rate or for diagnostic purposes. The RFC 3550 defines several RTCP packet types to carry variety of control information such as [SCF-2003]:

- *RR*: An RTCP receiver generates a reception report (RR) for each RTP stream it receives. These RRs are aggregated into a single RTCP packet which is sent via multicast to all participants in a session. Receiver reports (RR) consist of several entries. Each of these entries corresponds to one active receiver.
- *SR*: Sender reports (SR) contain information about the amount of sent data and the time this report was generated. A SR consists of several fields, e.g. SSRC, timestamp, total number of RTP data packets transmitted by the sender since starting the transmission up until the time this SR packet was created, total number of payload bytes and inter-arrival jitter
- *SDES*. The source description packets (SDES) include the identification information about the source.
- *BYE*. This packet is sent from a participant when he leaves the conference.
- *APP*. The application packets (APP) contain application specific information and can be used for experimental purposes.

- The primary function of RTCP is to provide feedback on the QoS being provides by RTP

Since RTCP control traffic may consume a lot of bandwidth by a large session size. To overcome this problem, RTCP provides a method which tries to limit control traffic, usually around 5% of the session bandwidth and is it divided among all participants. Based on the length of the RTCP packets and the number of members, each participant can determine the interval between sending two RTCP packets.

Also the senders can estimate the round trip delay to the receivers using the RTCP packets. The senders include in their RTCP messages a timestamp indicating the time the report was generated. For each incoming stream, the receivers send a report indicating the timestamp of the last received sender report (t_{lsr}) for that stream and the time between receiving the last sender report and sending the receiver report (t_{lrr}). Knowing the arrival time (t) of the RTCP packet the sender can calculate the round trip time (t_{rtt}):

$$t_{rtt} = t - t_{lrr} - t_{lsr}$$

This calculation doesn't require synchronisation between the clocks of the sender and receiver and therefore it is rather accurate.

3.12.1.2 Stream Control Transmission Protocol

Stream Control Transmission Protocol (SCTP) is a new transport protocol, existing at an equivalent level with UDP and TCP, which provides transport layer services to many Internet applications. SCTP is an IETF standard specified in RFC 4166, RFC 3286 and RFC 2960 [CB-2006, OY-2002, SXM-2000].

Like TCP, STCP provides a reliable, full-duplex connection and mechanisms for congestion control. Unlike TCP and UDP, SCTP offers new mechanisms, such as multi-homing and multi-streaming, which are particularly desirable for telephony signalling and multimedia applications.

3.12.1.2.1 SCTP Packet Format

Each SCTP packet consists of a common header and one or more data chunks (figure 3-107). The common header includes the following fields

- Source and destination port numbers: used together with the IP addresses to identify the association to which an SCTP packet belongs to.
- Verification tag: used by the receiver to validate the sender of this SCTP packet.
- Checksum: acts as a data integrity tool for each SCTP packet.

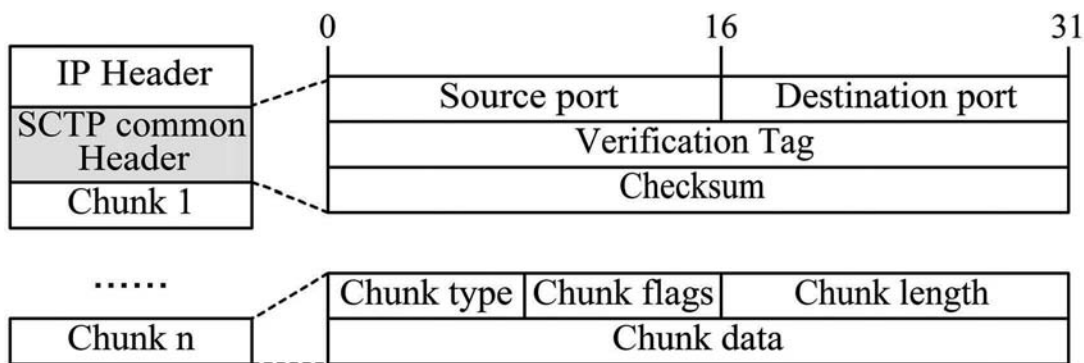


Figure 3-107. Sctp packet format

The remainder of Sctp packets contains one or more chunks. Chunks are concatenated building blocks that contain either control or data information. The fields within a chunk can be described as follows:

- *Chunk type*: identifies the type of chunk being transmitted
- *Chunk flag*: specifies whether bits will be used in the association
- *Chunk length*: determines the size of the entire chunk in bytes
- *Chunk data*: has variable length and includes the actual information to be transferred in the chunk

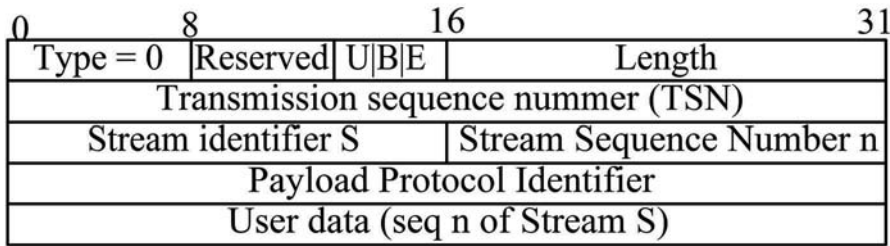


Figure 3-108: Data chunk format

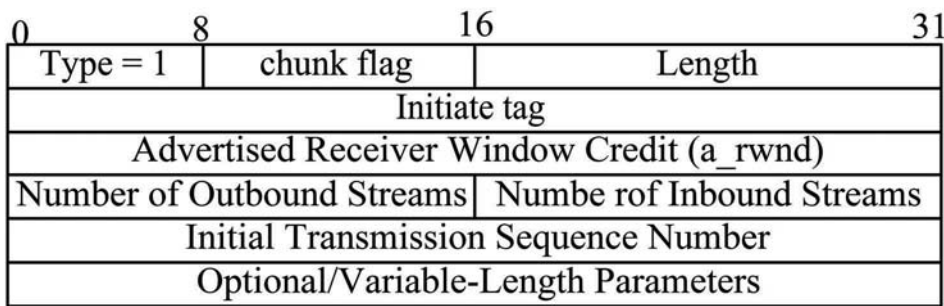


Figure 3-109: Control chunk format

There are 14 types of chunks - 13 types of control chunks (e.g. association establishment, association termination, data acknowledgement, destination fail detection) and one DATA chunk containing the actual data payload. The format of data chunks and control chunks are illustrated in figure 3-108 and 3-109.

3.12.1.2.2 SCTP Protocol Mechanisms

SCTP is a reliable transport protocol operating on top of IP. It provides the following protocol mechanisms: association phases, user data fragmentation, Multi-homing, Multi-streaming, and congestion control. These protocol mechanisms are summarized in this section.

3.12.1.2.2.1 Association phases

An SCTP association has three phases: association establishment, association shutdown, and data transfer.

- Association establishment:** SCTP uses a four-way handshake with a cookie mechanism that establishes an association to prevent blind SYN attacks. If host A initiates an association with host B, the following process is performed (figure 3-110): (1) An INIT chunk is sent from host A to host B. (2) when host B receives the INIT chunk, it replies with INIT-ACK; This INIT-ACK holds a cookie composed of information, which is verified by host B to check if host A is legitimate. (3) When host A receives the INIT-ACK, it returns a COOKIE-ECHO chunk to host B; this chunk may contain the first data of the host A and the cookie sent from host B. (4) On receiving the COOKIE-ECHO chunk, host B checks the cookie's validity. If it is valid, host B sends a COOKIE-ACK to host A. and only at this point, a association is established between host A and B, and the resource is allocated at host A and B. This SCTP's four-way handshake, in which a cookie mechanism establishes an association, prevents SYN attacks concerned with the TCP's three-way handshake.

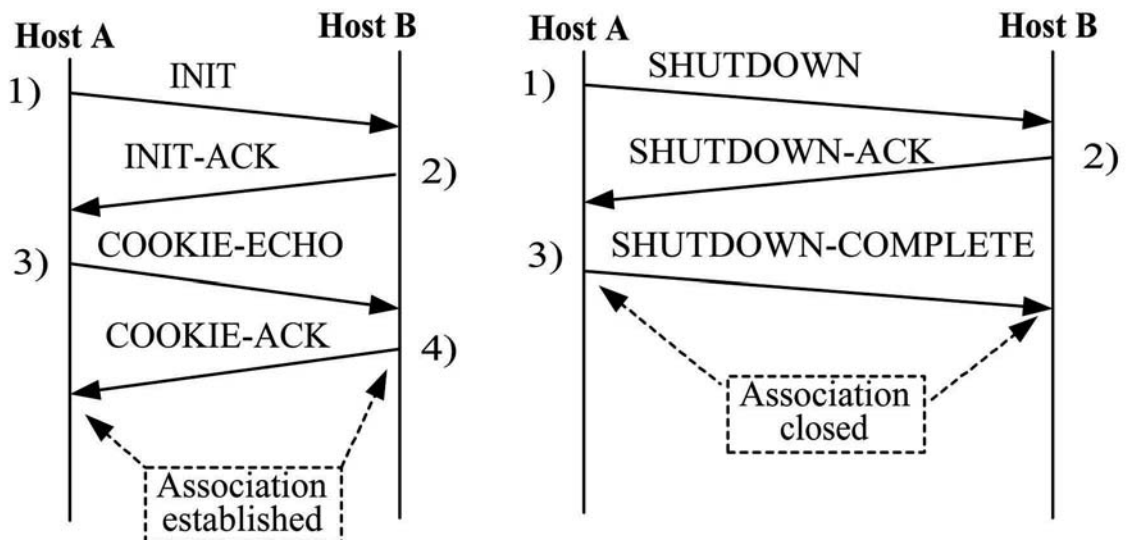


Figure 3-110: Association establishment and shutdown

- *Association shutdown.* In comparison to four way handshake of TCP, SCTP's association shutdown is a three-way handshake that does not allow half-closed connections, in which one end point shuts down while the other end point continues sending new data. Reason for this new design is that half-close was not used often enough in practise to warrant extra complexity in the SCTP shutdown procedure [CIA-2003].
- *Data transfer.* The transfer of SCTP's data chunks between a STCP sender and a STCP receiver over the Internet is performed via a combination of a set of mechanisms that provide reliability, congestion control, flow control, fragmentation, multi-homing and multi-streaming (figure 3.111). These mechanisms are described in [SXM-2000, CIA-2003] as follows.

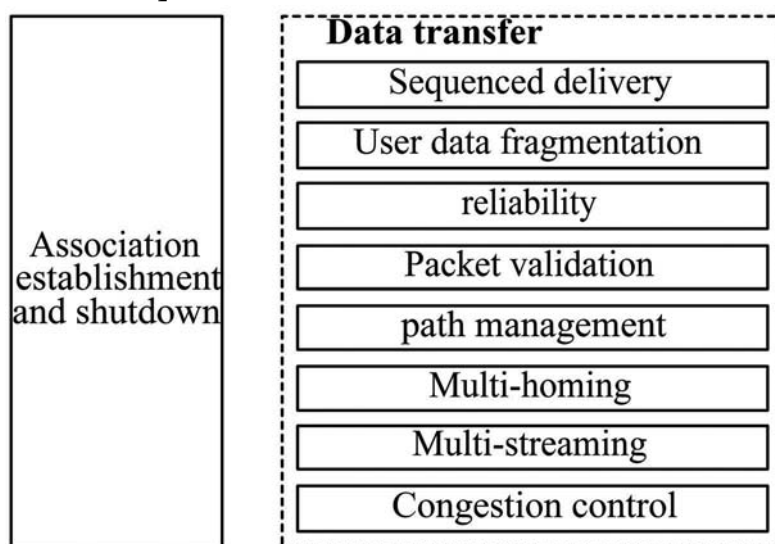


Figure 3-111: SCTP's protocol mechanisms

3.12.1.2.2.2 Sequence Number Delivery

In order to support reliability and congestion control, each STCP's chunk is assigned by a transmission sequence number (TSN) that is unique within an association. While TCP associates a sequence number with each data byte and hence wraps around faster, SCTP's sequence numbers only need to be associated with data chunks.

3.12.1.2.2.3 User Data Fragmentation

As opposed to TCP, which offers byte-oriented data transmission, SCTP's data transmission is message-oriented, similar to UDP. When an application has a message larger than destination path MTU, SCTP fragments this message into multiple data chunks, which can be sent in separate packets.

3.12.1.2.2.4 Reliability

Like TCP, SCTP maintains reliability through acknowledgements, retransmissions, and end-to-end checksum. In order to verify the packet, SCTP uses the 32 bit CRC checksum. SCTP acknowledgements carry cumulative (CumAck) and selective (GapAck) information. The CumAck indicates that the TSNs are received in sequence, and the receiver sets the CumAck to the last TSN successfully received in sequence. The GapAck blocks indicate that TSNs are received out of order beyond the CumAck.

3.12.1.2.2.5 Packet Validation

SCTP uses the value in the verification tag and 32 bit checksum field to validate the packets. The verification tag value is selected by each end of the association during association establishment. Packets received without the expected verification tag value are discarded.

The 32 bit checksum is sent by the sender of each SCTP packet. The receiver of an SCTP packet with an invalid checksum number silently discards the packet.

3.12.1.2.2.6 Path Management

The SCTP path management mechanism includes following functions:

- Selecting the destination transport address for each outgoing SCTP packet based on the SCTP user's instruction and the currently perceived reach-ability status of the eligible destination set.
- Monitoring the reach-ability through heartbeats and advising the SCTP user when reach-ability of any fair-end transport address changes.
- Reporting the eligible set of local transport addresses to the far end and during association establishment, and reporting the transport addresses returned from the far and to the SCTP user.

3.12.1.2.2.7 Multi-homing

Multi-homing enables the network redundancy at multiple network layers. It provides uninterrupted service during resource failures. SCTP supports multi-homing at the transport layer. A multi-homed SCTP end point (host) is accessible through more than one network interface and therefore through multiple IP addresses when that end point initialises an association. If one of its addresses fails, which is caused possible from interface or link failure, the destination host still receives data through an alternative interface. Currently, SCTP uses multi-homing only for redundancy, and not for load balancing.

SCTP keeps track of each destination address's reach-ability through two mechanisms: acknowledgements of data chunks, and heartbeat chunks. RFC 2960 [SXM-2000] specifies that if six consecutive timeouts occur on either data or heartbeat chunks to the same destination, the sender concludes that the destination is unreachable and selects an alternative destination address dynamically.

3.12.1.2.2.8 Multi-streaming

An SCTP association is like a TCP connection except that SCTP supports multiple streams within an association. All streams within an association are independent but related to the association. During the association establishment, the SCTP end point negotiates application-requested streams that exist for the life of the association. Within streams, SCTP uses stream sequence numbers to preserve the data order and reliability for each data chunk. Between streams, no data order is preserved. Thus, this approach avoids the TCP's head-of-line blocking problem, in which successfully transmitted segments must wait in the receiver's buffer until a TCP sending end point retransmits any previously lost segments.

3.12.1.2.2.9 Congestion Control

The SCTP congestion control algorithms are based on the TCP congestion control mechanisms specified in RFC 2581 [APS-1999]. The biggest difference between SCTP and TCP is the multi-homing feature. This difference leads to the distinction in the congestion control of these protocols. This section summarizes the difference of the SCTP congestion control from the TCP congestion control described in RFC 2961.

The different IP addresses by the SCTP multi-homing lead to different data paths between the two end points and thus to different destination addresses. The sender uses the same destination address until being instructed by the upper layer. SCTP may change to an alternative destination when it recognizes that the actual address is inactive.

Like TCP, SCTP implements slow start, congestion avoidance, fast retransmit and fast recovery phases. In comparison with the TCP congestion control that applied to a TCP connection and thus to a stream, the congestion control by SCTP is always employed in regard to the entire association and not to individual streams. Like TCP, SCTP uses three control variables to regulate its transmission rate: receiver advertised window size (rwnd), congestion window (cwnd) and slow start threshold (ssthresh). SCTP requires one

additional control variable, `partial_bytes_acked`, which is used during the congestion avoidance phase to facilitate `cwnd` adjustment.

The multi-homing leads to different destination addresses for a given SCTP sender. In order to enable congestion control for multi-homing, the SCTP sender keeps a separate set of congestion control parameters (e.g. congestion window (`cwnd`), slow start threshold (`ssthresh`), and `partial_bytes_acked`) for each of the destination addresses of its peer. Only the receiver advertised window size (`rwnd`) is kept for the whole association. For each of the destination addresses, an end point does slow-start upon the first transmission to that address.

3.12.1.3 Datagram Congestion Control Protocol

The Datagram Congestion Control Protocol (DCCP) is a newly specified transport protocol that exists at an equivalent level with UDP, TCP and STCP. This protocol provides bi-directional unicast connections of congestion-controlled unreliable datagrams. DCCP can be used to date most such applications that have used either TCP, whose reliability and in-order delivery semantics can introduce arbitrary delay, or UDP that doesn't support any congestion control mechanism. DCCP is an IETF standard specified in RFC 4340, RFC 4341 and RFC 4342 [KHF-2006, PK-2006, FKP-2006].

DCCP provides an unreliable end-to-end data transmission service for unicast datagrams, but a reliable end-to-end acknowledgement transmission. It also offers a reliable handshake for connection establishment and teardown and a reliable negotiation of features. The biggest difference to other transport protocols is that DCCP enables applications to choice of modular congestion control mechanisms. DCCP is suitable for use by applications such as streaming multimedia, Internet telephony, and on-line games.

3.12.1.3.1 DCCP Packet Format

The DCCP header can be from 12 to 1020 bytes. The DCCP header is illustrated in figure 3-112. It consists of a generic header, additional fields and Options filed. DCCP generic headers can have different forms depending on the value of the extended sequence number field `X`. If `X` is zero, only 24 bits of sequence number are transmitted, and the generic header is 12 byte long. If `X` is one, the Sequence Number field is 48 bits long and the generic header takes 16 bytes.

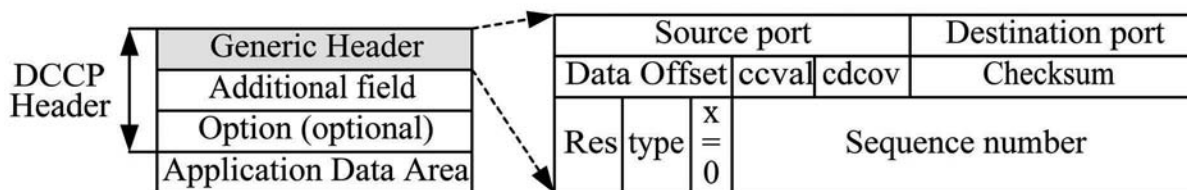


Figure 3-112: DCCP header and generic header

The generic header fields are defined in [KHF-2006] as follows:

- o *Source and destination ports* (16 bits each): Identify the connection, similar to the corresponding fields in TCP and UDP.
- o *Data Offset* (8 bits): indicates the size of DCCP header (from the start of the packet's DCCP header to the start of its application data area).
- o *Ccval* (4 bits): specifies the Congestion Control Identify (CCID) to be used
- o *Cscov* (checksum coverage, 4 bits): determines the path of packet that are covered by the checksum field.
- o *Checksum* (16 bits): is the Internet checksum of the packet's DCCP header.
- o *Type* (4 bits): specifies the type of the DCCP packet.

Currently, 10 packet types are implemented in DCCP (e.g. DCCP-Request, DCCP-Response, DCCP-Data etc.) Each of these packets has a common generic header format and additional specific fields and Options fields in the DCCP header.

3.12.1.3.2 DCCP Protocol Mechanisms

DCCP provides a congestion-controlled service and an unreliable end-to-end data transmission for unicast datagrams, but a reliable end-to-end transmission for acknowledgements. It also offers a reliable handshake for connection establishment and teardown and a reliable negotiation of features. The biggest difference to other transport protocols is that DCCP enables applications to choice of modular congestion control mechanisms. DCCP is suitable for use by applications such as streaming multimedia, Internet telephony, and on-line games.

3.12.1.3.2.1 Connection setup and teardown

DCCP connection establishment phase consists of three-way handshake: an initial DCCP-Request packet sent by the client, a DCCP-Response sent by the server in reply, and finally an acknowledgement from the client, usually via a

DCCP-Ack or DCCP-DataAck packet. DCCP-Request packets commonly carry feature negotiation options that open negotiations for various connection parameters, such as preferred CCIDs, ECN-capable, initial sequence number. In the second phase of the three-way handshake, the server sends a DCCP-Response message to the client. With this response message, the server will specify the features it would like to use, such as the CCID is expected to be used at the server. The server also may respond to a DCCP-Request packet with a DCCP-Reset packet in order to refuse the connection.

DCCP connection teardown uses a handshake consisting of a DCCP-CloseReq packet, a DCCP-Close packet, and a DCCP-Reset packet. The sequence of these three packets is used when the server decides to close the connection but don't want to hold the time-wait state. The server can decide to hold the time-wait state by using the sequence of DCCP-Close packet and DCCP-Reset packet [KHF-2006].

3.12.1.3.2.2 Reliable Acknowledgement Transmission

Congestion control requires that receivers transmit information about packet losses and ECN marks to the senders. DCCP receivers report all congestion events they experience, as defined by CCID profile. DCCP acknowledgements are congestion-controlled and require a reliable transmission service. In order to do it, each CCID defines how acknowledgements are controlled when congestion occurs. For example, on a half-connection with CCID 2 (TCP-Like), the DCCP receiver reports acknowledgement information using Ack vector option giving a run-length encoded history of data packets received at this receiver.

3.12.1.3.2.3 Congestion Control Mechanisms

In order to attract developers, DCCP aims to meet application needs as much as possible without grossly violating the TCP friendliness. But unlike TCP, DCCP applications have a choice of congestion control mechanisms. In fact, the two haft-connections can be administrated by different congestion control mechanisms, which are denoted by congestion identifiers CCIDs. During connection establishment, the endpoints negotiate their CCIDs. Each CCID describes how the half-connection sender regulates data packet rate and how the half-connection receiver sends congestion feedbacks through acknowledgements.

Currently, only CCIDs 2 and 3 are implemented. CCID 2 provides TCP-like congestion control described in section 3.5.4.2. And CCID 3 offers TCP-Friendly Rate Control presented in section 3.5.4.1.

3.12.1.3.2.4 Explicit Congestion Notification (ECN)

DCCP is fully ECN-aware [KHF-2006]. Each CCID specifies how its endpoints react to ECN marks. Unlike ECN by TCP, DCCP allows senders to control the rate at which acknowledgements are generated. Since acknowledgments are congestion controlled, they qualify as ECN-capable transport. Like TCP, a sender sets ECN-capable transport on its IP headers unless the receiver doesn't support ECN or the relevant CCID disallows it.

3.12.1.3.2.5 Feature Negotiation

DCCP endpoints use Change and Confirm options to negotiate and agree on a set of parameters (e.g. CCIDs, ECN capability, and sequence number) during the connection establishment phase.

3.12.2 Architectures

This section summarizes the architectures that enable the transport of audio and video over the Internet.

3.12.2.1 Voice over IP

Voice over IP (VoIP) is the technology for transmission of voice over a data network using the Internet protocol. Common VoIP network connections normally include the connections from phone to phone, phone to PC (IP terminal or H.323/SIP terminal) or PC to PC, as shown in figure 3-113. The circuit switched network can be a wired or wireless network, such as PSTN, ISDN or GSM.

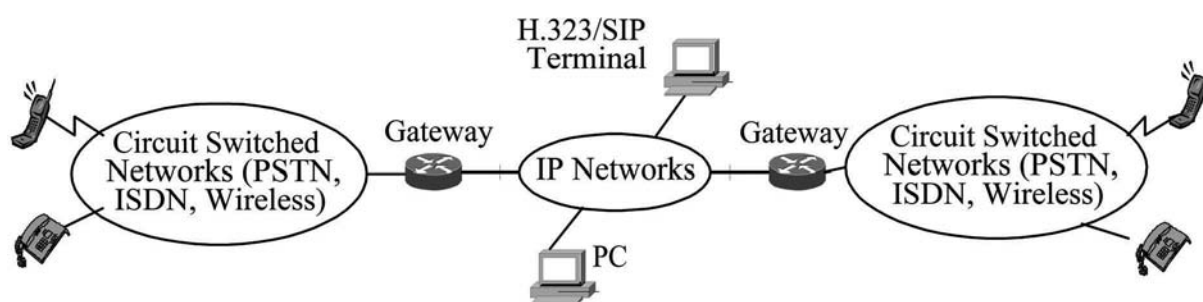


Figure 3-113: VoIP network connections

3.12.2.1.1 VoIP Protocol Architecture

In order to support VoIP, a set of protocols illustrated in figure 3-114 is developed. In this figure, the protocols that provide basic voice data transport

(RTP), QoS feedback (RTCP) and call-setup signalling (H.323 and SIP) are shown.

The signalling part with H.323 and SIP is illustrated in the section 3.9.3 above. RTP and RTCP are found in 3.12.1.1.

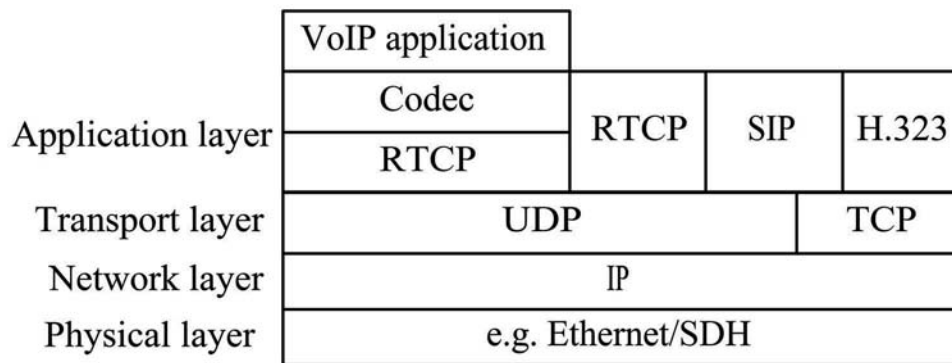


Figure 3-114: VoIP protocol architecture

3.12.2.1.2 VoIP System Structure

A basic VoIP system (signaling part is not included) is shown in figure 3-115. This system structure consists of three parts: the sender, the IP network and the receiver. At the sender, the voice stream generated from a voice source is first digitized and compressed by the encoder. Then, several coded speech frames are packetized to form the payload part of a voice packet (e.g. RTP packet). The headers (e.g. IP/UDP/RTP) are added to the payloads and form packets, which are sent into IP networks. These voice packets may suffer different network impairment (e.g. packet loss, delay and jitter) in IP networks. At the receiver, the packet headers are stripped off and the speech frames are extracted from the payload by depacketizer. Play-out buffer is then used to compensate for the network jitter at the cost of further delay (buffer delay) and loss (late arrival loss). The de-jittered speech frames are decoded to recover speech with lost frames concealed from previous received speech frames.

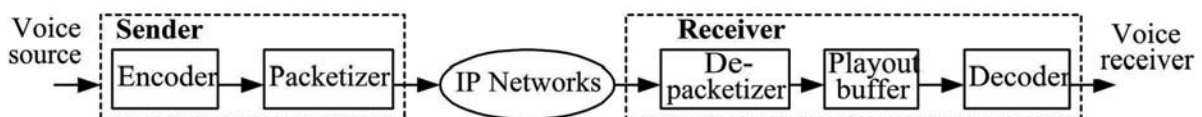


Figure 3-115: VoIP system structure

3.12.2.2 Internet Protocol Television (IPTV)

Internet Protocol Television (IPTV) is a technology for delivering broadcast TV and for providing other media-rich services using the Internet Protocol (IP) over

a broadband IP network with desired QoS to the public with a broadband internet connection. IPTV broadly encompasses a rich functionality ranging from acquisition, encoding and decoding, access control and management of video content, to delivery the digital TV, movies on demand, viewing of stored programming, personalized program guides.

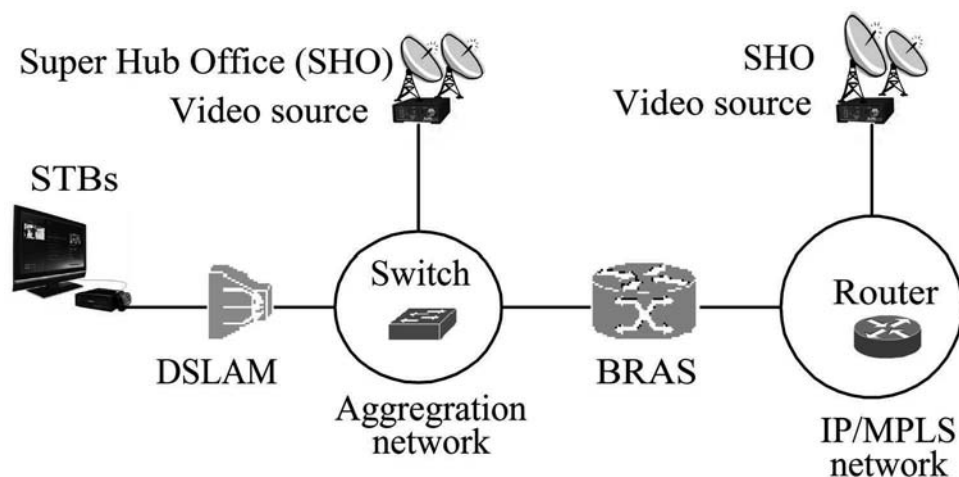


Figure 3.116: IPTV delivery Infrastructure: Access and home networks

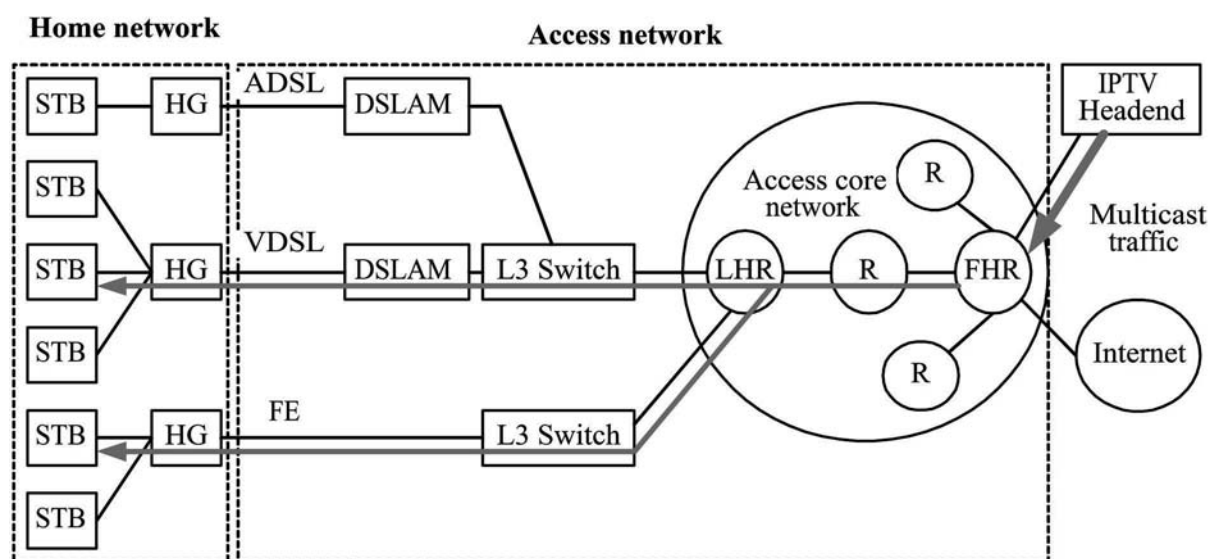


Figure 3.117: Network architecture for IPTV

The process of IPTV delivery is shown in figure 3.116 and 3.117. Basically, the IPTV traffic (local video source and national/regional video source) is sourced at the so-called IPTV headend.

IPTV headend is connected to an edge router. We call this router First Hop Router (FHR). Video streams are created on IPTV headend and sent via

multicast from to the FHR. The multicast video streams may be further processed and transmitted to the Last Hop Routers (LHRs) via several multicast routers in the access network. LHR is the last multicast router that any multicast streams go through and the first multicast router that is connected with the Home Gateway (HG), which is a device connecting home network and access core network.

Multicast streams are then transmitted out towards the customers at the home networks via the DSL Access Multiplexer (DSLAM). At the home network, the IPTV client, such as a set-top box (STB), is the functional unit, which terminates the IPTV traffic at the customer premises. In order to route traffic to and from DSLAM on an Internet Service Provide core, a broadband remote access router (B-RAS) is used.

In order to deliver IPTV traffic, compression methods as well as transport protocols for IPTV and IP multicast are required.

3.12.2.2.1 The IPTV System Architecture

For the IPTV delivery infrastructure shown in figure 3-116, a generic IPTV system architecture recommended in ITU recommendation H.610 is described in figure 3-118.

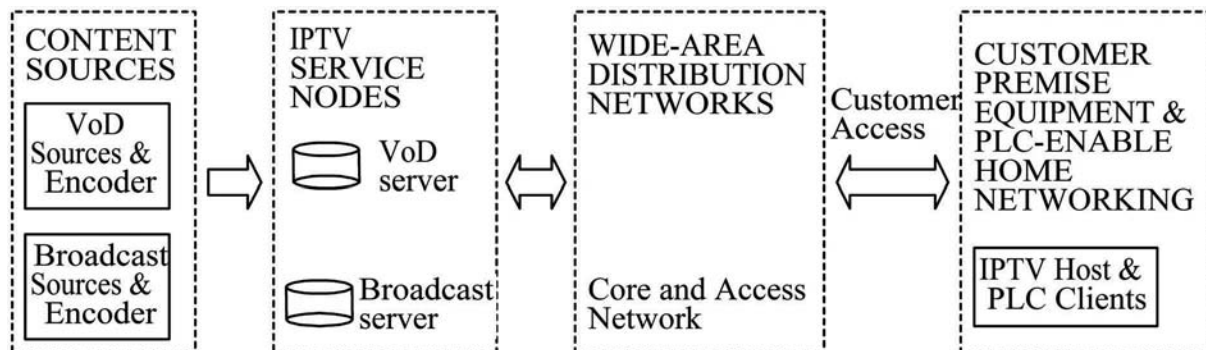


Figure 3-118: IPTV System Architecture

An IPTV system is made up of following major functional components:

- o *Content Sources* (Headends). As with a digital cable or digital satellite television system, an IPTV service requires content sources that receive video content from produces and other sources, encoding the content, capturing and formatting the content for distribution into an IP network. A variety of equipments may be Headend including satellite dishes to receive signals, content decoders and encoders, media servers and media gateways.

- o *IPTV Service Nodes.* The Service Nodes provide the functionality to receive video streams in various formats, and to encapsulate them with appropriate QoS indications for delivering to customers. Service nodes communicate with the Customer Premise Equipments (CPE) through Wide-Area Distribution Networks for providing services, session and digital right management. Service nodes may be centralized or distributed in a metro area.
- o *Wide-Area Distribution Networks.* These networks are responsible for TV distribution and QoS assurance that are necessary for reliable and timely distribution of IPTV data streams from the Service Nodes to the Customer Premises. Core and the Access Networks consist of optical distribution backbone networks and various Digital Subscriber Line Access Multiplexers (DSLAMs) located at the central office or remote distribution points.
- o *Customer Access Line.* At the customer site, IPTV access to homes is available over a existing loop plant and phone by using the higher-speed DSL technologies (e.g. ADSL2+, VDSL).
- o *Customer Premises Equipment (CPE).* A CPE device located at the customer premise provides the broadband network termination (B-NT) functionality, and may include other integrated functions such as routing gateway, set-top box (STB) and home networking capabilities.
- o *IPTV Client.* This is a device, such as a set-top box (STB), which performs a set of functions including the setting up the connection and QoS with the Service Node, decoding the video streams, changing the channel, controlling the user display.

3.12.2.2.2 Protocols for IPTV

Figure 3-119 and 3-120 show the protocol stacks that each component in the network should support to enable the IPTV services.

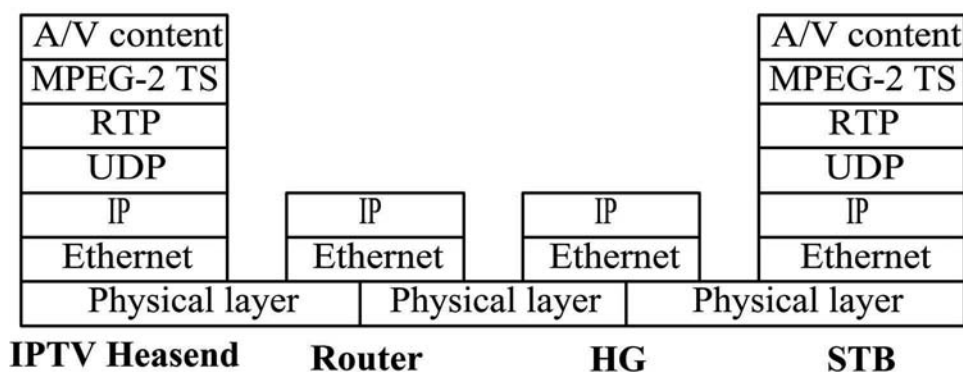


Figure 3-119: Transport Protocols for IPTV

The IPTV headend encapsulates MPEG-2 contents with MPEG-2 TS, enveloping the MPEG-2 TS packets with RTP, UDP, and IP multicast, and sending them to the network. Multicast routers in the core and access network forward the multicast packets to the right directions using the destination address of the multicast packets. The multicast packets arrive at the home gateway (HG), which forwards these packets to the home network based on destination addresses of them. At the home network, STB extracts the MPEG-2 TS packets from the IP packets, multiplexes them, decodes them, and renders them.

IP multicast services are used for delivering the TV contents to many receivers simultaneously. Figure 3-120 shows the protocol stacks for the channel join and leave in IPTV service. The IGMP protocol is used at the STB for joining and leaving the channel. It works as follows. The home gateway sends IGMP join and leave messages to its upstream router LHR, and responds to IGMP query messages of the upstream routers on behalf of hosts in the home network. LHR should support both IGMP protocol and multicast routing protocol, for example PIM-SSM. LHR receives IGMP join or leave messages from home gateways and sends IGMP query messages to them. At the same time LHR uses multicast routing protocol messages for notifying the other routers that the memberships of the hosts are changed.

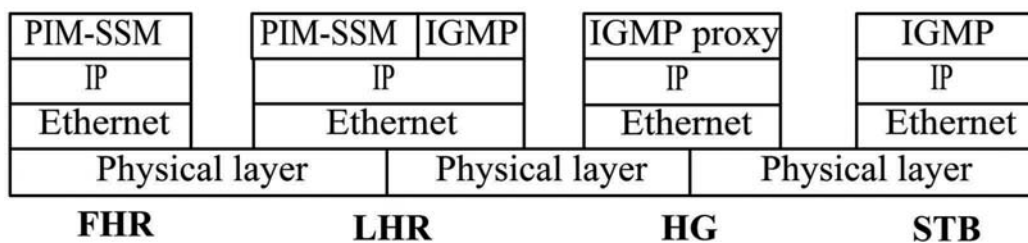


Figure 3-120: IP multicast protocols for IPTV

3.13 Virtual Private Networks

A Virtual Private Network (VPN) is a private communication network consisting of multiple remote peers (sites and users) that makes use of a network infrastructure (generally a shared IP backbone) such as Internet to securely transmit private data to one another. VPNs can be used within a single organization or several different organizations. In a VPN, inbound and outbound network traffic are protected by using tunnels that encapsulate all data at IP level. The main purpose of a VPN is to provide a company with capabilities of

private leased lines at a much lower price by using the Internet as a shared public network infrastructure with open transmission protocols.

3.13.1 VPN Devices

Before we describe various VPN technologies and protocols, it is useful to explain the customer and provider network devices. These devices fall into following categories:

- *Customer (C) devices*: C devices are routers and switches located within a customer network. These devices do not directly connect to the service provider network and thus they are not aware of the VPN.
- *Customer Edge (CE) devices*: CE devices are located at the customer networks and directly connect to the provider network via provider edge devices. In CE-based VPNs, CE devices are aware of VPNs. But in PE-based VPNs, CE devices are unaware of the VPN. CE devices are classified either as CE routers or CE switches.
- *Service Provider (P) devices*: P devices are devices such as routers and switches within the provider network that do not directly connect to customer networks. Thus, P devices are unaware of customer VPNs.
- *Service Provider Edge (PE) devices*: PE devices connect directly to customer networks via CE devices. PE devices are aware of the VPN in the PE-based VPNs, but are unaware of VPNs in CE-based VPNs.

3.13.2 Classification of VPNs

VPNs can be either service provider provisioned VPNs (PPVPNs) that are configured and managed by a service provider, or customer provisioned VPNs that can be configured and managed by customers. Customers of a service provider may be either an enterprise or another service provider. Additionally, a VPN service might be offered over backbone networks of multiple cooperating autonomous systems and/or service providers.

The provider provisioned VPNs and customer provisional VPNs can be categorized into two types: remote access VPNs and site-to-site VPNs. The service development of these VPN types ranges from service providers delivering managed services to companies building and managing their own VPNs.

3.13.2.1 Site-to-Site VPNs

Site-to-Site VPNs allow connectivity between organisation's geographically dispersed sites, such as head office and branch office. Figure 3-121 illustrates a

typical Site-to-Site VPN. There are two types of Site-to-Site VPNs: Intranet VPNs and Extranet VPN [CISCO-2005].

- *Intranet VPNs:* In order to provide an internal access to central repositories of information, corporations normally connect remote sites via leased line or frame relay. This approach results in the recurring costs for the dedicated links. These costs rise with the amount of bandwidth and the distance between sites. To reduce these costs, Intranet VPNs can be used for allowing connectivity between sites of a single organization. With an Intranet VPN, a company can replace an expensive dedicated link with a less expensive connectivity via the Internet to dramatically reduce bandwidth charges, since Internet connection is not distance sensitive.
- *Extranet VPNs.* An extranet VPN allows connectivity between sites of a single organization. Its concept is similar to the concept of intranet VPNs except that it requires additional security considerations. If two or more companies decide to work together and allow each access to their networks, security must be taken to ensure that the correct information is easily reached to each company's partner and that the sensitive information is closely guarded from unauthorized users. Firewall and user authentication are important concepts to ensure that only authorized users are allowed to access to the network.

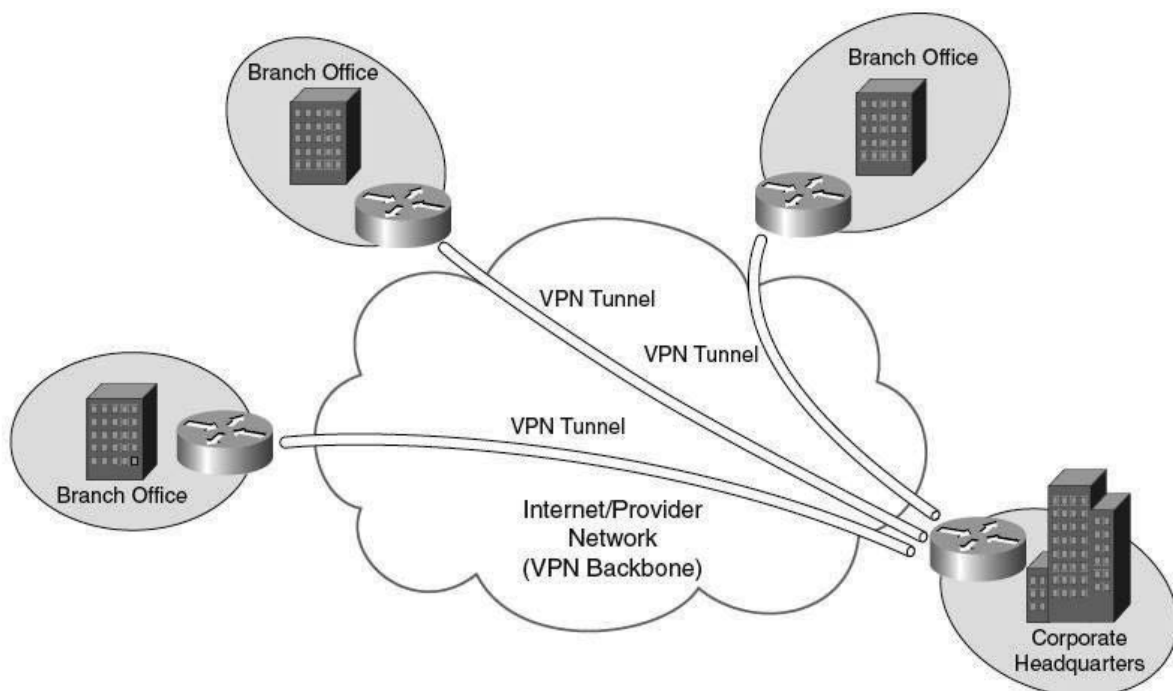


Figure 3-121: Typical Site-to-Site VPNs [CISCO-2005]

3.13.2.2 Remote Access VPNs

Remote access VPNs permit mobile or home-based users to access to an organization's resource remotely. This VPN type allows the users to place calls to a local Point of Presence (POP) and then tunnel the calls through the Internet to avoid long-distance phone charges or bill-backs of toll-free numbers. The calls are then consolidated at the primary locations and fed onto the corporate network. Thus, remote access VPNs have received the most publicity because they can dramatically reduce monthly charges for dial-up or leased line. Typical remote access VPNs are illustrated in figure 3-122. There are two types of remote access VPNs [CISCO-2005]:

- *Client-Initiated*: Remote users use clients to establish a secure tunnel through a shared network to the enterprise.
- *NAS-Initiated*. Remote users dial in to an ISP network access server (NAS). The NAS then establishes a secure tunnel to enterprise private network.

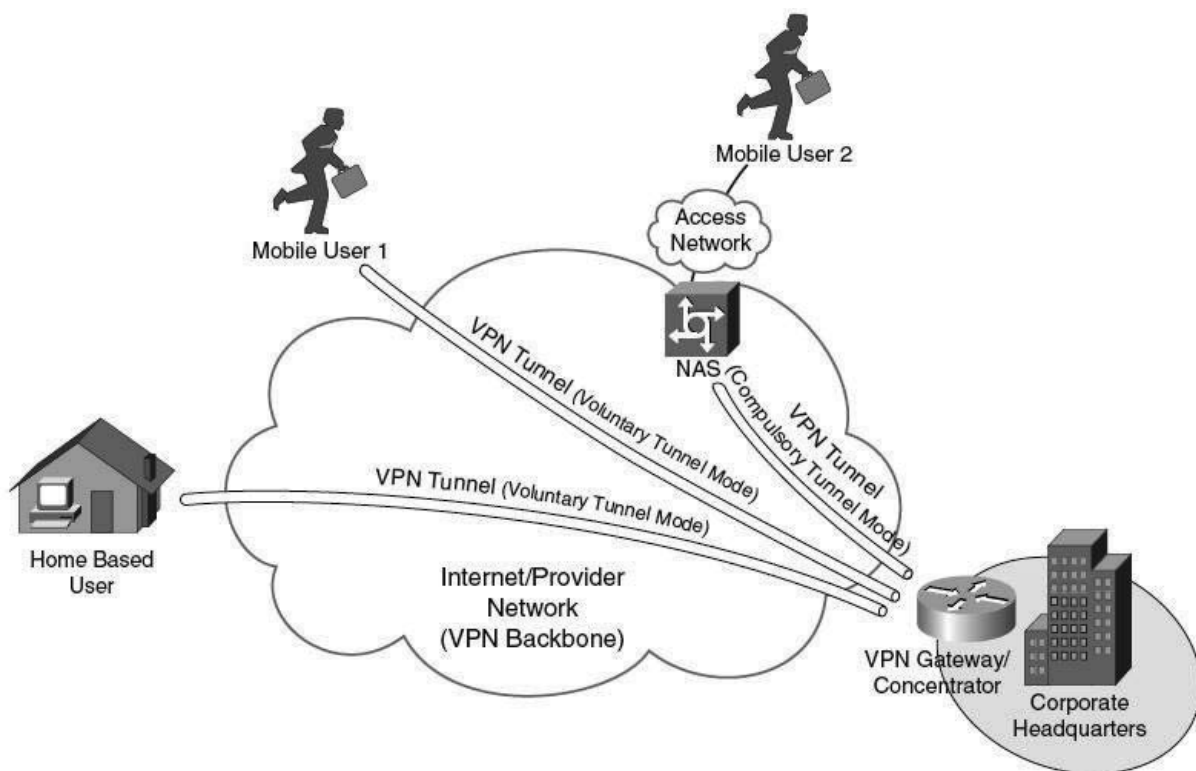


Figure 3-122: Typical Remote Access VPNs [CISCO-2005]

3.13.2.3 Service Provider Provisioned Site-to-Site VPNs

Service Provider Provisioned Site-to-Site VPNs fall into one of two categories: Layer-2 VPN and Layer-3 VPN [AM-2005].

2.13.2.3.1 Layer 2 VPNs

Layer 2 VPNs (L2VPNs) can be provisioned between switches, hosts and routers. These technologies allow data link layer connectivity between separate sites.

The communication between customer switches, hosts and routers is based on the layer 2 addressing. Forwarding of the customer traffic at the Provider Edge (PE) devices is performed based on the layer 2 header information, such as MAC address, data link connection identifier, CoS field.

Solutions for supporting provider-provisioned layer 2 VPNs are defined and specified in the IETF working group l2vpn. Techniques and protocols for enable L2VPN are specified in [KR-2007, LK-2007, AA-2006, AS-2006].

2.13.2.3.2 Layer 3 VPNs (L3VPNs)

Layer 3 site-to-site VPNs (L3VPN) interconnects hosts and routers at separate customer sites. Communications between customer hosts and routers are based on layer 3 addressing. Forwarding customer traffic at PE devices is based on incoming link and on addresses contained in the IP packet header. Solutions for L3VPNs are specified in the IETF working group l3vpn. There are two overall types of L3VPNs solutions:

- *PE-based VPNs*: In a PE-based VPNs, each PE device maintains VPN states, isolating users of one VPN from users of another. PE devices participate in customer network routing, knowing that certain traffic is VPN traffic, and forwarding this traffic by using of the IP destination address and of other information in the IP packet header. The VPN traffic forwarded between PE devices over VPN tunnels may take the form of GRE, IPsec or MPLS tunnels. In this case, CE devices are not aware that they are participating in a VPN. PE-based L3VPNs can be further classified as
 - *BGP/MPLS IP VPNs*: In this VPN type, the PE devices maintain separate routing and forwarding tables for each VPN. BGP/MPLS IP VPNs enables the marking of route advertisements with attributes that identify their VPN context. By this way, multiple forwarding table instances can be maintained while running only single BGP instance.

- *Virtual Router (VR) style*: In this VPN type, completely separate logical routers are maintained on the PE devices for each VPN. Each logical router maintains a unique forwarding table and its own entirely separate routing protocol instances.
- *CE-based VPNs*: in a CE-based L3VPN, PE devices do not participate in customer network routing and forward customer traffic based on globally unique addressing. All the VPN-specific procedures are performed in the CE devices. And tunnels are configured between CE devices using protocols such as GRE or IPsec.

Solutions and standards for L3VPNs are specified in [AM-2005, Mor-2007, RR-2006, CM-2005].

3.13.3 Protocols to Enable VPNs

VPN protocols and technologies can be classified via three different categories:

- *Site-to-site vs. remote access*: VPN protocols can be classified into protocols used for site-to-site VPNs and protocols used for remote access VPNs.
- *Secure vs. Trusted*: VPN protocols can be categorized into protocols used for secure VPNs and protocols used for trusted VPNs.
- *Unicast vs. multicast*: Relating to unicast and multicast communication, VPN protocols can be classified into protocols for supporting multicast VPNs and protocols for supporting unicast VPNs.

In site-to-site VPNs, customer traffic is either tunneled between CE devices or between PE devices. Protocols and technologies used to enable site-to-site VPNs include IP security (IPsec), Generic Routing Encapsulation (GRE), The layer two tunneling protocol (L2TP), the layer two Tunneling Protocol version 3 (L2TPv3), MPLS Label Switched Path (LSP).

- *IPsec* [RFC4301, RFC4309] is a framework of open standards designed to provide data confidentiality, data integrity, and data origin authentication between peers (security gateways or hosts) that are connected over unprotected networks such as Internet. IPsec tunnels are used to build a site-to-site between CE devices.
- *GRE* [RFC2784, RFC2890] can be used to construct tunnels and to transport traffic between CE devices in a VPN.
- *L2TP* [RFC2661] is an IETF standard tunnelling protocol for VPNs. L2TP is designed to tunnel PPP traffic over LANs or public networks
- *L2TPv3* [RFC3931] allows the transport of point-to-point traffic such as frame relay, ATM, Ethernet, HDLC and PPP traffic over IP or other backbone.

- *MPLS layer 3 VPNs (MPLS/BGP VPNs) [RFC4364]*: While BGP is used for distributing the routing and VPN-related information between PE routers, MPLS is used to forward VPN traffic through provider networks.
- *MPLS Layer 2 VPNs [RFC4448, RFC3985]* enable the transport of layer 2 frames over MPLS backbone.

Protocols used to enable remote access VPNs include the following:

- *Layer Two Forwarding (L2F) Protocol*: L2F is developed from Cisco. It enables the creation of Network Access Server (NAS)-initiated tunnels by forwarding Point-to-Point (PPP) sessions from one endpoint to another across a shared network infrastructure.
- *Point-to-Point Tunnelling Protocol (PPTP)*: Like L2TP, PPTP tunnels the layer-2 PPP traffic over LANs or public networks. PPTP creates client-initiated tunnels by encapsulating packets into IP datagrams for transmission over the Internet or over other TCP/IP-based networks.
- *Layer two Tunnelling Protocol version 2 and 3 (L2TPv2/L2TPv3)*: L2TP is an IETF standard and combines the best features of L2F and PPTP. L2TP allows either tunnelling of remote access client PPP frames via a NAS to a VPN gateway/concentrator or tunnelling of PPP frames directly from the remote access client to the VPN gateway/concentrator.
- *IPsec*: IPsec can be used to securely tunnel data traffic between remote access or mobile users and a VPN gateway/concentrator.

Technologies and protocols for support secure VPNs are for example IPsec and L2TP. For trusted VPNs, technologies such as MPLS/BGP and transport of layer 2 frames over MPLS can be used.

Multicast VPNs deal with Technologies and protocols that enable the delivering of multicast traffic between different sites of customer networks. Protocols for multicast VPNs are for example

- *Protocol Independent Multicast (such as PIM-SM, PIM-SSM)*: PIM is used to create the multicast distribution tree.
- *IP tunnelling (such as GRE)*: This method is used for eliminating the customer multicast state at P devices, because the IP tunnels are overlaid across the MPLS/IP network. It also prevents the service provider from having to run any IP multicast protocols in the P devices, because all packets are sent as unicast.
- *Multicast domains (MDs)*: MDs enable CE routers to maintain PIM adjacencies with their local PE routers instead with all remote PE routers. This is the same concept as deployed with layer 3 MPLS VPNs, where only a local routing protocol adjacency is required rather than multiple ones with remote CE routers.

In the next following sections, MPLS VPNs and multicast VPNs will be described.

3.13.4 MPLS VPN

MPLS VPN is a framework of protocols that uses the power of MPLS in supporting of traffic isolation and of service differentiation to create VPNs. As described in the last sessions, MPLS VPN approaches can be classified into MPLS Layer 3 VPNs and MPLS Layer 2 VPNs.

3.13.4.1 MPLS Layer 2 VPNs

The MPLS Layer 2 VPNs provides complete separation between the provider's network and the customer's network, i.e. there is no route exchange between the PE devices and the CE devices. This separation between the provider's network and the customer's network provides simplicity.

The MPLS Layer 2 VPNs approach addresses two connectivity problems: Point-to-Point connectivity, and Multi-Point connectivity.

- *Point-to-Point connectivity.* This approach is described in the RFC 4906 [MRE-2007]. In order to carry layer-2 frames across an MPLS network, the concept of Virtual Circuit (VCs) is introduced. An MPLS LSP operates as a tunnel, which carrying multiple VCs through MPLS backbone. A VC is a LSP within the original tunnel LSP. While the tunnel LSP provides the tunnel between two PE routers, a VC carries frames of a customer only. The tunnel LSPs between PE routers can be created using any protocols e.g. LDP or RSVP extension for traffic engineering (RSVP/TE). PE routers exchange the VC labels via LDP in downstream-unsolicited mode. At the begin of the tunnel, the PE router encapsulates each subscriber layer-2 frame, attaches a VC and a tunnel label, then sends the frame over the tunnel LSP. At the other end of the tunnel LSP, the receiving PE router removes the tunnel label. Based on the VC label, the PE router determines which customer port the packet should be delivered to. It then extracts the original layer-2 frame and sends it out to the port defined above.
- *Multi-Point connectivity.* The goal is to develop solutions that facilitate carrying customer layer-2 frames over the IP/MPLS network from and to multiple sites belonging to a given customer. A popular approach for multi-point connectivity is called Virtual Private LAN Service (VPLS) specified in RFC 4761 and RFC 4767 [LK-2007, KR-2007]. VPLS builds a VPN by creating a full mesh of VCs between the PE routers facing the

sites that make the VPN. In VPLS, exchanging the VC labels between PE routers is performed via LDP. Customer VPNs are identified via a unique 32-bit VPN ID. PE routers perform the source MAC address learning to create layer-2 forwarding table entries. Each entry associates with a MAC address and a VC number. Based on MAC addresses and VC numbers in the forwarding table, PE router can forward the incoming frames.

3.13.4.2 MPLS Layer 3 VPNs

The problem with the Layer 2 VPN technology is that it does not scale well. By growing of the network, the number of required virtual circuits achieving optimal routing scales non-linearly. Moreover, it is also difficult to provide traffic engineering using a Layer 2 VPN approach. To solve these problems, a MPLS Layer 3 VPN standard [RR-2006, CM-2005] called BGP/MPLS IP VPNs is developed, which provides Layer 3 VPN solutions using BGP to carry the VPN information over a MPLS core.

The key of this approach is to use BGP and the so called BGP-VPN extensions to allow separate route forwarding information to be maintained for each VPN client. This separate route forwarding information is carried via BGP over MPLS using the Label Distribution Protocol.

In particular, the BGP/MPLS IP VPNs approach relies on taking customer IP datagrams from a given site, looking up the destination IP address of each datagram, in a forwarding table, then sending this datagram to its destination across the provider's network using an MPLS Label Switched Path (LSP). The fundamental mechanisms of BGP/MPLS IP VPNs can be summarized as follows:

- *Addressing the customer sites.* Each site belonging to a VPN has a number. This number is configured as 8-byte Router Distinguisher (RD), which is used to prefix the IP address for this site. A PE router can learn an customer IP prefix from a CE router through a BGP session with the CE router, or through the RIP exchange with the CE router. After it learns the IP prefix, the PE router converts it into a VPN-IPv4 route by combining it with an 8-byte route distinguisher (RD). The generated prefix is used to uniquely identify the customer site.
- *Distributing the VPN routing information among PE routers via BGP.* PE routers can distribute VPN-IPv4 routes to each other by means of a BGP connection between them. When a PE router distributes a VPN-IPv4 route via BGP, it sets the BGP NEXT_HOP equal to its own address. It also assigns and distributes MPLS labels for this route.
- *Maintaining multiple routing and forwarding tables on each PE router.* To address the problem of overlapping VPNs address space, where one

site could belong to more than one VPN, multiple Virtual Routing and Forwarding (VRF) tables are created on each PE router, in order to separate the routes belonging to different VPNs on a PE router. A VRF table is created for each site connected to a PE router.

- *Forwarding the packets between VPN sites.* Based on the routing information stored in the VRF tables, packets are forwarded to their destination using MPLS. A PE router binds a label to each customer IP prefix learned from a CE router and includes the label in the network reach-ability information that it advertises to other PE routers. When a PE router forwards a packet received from a CE router across the provider network, it attaches two MPLS labels to the packet in order to forward it to its destination. The outer label is for the LSP leading to the BGP NEXT_HOP. It is used to direct the packet to the correct PE router. The inner label is used by the destination PE to direct the packet to the CE router. When the destination PE router receives the labelled packet, it removes the label and uses it to deliver the packet to correct CE router. This MPLS label forwarding across a provider backbone is based either on label switching or traffic engineering paths.

3.13.5 Multicast VPNs

A fundamental problem of service providers when offering native multicast services to end customers is the amount of multicast distribution information ([S, G] or [* , G]) states), which is needed to be maintained at the provider routers. When a multicast source becomes active within a particular customer site, the multicast traffic must be delivered through service provider network to reach all PE routers that have receivers connected to CE routers for that multicast group.

To avoid unnecessary traffic delivery, the standard IP multicast technology enables the service provider to prevent sending traffic to PE routers that do not have receivers. To achieve it, each P router must maintain the multicast state information for all active customer distribution trees. But, the service provider does not know, how the multicast is managed at its end customers within their enterprise. Furthermore, the service provider does not have control over the distribution of multicast sources, receivers and number of multicast groups that are chosen by end customers. Therefore, the P routers must maintain an unbounded amount of multicast state information based on the enterprise customer's applications of multicast.

A common solution, which eliminates the need for any state information to be maintained in the P routers while delivering the multicast over provider IP or

MPLS VPN network, is to use generic routing encapsulation (GRE) tunnels between CE routers. However, the disadvantage of this solution is that if the customer does not implement a full mesh GRE tunnels between CE routers, optimal multicast routing can not be achieved. Moreover, multicast over GRE is not scalable because of the potential number of tunnels required and the amount of operational and management overhead.

A more scalable approach called multicast VPN (MVPN) is to provide multicast within a VPN is achieved in a Layer 3 MPLS VPN. The reasons for developing multicast VPNs in an MPLS VPN are:

- In a MPLS VPN, a P router maintains routing information and labels for global routing table only. It does not hold routing or state information for customer VPNs.
- In a MPLS VPN, a CE router maintains a routing adjacency with its PE router neighbours only. CE routers do not peer with other CE routers but still have the ability to access to other CE routers in their VPNs through optimal routes provided by P routers.

MVPN introduces the concept of multicast domain, in which CE routers maintain PIM adjacencies with their local PE routers instead of with all remote CE routers. Thus CE routers do not have multicast peering with other CE routers, but they can exchange multicast information with other CE routers in the same VPN. In this approach, a P router does not maintain multicast state entries for customer VPNs but instead it maintains multicast state entries for global routing table only, regardless of the number of multicast groups deployed by the end customers. The section gives a short summary of the MVPN approach specified in [Ros-2007].

MVPN consists of several components. The key components of MVPN include:

- *Multicast domain (MD)*. MD consists of a set of VRFs used for forwarding multicast traffic to each other. The multicast domain allows the mapping of all customer's multicast groups that exist in a particular VPN to a single unique multicast group in the provider network. This is achieved by encapsulating the original customer multicast packets within a provider packet by using GRE. The destination IP address of the GRE packet is the unique multicast group that the service provider has allocated for that multicast domain. The source address of a GRE packet is the BGP peering address of the originating PE router.
- *Multicast VRF (MVRF)*. MVRF is a VRF that supports both unicast and multicast routing and forwarding tables.
- *Multicast Distribution Tree (MDT)*. MDT is used to carry customer multicast traffic between PE routers in a common MVPN. It takes the

form of multicast tree in the core network. An MDT is sourced by a PE router and has a multicast destination address. PE routers, which have customer sites for the same MVPN, will source to a default-MDT and join to receive the multicast traffic. In order to save bandwidth used for multicast traffic and to guarantee the QoS for the multicast applications, two additional sub-components are defined: the Default-MDT and the Data-MDT (Figure 3-123).

- o Default-MDT is enabled per customer VRF on every PE router that will forward multicast packets between customer sites. The Default-MDT is created to delivery PIM control traffic, and to flood the multicast channels for low-bandwidth groups. Hence, the Default-MDT is always present.
- o A Data-MDT is only created for higher bandwidth multicast source. It can be created on PE routers per VRF. Only routers, which are part of the multicast tree for the given high bandwidth source, receive the multicast packets generated by this source. Thus the Data-MDT is created only on demand of high-bandwidth sources for each pair of (S, G) MVPN.

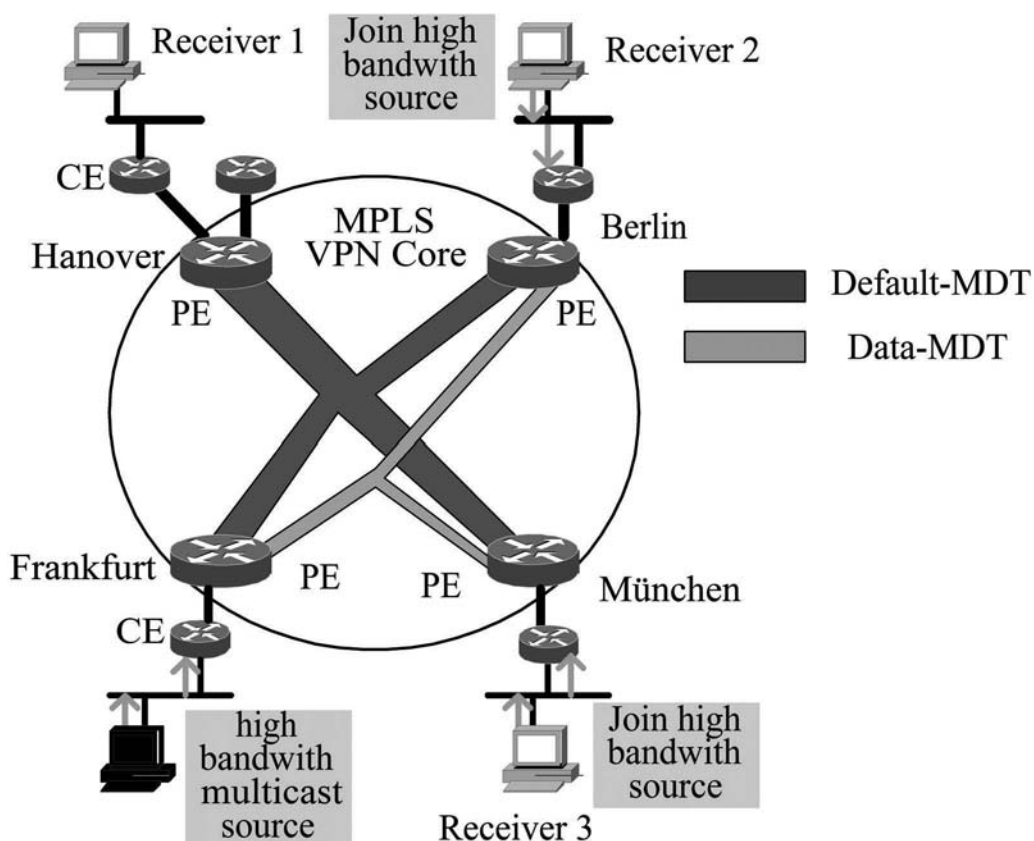


Figure 3-123: Multicast VPN concept

In order to support both default-MDT and data-MDT, every PE router has one or more multicast routing table and has at least one default table for the provider network. Additionally, a multicast routing table exists for each VPN to which the PE is attached.

In order to provide MVPN, following mechanisms are needed [Ros-2007]:

- *Discovering MVPN control information.* Like the layer 3 MPLS VPNs, MVPN control information is discovered via BGP.
- *Creating and maintaining multicast VRF tables.* Multicast VRF tables are created and maintained via multicast routing protocols such as PIM-SSM or PIM-SM. Multicast VRF tables are the PE router's view into the enterprise VPN multicast. Multicast VRF tables contain all the multicast routing information for each VPN. This information includes the state entries for MDT or RP (if PIM-SM is being used)
- *Building default-MDT and Data-MDT (PIM-SSM):* MDTs are created on the basis of the multicast routing protocols and the multicast VRF tables.
- *Forwarding multicast traffic.* When a PE router receives a MDT packet from a CE router interface, it performs an Reverse-Path Forwarding (RPF) check. During the transmission of the packet through the provider network, RPF rules are applied for duplication check. When the customer's packet arrives at the destination PE router, this PE router needs to ensure that the originating PE router was the correct one for that CE router. The PE router performs it by checking the BGP next hop address in the customer's packet source address. This next hop address should be the source address of the MDT packet. Moreover the destination PE router also checks that there is a PIM neighbour-ship with the remote PE router.

3.14 Summary

The Internet is increasingly used for multimedia and wireless applications, which require services that should be better than the best-effort service provided by the traditional IP-based network. Consequently, new techniques have been added into the Internet to offer new services and to provision the QoS for multimedia and wireless applications. Thus, not only techniques for data communications, but also techniques for multimedia and wireless communications must be taken into consideration. For this reason, we first provided a rather self-contained survey of techniques covering mechanisms, protocols, services and architectures to control the traffic and to ensure the QoS for data and multimedia applications. Important to note is that most of these techniques can be implemented in various protocols and in several layers of computer networks.

Communication errors may arise in all layers of computer networks. For a reliable data transmission, mechanisms for discovering and correcting such errors are necessary needed. Thus, we started with the mechanisms for detecting and correcting the bit-level and packet-level errors - basic mechanisms implemented in various protocols of the TCP/IP suite.

In a shared medium with multiple nodes, when multiple nodes send messages at the same time into the shared medium, transmitted messages may collide at all receivers. Thus, all messages involving in a collision are lost. To avoid this problem, multiple access control is needed. Its job is to shared a single broadcast medium among competition users.

As the Internet becomes increasingly heterogeneous, the issue of congestion control becomes more important. A way for avoiding the network congestion is to filter the source traffic flows at entry nodes and at specific nodes. Once a connection is accepted, its emitting traffic to the network should confirm the traffic descriptor. Otherwise, the excess traffic can be dropped, marked with a lower priority, or delayed. This is performed via the traffic access control including traffic description, traffic classification, policing, shaping, marking and metering. In order to provide the delay guarantee and bandwidth assurance for data and multimedia applications, scheduling disciplines together with their advantages and disadvantages are described.

Congestion in Internet directly refers to packet loss and thus affects the QoS of data and multimedia applications. To manage and control the congestion, mechanisms for congestion control at the end hosts and at the routers for unicast and multicast applications are addressed. Also, congestion can arise because of the failure or bottleneck of the selected route, to which the packets should be delivered to reach its final destination. Determination of such routes is done by routing - an important component keeping the Internet to operate. In order to transfer data and multimedia traffic over Internet, mechanisms and protocols for unicast routing, multicast routing and for QoS routing are then investigated.

As the IP technology becomes more and more basis of the Next Generation Network (NGN), QoS is required to support real-time multimedia applications. To guarantee such QoS at a smaller time scale, admission control and signalling are used. Therefore, admission control mechanisms developed to enable network devices for deciding the permission of a connection are addressed. Following it, signalling mechanisms (Resource Reservation Protocol, Next Step in Internet Signalling and signalling for voice over IP) allowing the devices to exchange the control information are described in detail.

To provide the end-to-end QoS at the Internet layer, IntServ (Integrated Services), DiffServ (Differentiated Services) and MPLS (Multi-Protocol Label Switching) are described. IntServ was the first architecture that support per-flow

QoS guarantees, requiring relatively complex packet classify, admission control, per-flow and per-router signalling within any router belonging to the end-to-end data transmission path. In contrast to IntServ, DiffServ handles packets on the per-class basis that allows the aggregation of several flows to one class, and does not need the per-router signalling as in IntServ. In comparison with IntServ and DiffServ, MPLS additionally support the traffic engineering that allows explicitly non-shortest path routing to be used.

Mobile networking is one of the important drivers for the multi-service networks. To support mobility at the Internet layer, architectures, protocols and mechanisms for providing mobility in IPv4 (MIP4) and in IPv6 (MIP6) are expressed. The IP mobility problem is solved via the introduction of a fixed home address used by the correspondence node (CN) and a dynamic care-of-address used by the mobile node (MN). Relaying packets between CN and MN in MIP4 is done via intercepting the packets by Home Agent (HA) and via tunnelling the intercepted packets from HA to Foreign Agent (FA). By MIP6, no FA is needed. Packets are tunnelled by HA directly to the MN.

In order to provide QoS for multimedia applications at the application and at the transport layer, new transport protocols are needed. Therefore, concepts and mechanisms of RTP, SCTP and DCCP are explained. RTP (Real Time Protocol) operates on top of UDP. Unlike UDP, RTP provides sequence numbers, time stamps and QoS parameters to the applications. These parameters will enable the application developers to add mechanisms to ensure timely delivery, to promise the reliable delivery of packets or to prevent their out-of-order delivery, to provide QoS guarantee, to control and avoid the congestion. SCTP (Stream Control Transmission Protocol) has been developed because of the TCP limitation for supporting the transport of PSTN signalling across the Internet. It is a reliable transport protocol operating on top of IP. In comparison to TCP, it provides a set of new mechanisms, such as message-oriented data transfer, association phases, user data fragmentation, path management, multi-homing, and multi-streaming. These mechanisms are particularly desirable for telephone signalling and multimedia applications. DCCP (Datagram Congestion Control Protocol) is a newly specified transport protocol existing at an equivalent level with UDP, TCP and SCTP. A special feature of DCCP is that it provides an unreliable end-to-end data transmission service for unicast datagrams, but a reliable end-to-end acknowledgement transmission between the sender and the receiver. Like SCTP, it also offers a reliable handshake for connection establishment and teardown and a reliable negotiation of features. The biggest difference of DCCP to TCP and SCTP is that DCCP enables applications to choice of modular congestion control mechanisms, which can be either TCP-like congestion control or TCP-Friendly rate control.

Based on the protocols described above, standard architectures for VoIP and for IPTV are then given in detail. These architectures are used to deliver voice and television traffic over Internet with QoS guarantees.

Another effective way to securely transfer of user data, generated from different customer sites with performance provision and QoS guarantees, is the use of VPNs. Thus, we started with a depth overview of VPN concepts and architectures including layer-2 and layer-3 VPNs. We then gave a summary of the protocols used to enable site-to-site VPNs, remote access VPNs and multicast VPNs. As a basic for our developed algorithms, mechanisms for MPLS VPNs and multicast VPNs are given.

We have investigated protocols and architectures (shown in table 3.3) for traffic management and QoS control. This table represents the diversity of the existing techniques that are developed in several layers, for different applications and for varying communication forms. These techniques directly or indirectly influence the network performance and the QoS of the applications.

Traffic management and QoS control		Archiving QoS					Layer (1-5)	Unicast/Multicast/ Broadcast	Elastic/Stream Ap- plications
		Delay	Jitter	Throughput	Loss rate	Reliability			
Error control	Bit level				x		2-5	U/M	E/S
	Packet level				x		2-5	U/M	E/S
Multiple Access Control	FDMA, TDMA							B	
	ALOHA, Slotted ALOHA	x			x		2	B	
	CSMA	x			x		2	B	
	CSMA/CD	x			x		2	B	
	CSMA/CA	x			x		2	B	
Traffic Access Control	Description						2-5	U	E/S
	Classification						2-5	U	E/S
	Policing	x	x	x	x		3	U	E/S
	Shaping	x	x	x	x		3	U	E/S
	Marking	x	x	x	x		2-4	U	E/S
	Metering	x	x	x	x		3	U	E/S
Scheduling	FIFO						3*	U	E/S
	PS	x	x	x	x		3*	U/M	E/S
	RR	x	x	x	x		3*	U/M	E/S
	WRR	x	x	x	x		3*	U/M	E/S
	DRR	x	x	x	x		3*	U/M	E/S
	WFQ	x	x	x	x		3*	U/M	E/S

* can be implemented in another layers

Active Queue Management	Drop From Tail				x		3	U/M	E
	RED				x		3	U/M	E
	WRED				x		3	U/M	E/S
Congestion Control (CC)	TCP congestion control				x		4	U	E
	ECN				x		3-4	U	E
	TFRC			x	x		4-5	U	E
	TLCC			x	x		4-5	U	E
Routing	RIP					x	5	U	S
	OSPF					x	3	U	S
	BGP					x	5	U	E/S
	QoS routing	x			x	x	3	U	E/S
	DVMRP					x	3	M	S
	MOSPF					x	3	M	S
	PIM (SM, DM, SSM)					x	3	M	S
	IGMP					x	3	M	E/S
Signalling	RSVP	x	x	x	x	x	4	U/M	E/S
	NSIS	x	x	x	x	x	3-5	U	E/S
	SIP					x	5	U	S
	H.323					x	5	U	S
Admission Control	PBAC			x			4	U	E/S
	MBAC			x			3	U	E/S
	EBAC			x			3	U	E/S
	PbAC			x			3	U	E/S
QoS Architectures	IntServ	x	x	x	x	x	3	U/M	E/S
	DiffServ	x	x	x	x	x	3	U	E/S
	MPLS	x	x	x	x	x	2-3	U/M	E/S
Internet Protocol	IPv4						3	U/M	E/S
	IPv6						3	U/M	E/S
Mobile IP	Mobile IPv4						3	U/M	E/S
	Mobile IPv6						3	U/M	E/S
Audio and Video Transport	TCP				x		4	U	E
	UDP						4	U/M	E/S
	RTP/RTCP						5	U/M	S
	SCTP				x		4	U	E/S
	DCCP				x		4	U	S
	VoIP						3-5	U	S
	IPTV				x		3-5	M	S
Virtual Private Network	MPLS VPNs					x	2-5	U/M	E/S
	Multicast VPNs					x	3	M	E/S

Table 3.3: The investigated protocols and architectures

4. Internet Protocol Suite

So far we had presented the mechanisms for traffic management and QoS control in telecommunication networks and in the Internet without considering of the layered network architecture of the communication systems. From our point of view, each of these mechanisms may be used in different protocols, in the several layers and in different communication systems. Based on the basic of these mechanisms, in this chapter, an architecture overview of the Internet and several selected protocols in each layer of TCP/IP protocol suite will be illustrated. The main goal is to introduce you how you can design and develop protocols on the basic of existing mechanisms described in chapter 3.

4.1 Introduction

The Internet protocol stack is specified in five layers of the TCP/IP reference model – the physical, data link, network, transport and application. Each layer can be implemented in hardware or in software that cover its own protocols that solve a set of problems involving the data transmission and providing services to the upper layer protocol instance.

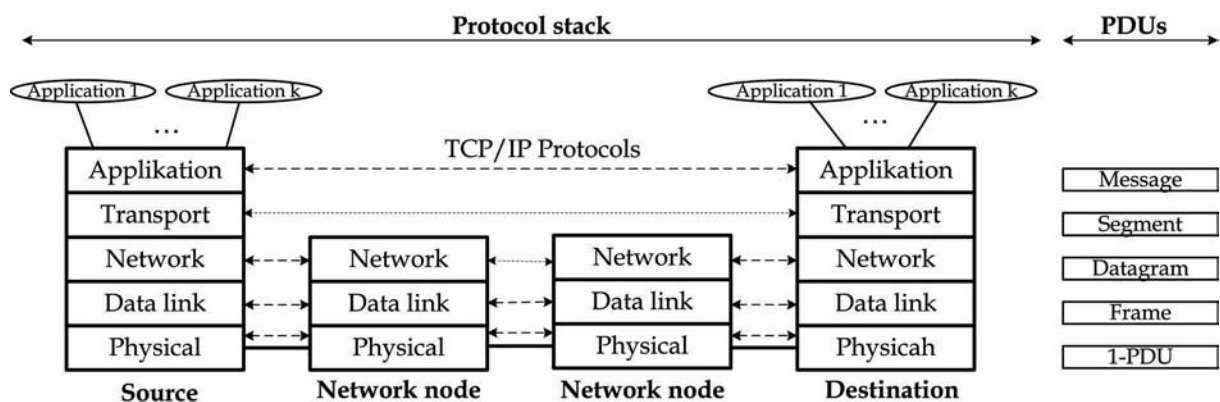


Figure 4-1: the Internet protocol stack and the Protocol data Unit

Instead of using the terminology n-PDU of the OSI reference model, special names for PDUs in the Internet protocol stack are determined: message, segment, datagram, frame and 1-PDU. Each PDU has two parts: header and payload. The header contains the information used for treating the PDU at this layer. The payload holds the user data and the header of the upper layers. The Internet protocols stack and the PDU names are illustrated in figure 4-1.

Figure 4-2 shows an example how PDUs can be transmitted using the Internet protocol stack. The sending process of the application A at the source

host needs to send data to the receiving process of the application A at the destination host. The sending process first passes the data to the application layer instance, which adds the application header (AH) to the front of data and gives the resulting message to the transport layer protocol instance. The transport layer instance attaches the transport header (TH) to the front of the message and passes it to the network layer. The network layer instance adds the network header to the front of the segment arrived from the transport and sends it to the data link layer. This process is repeated until the data reaches the physical layer. At the destination, each layer removes its own header and passes the payload to the upper layer until it reach the receiving process at the application layer.

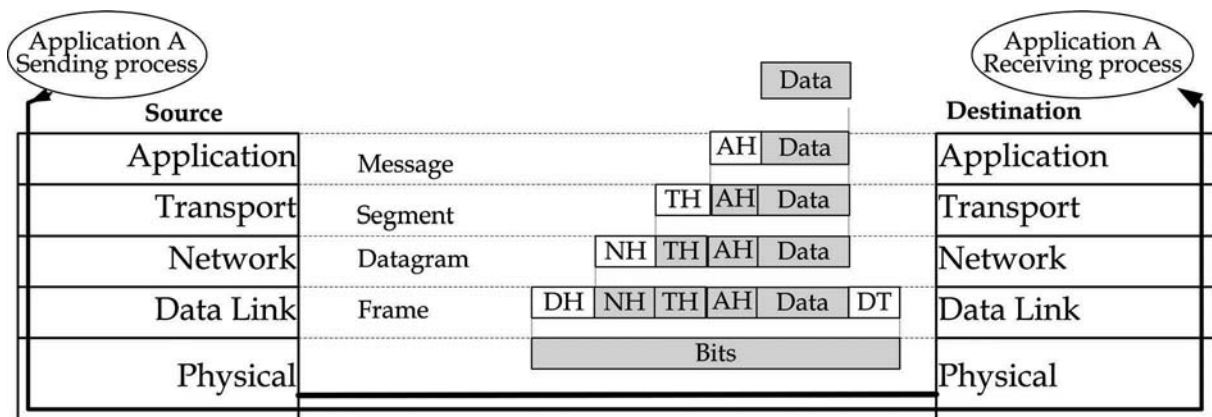


Figure 4-2: Transmitting the PDUs within the Internet protocol stack

4.2 Physical Layer

The physical layer is the lowest layer in the OSI reference model and in the TCP/IP reference model. It involves the basic hardware transmission technologies of a communication network. Within the OSI reference model, the physical layer receives raw bit streams arriving from the data link layer, translates them into hardware-specific operations, and attempt to deliver them over a hardware transmission medium. In this transmission process, bit streams are not guaranteed to be error free. This means, the number of bits received may be less than, equal to or more than the number of bits transmitted. These errors should be detected and if necessary corrected up to the data link layer. The major mechanisms performed by the physical layer are:

- *Characterization of hardware specifications:* Details of operations on cables, connectors, wireless radio transceivers, network interface cards and other hardware devices are generally mechanisms of the physical layer.

- *Signaling encoding*: the physical layer is responsible for transforming the data from bits that reside within a computer or other device into signals that can be sent over the network. The well-known signaling encoding mechanisms are Non-return to zero, Non-return to zero inverted (NRZI) and Manchester encoding. In Non-return-to-Zero, a logic-1 bit is sent as a high value and a logic-0 bit is sent as a low value. NRZI makes a transition from the current signal to encode a 1, and stays at the current signal to encode a 0. In a Manchester encoding, a logic-1 bit is sent 1 to 0 and a logic-0 bit is sent 0 to 1. These mechanisms are detailed illustrated in [Tan-2002].
- *Data transmission and Reception*: after converting the data from bits into signals, the physical layer instance sends these signals to the destination. At the receiving site, the physical layer instance receives the signals. Both are done across a communication circuits (e.g. cable).

The protocol design issues of this layer is to make sure that when one side sends a 1 bit, it is received by the other side as a 1 bit, and not as a 0 bit.

4.3 Data Link Layer

The data link layer involves protocols, which are responsible to transfer a frame across an individual link between two direct connection nodes (e.g. hosts or routers) (see figure 4.3).

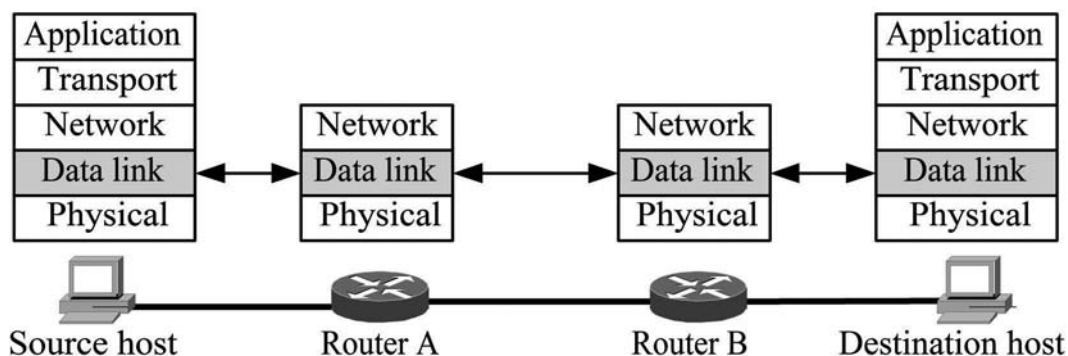


Figure 4.3: The data link layer

A data link layer protocol defines the format of PDUs exchanged between the nodes at the end of a link, and the rules for nodes when sending and receiving the PDUs. On one hand, the data link layer instance receives the datagrams coming from the network layer, encapsulating them into frame and finally delivering them to the physical layer instance. On the other hand, the data link layer receives the frame from the physical layer, decapsulating them and sending them to the network layer protocol instance.

4.3.1 Data Link Layer Services

According to the ISO reference model, the design issue of the data link layer is to provide following possible services to the network layer.

- Addressing
- Reliable delivery
- Framing
- Error control
- Congestion control
- Multiple access control
- Point-to-point line communication
- Authentication
- Network layer protocol support

4.3.1.1 Addressing

The data link layer of sending station send frames to directly connected data link layer entries. In order to enable a LAN station to know whether the arriving frame is to this entry, the layer-2 entries (LAN stations) must be able to address each frame when sending it. The address in the frame header is called MAC address.

4.3.1.2 Reliable delivery

Reliable delivery service guarantees to deliver each network layer datagram across the link without error. This service is achieved with acknowledgement and retransmission mechanisms (see session 3.1) performed at the data link layer where the errors occur. The goal is to correct the error locally, rather than forcing an end-to-end retransmission of data by a transport or application layer protocol. However, reliable delivery at the data link layer can result an unnecessary overhead for low bit error links. For this reason, many link layer protocols do not support a reliable delivery service.

4.3.1.3 Framing

All link layer protocols encapsulate each network layer datagram into a data link layer frame before sending it into a link. This mechanism is called framing in which the streams of bits to be sent “on-the-wire” are spitted into units encapsulated by data link frames (figure 4-4). The basic idea of the framing mechanisms at the data link layer of the sender is to break the bit streams up to discrete frames and to computing the internet checksum for each frame. At the data link layer of the receiver, the checksum for each arriving frame is

recomputed. If the newly computed checksum is different from one contained in the frame, bit error is occurred and the data link layer protocols taken step to do with it.

The problem to be solved within each framing mechanisms is to break the bit stream up into frames so that the receiver can recognize begin and end of each frame. Popular methods are character count, character stuffing, bit stuffing and physical layer coding violations [Tan-2002].

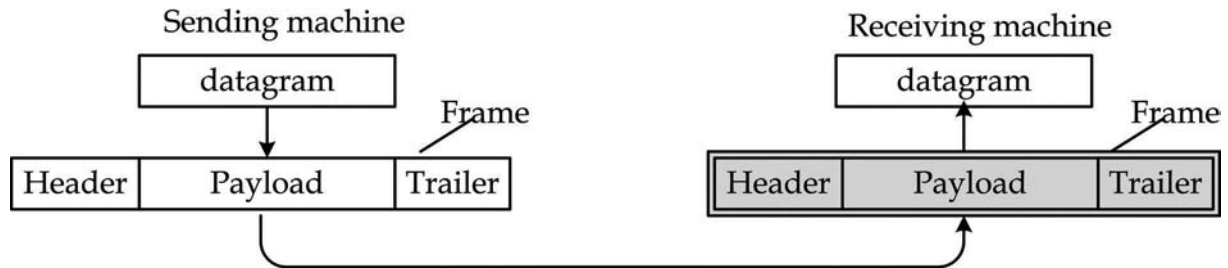


Figure 4-4: Framing example

The first method uses a field in the frame header to specify the number of the characters in the frame. When the data link layer at the receiver sees the character count, it knows how many characters follow and thus it recognize the end of the frame. The drawback of this method is that the counter can be corrupted by a transmission error. For example, instead of 5, the count becomes a 6. Therefore the receiver will get out of synchronization and will be unable to locate the start of the next frame (figure 4-5).

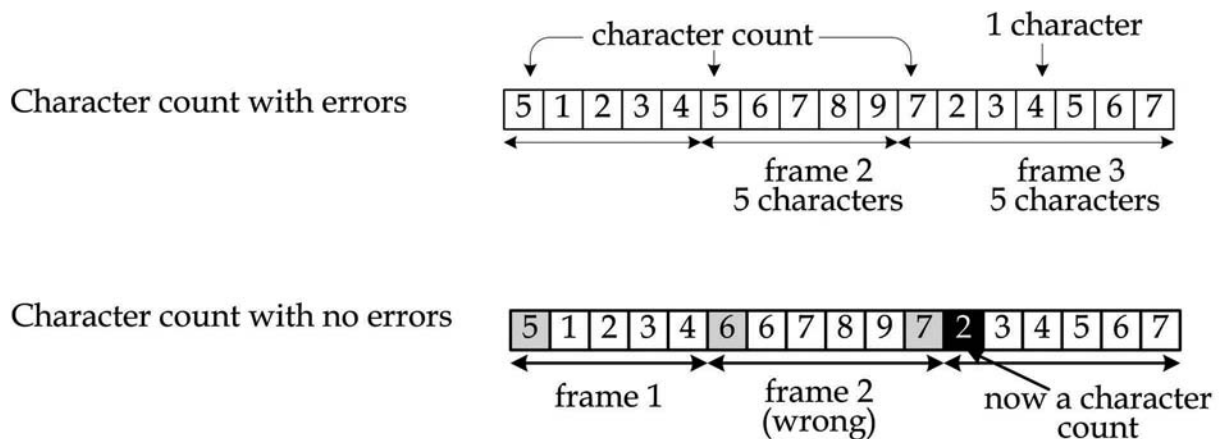


Figure 4-5: Character count

The bit stuffing method allows to have each frame begins and ends with a special bit pattern, 01111110, which is called flag byte. Each time the sender's data link layer sees five consecutive one in the data, it automatically adds a 0 bit into the outgoing bit stream. When the receiver sees five consecutive incoming 1 bits, following by a 0 bit, it automatically removes the 0 bit.

Instead of using the character codes to enable the receiver to recognize the begin and the end of each frame, the physical layer coding violations method uses invalid signal elements to indicate the frame begin and end. Example for this method is the Manchester encoding.

In order to eliminate the problem of synchronization after an error of the character count method, the character stuffing method starts each frame with ASCII frame sequence DLE STX and ending with the sequence DLE DTX. By this way, if the receiver loses track of the frame boundaries, it has only to look for DLE STX or DLE ETX characters to find out where it is. A problem of this method is that the character DLE STX or DLE ETX occur in the data text, which will interfere with the framing. One way to solve this problem is to allow sender's data link layer to insert an ASCII DLE character just before each DLE character in the data. The data link layer at the receiver removes this DLE before it passes the data to the network layer.

4.3.1.4 Error Control

As described in the previous session, the physical layer only accepts raw bit streams and attempt to deliver them over a hardware transmission medium. The bit streams are not guaranteed to be error free. The number of bits received may be less than, equal to or more than the number of bits transmitted. These errors should be detected and if necessary corrected at the data link layer by using of the bit-level error control described in session 3.1.1.

4.3.1.5 Congestion control

Each of the sending and receiving station of a physical link only has a limited amount of buffer capacity. Thus, congestion can be introduced when the sending rate of the sender node is faster than the receiver station can process. Therefore congestion control is needed at the data link layer for preventing the sending station on one side of a link from the overwhelming the receiving station on the other side of the link. The most well-known mechanism used for data link layer congestion control is the window-based mechanism described in session 3.5.

4.3.1.6 Point-to-point line communications

The data link layer protocol in the Internet supports the point-to-point line communications between routers over leased lines and the dial-up to a host via a modem. Two protocols supporting point-to-point line communications are SLIP and PPP, which will be discussed in this section.

4.3.1.7 Multiple access control

The data link layer does not only provide services for point-to-point link consisting of a single sender and a single receiver. It also offers services for a broadcast link having multiple sending and receiving stations connected to the same shared broadcast channel. In a broadcast channel, when a station transmits a frame, the frame is broadcasted on the channel and each of other stations receives a copy of the frame. As a result, the transmitted messages collide at all receivers. Thus, all messages involved in collision are lost. Solution for this problem is the Multiple access control (MAC) illustrated in session 3.2. MAC is used for sharing a single broadcast medium among competition users to eliminate the collision.

4.3.1.8 Authentication

Authentication is the process of determining whether someone or something is, in fact, who or what it is declared to be. In private and public computer networks, authentication allows the sites to exchange authentication messages to authenticate each other before a connection is established. Not all network layer protocols support authentication services. For example, PPP supports authentication, but Ethernet and SLIP do not provide any authentication.

4.3.1.9 Network Layer Protocol Support

A given network device may support multiple network-layer protocols, and use different protocols for different applications. For this reason, a network layer protocol instance needs to know to which network layer protocol (e.g. IP, Novell IPX or AppleTalk) it should pass the payload of a data link layer frame. In order to support multiple network layer protocol, the data link layer protocol can use a field in his header to describe the number of a network layer protocol.

4.3.2 Data Link Layer Protocol Examples

As described above that the data link layer provides services for both point-to-point and for broadcast communication. For a point-to-point link the HDLC (High-level Data Link Control) [Tan-1996], PPP (Point-to-Point Protocol) [RFC 1661, RFC 1662, RFC 1663, RFC 2153] and the SLIP (Serial Line IP) [RFC 1055, RFC 1144] are used. These point-to-point protocols are commonly used over direct-wired connections, such as serial cables, telephone lines, or high-speed dialup such as DSL. For a broadcast channel, multiple access

controls (e.g. Ethernet, Token-Ring, Token-Bus, CSMA/CD) are taken. In this session some examples of these protocols will be illustrated.

4.3.2.1 Serial Line IP (SLIP)

SLIP [RFC 1055, RFC 1144] is the first standard protocol used to dial up connection between residential host and ISP. It is designed to work over serial ports and modem connections – the so called point-to-point links. The SLIP sending station sets begin and end of the frame with two special characters ESC and END respectively. To enable the receiving station to recognize the begin and the end of frames, the SLIP sender applies the character stuffing on this frame before sending the frame over the line to the SLIP receiving station.

SLIP is extremely simple. But it does not provide a lot of mechanisms such as dynamic address negotiation, packet type identification, error control and compression mechanisms.

4.3.2.2 Point-to-Point Protocol (PPP)

The PPP [RFC 1661] solved all problems of the SLIP and is an official Internet standard [RFC 1661]. PPP is designed to support router-to-router and host-to-network connections over synchronous and asynchronous circuit, which can be either dialup or leased line. PPP provides framing and error detection, supporting multiple network layer protocols, enabling the IP addresses to be negotiated at connection time and allowing authentications. The protocol mechanisms of the PPP are as follows

- *Framing*. Encapsulating a multi network-layer datagram in a PPP frame, identifying the beginning and the end of the frame and detecting the error in the frames managing.
- *Multiple network layer protocols*. PPP can negotiate link options dynamically and can support multiple layer 3 protocols (IP, IPX, Apple Talk,..). PPP accomplishes these two tasks by encapsulating layer 3 datagrams with a specialized frame – the PPP frame.
- *Link Control Protocol (LCP)*. PPP defines the LCP for establishing, configuring and testing the data link connection. There exist three classes of LCP packets: link configuration packets used for establishing and configuring a data link connection; link termination packets used for termination a data link connection; link maintenance packets used for managing und debugging a data link connection.
- *Network Control Protocol (NCP)*. Establishing and configuring different network-layer protocols.

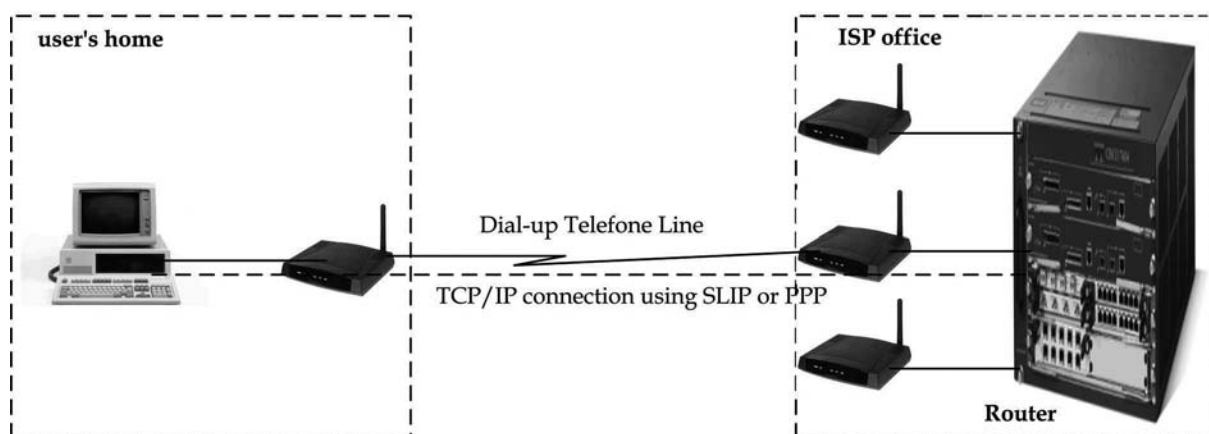


Figure 4-6: Connecting the home PC to the ISP

The basic principle of PPP for a host-to-network connection can be explained in figure 4-6 in the following steps:

1. When a host negotiates a PPP connection with a router at the ISP side, host and router exchange LCP packets. These packets allow data link layer partners to dynamically negotiate link options, including authentication, compression and multi layer protocol. The data link layer on the PC and on the router exchanges control packets in order to select the best PPP parameters. When step 1 is done, the Network Control Protocol (NCP) on both sides takes over. The data link layer partners exchange NCP packets to establish and to configure different network-layer protocols including IP, IPX and AppleTalk. If the host wants to use the IP, the router gives the home PC a temporary IP address. The NCP can build up and tear down multiple layer 3 protocol sessions over a single data link.
2. When a host requests that the connection be terminated, the NCP tears down the layer 3 sessions and then the LCP tears down the data link connection.

The PPP supports a lot of mechanisms but it does not provide the error correction and recovery, the congestion control and the sequencing. Frames with these errors will be handled in the upper layer protocols.

4.3.2.3 Ethernet

Ethernet is a technology for local area network products defined by the IEEE 802.3 standard. An Ethernet can run over coaxial cable, twisted-pair copper wire or fiber optics. Regarding to the OSI reference model, the Ethernet provides services to the network layer. These services are connectionless, unreliable, addressing, encoding, synchronization and framing, multiple access control, and frame check sum for bit-error detection. These services are summarized in the followings.

4.3.2.3.1 *Connectionless*

When an adapter receives an IP datagram from the network layer protocol, the adapter encapsulates the datagram into an Ethernet frame and sends the frame into the LAN if it senses no collision. The Sending adapter does not need any connection set-up with the receiving adapter.

4.3.2.3.2 *Unreliable*

Also the Ethernet provides an unreliable service to the network layer. When adapter B receives a frame from A, adapter B does not send an acknowledgement when a frame passes the bit error check, nor it does send a NACK when a frame fails the bit error check. Adapter A doesn't know whether its transmitted frame was received correctly or incorrectly. And if a frame fails the bit-error check, the adapter B simply discard the frame.

4.3.2.3.3 *Addressing*

The MAC addresses (Source and destination) added into each Ethernet frame are used to deliver this frame to reach its destination adapter. When an adapter is manufactured, a MAC address is burned into the adapter's ROM. No two adapters have the same MAC address.

When an adapter wants to send a frame to same adapter on the same LAN, the sending adapter inserts the destination's MAC address into frame. It also inserts its MAC address into source MAC address and sends the frame over a broadcast channel. When this frame arrives to a LAN station, the station verifies the Ethernet header of this frame. If the destination address of the frame matches its MAC address, then the station copies the frame. It then extracts the IP packet from the Ethernet frame and passes the IP packet up to the IP instance of the IP layer. If the destination address does not match its MAC address, the station ignores the frame

4.3.2.3.4 Encoding

The Ethernet protocol uses the Manchester encoding described in [TAN-2006].

4.3.2.3.5 Synchronization and Framing

8 bytes preamble is used by the Ethernet.

4.3.2.3.6 Multiple Access Control

Ethernet uses 1-persistent CSMA/CD as the multiple access control described in session 3.2 as its MAC protocol.

4.3.2.3.7 Frame check sum for bit-error

Ethernet provides bit-error detection mechanism based on CRC, but it does not provide the bit error correction. The Ethernet frame format is shown in figure 4.7.

Preamble (8 bytes)	Destination Address (6 bytes)	Source Address (6 bytes)	Type (2 bytes)	Data (46-1500 bytes)	FCS (4 bytes)

Figure 4-7: Ethernet header format

- *Preamble* (8 bytes). The 8-bytes preamble field is used for synchronizing the receiver. The preamble is build up by first 7 bytes of “10101010” and the last byte with “10101011” is used as bit stuffing.
- *Source Address* (6 bytes). This field carries the MAC address of the adapter that sends the frame into the LAN.
- *Destination Address* (6 bytes). This field carries the MAC address of destination adapter.
- *Type* (2 bytes). The value of this field indicates the network layer protocol.
- *Frame Check Sum* (FCS,): this field is used for detecting the bit error. The frame check sum is computed by using of the cyclic redundancy check (CRC).

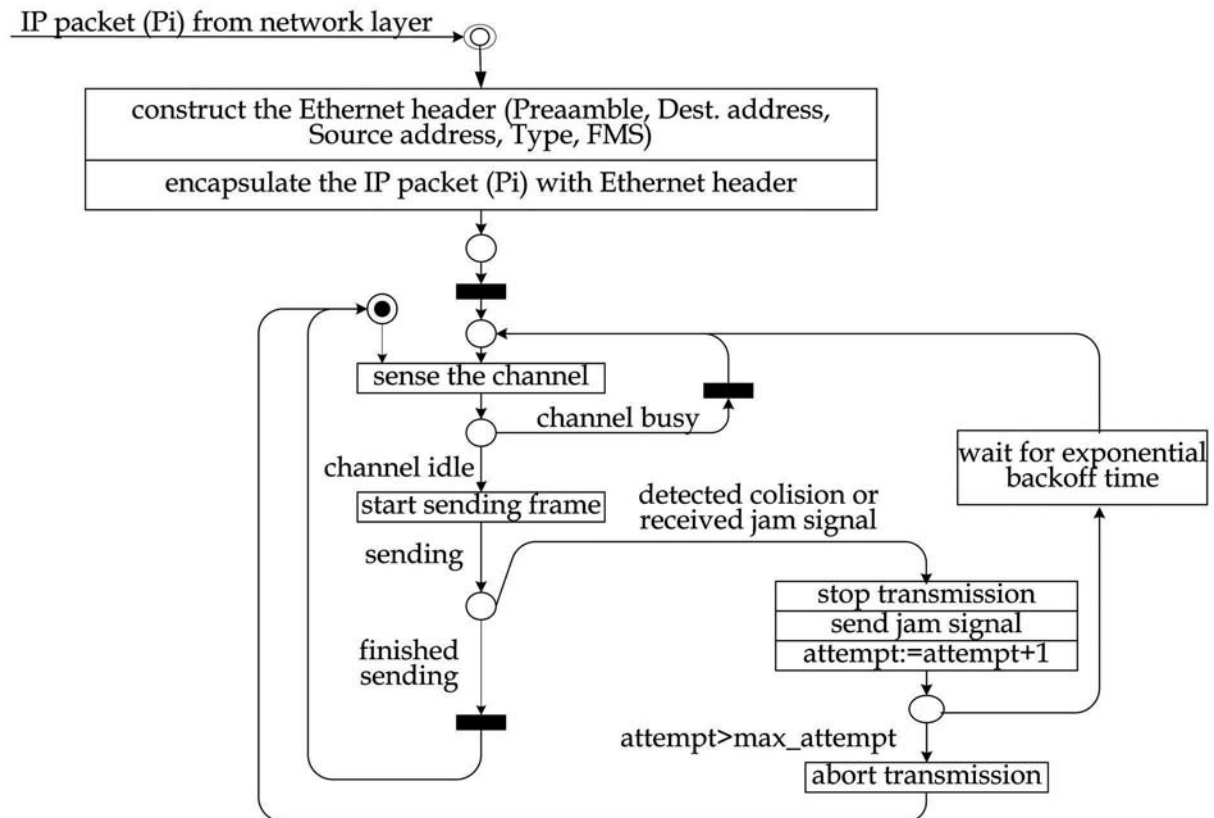


Figure 4-8 A Modelling the Ethernet protocol at the sending side

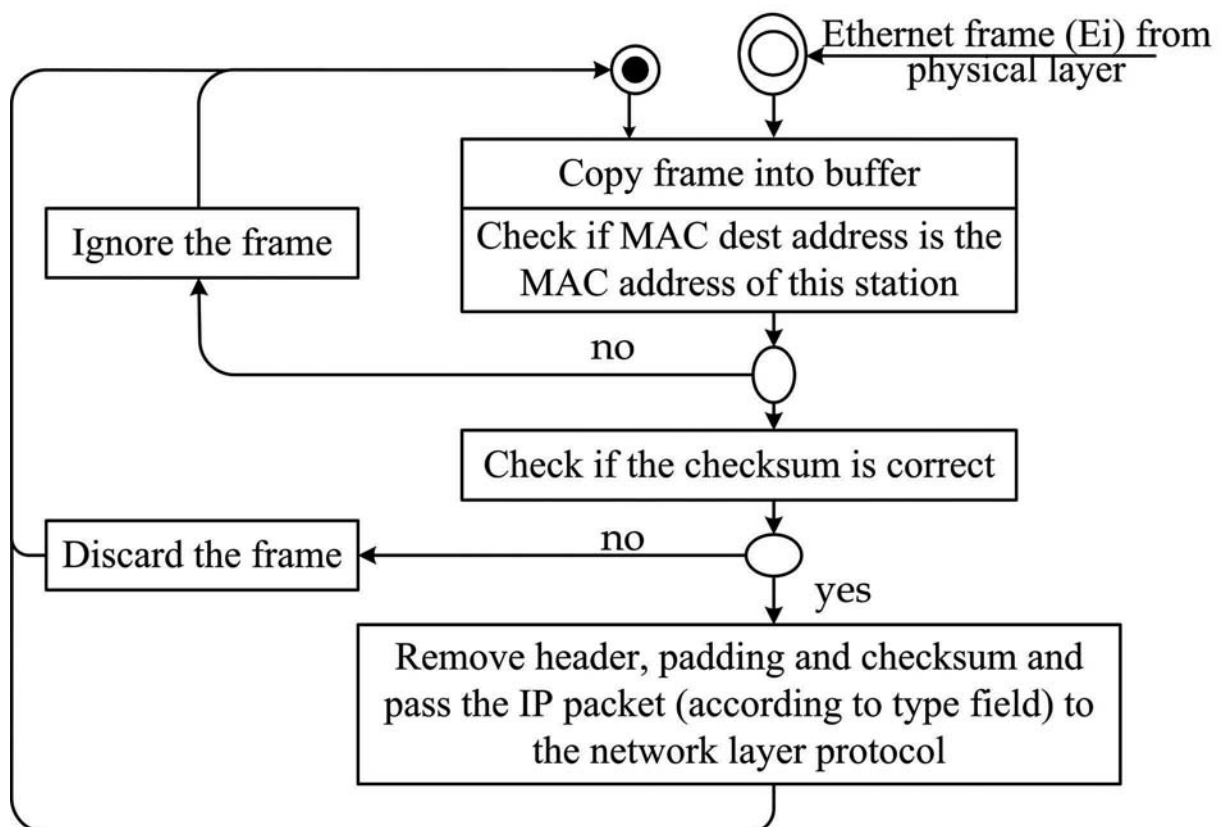


Figure 4-9: A modelling of the Ethernet protocol at the receiving side

The basic principle describing how Ethernet protocol works is described in figure 4-8 and figure 4-9 for sending side and for receiving side, respectively.

At the sending side, a Ethernet protocol instance obtains IP packets arriving from the network layer. For each IP packet, the Ethernet protocol instance constructs a Ethernet header and encapsulates the IP packet within a Ethernet frame. The protocol instance then senses the channel. If the channel is idle, it starts sending the Ethernet frame into the channel. During the sending process, the Ethernet protocol instance listens to the channel. If it receives collision or jam signal, it stops the transmission and increases the attempt time to one. If the attempt time reaches the pre-defined maximum number, it aborts the transmission. Otherwise, it waits for a exponential backup time and starts to sense the channel again.

At the receiving side, the Ethernet protocol instance receives the Ethernet frame from physical layer. It copies the frame into its buffer. If the MAC destination address of this frame is not the same as the MAC address of this station, the Ethernet protocol instance ignores the frame. Otherwise, the protocol instance verifies the checksum. If the checksum is not correctly, it discards the frame. Otherwise, it removes the Ethernet header, padding and checksum and passes the IP packets according to type field to the corresponding network layer protocol instance.

4.3.3 Summary

The data link layer protocol examples described in this session and their mechanisms are summarized in the table 4.1 below.

Protocol mechanisms		Data Link Layer Protocol Example		
		SLIP	PPP	Ethernet
Multiple Access Control				x
Point-to-Point		x	x	
Addressing	MAC Address			x
	IP Address	x	x	
Connection Man- agement	Connectionless	x		x
	Connection oriented		x	
Framing		x	x	x
Congestion Control			x	x
Multiple network layer protocol support				x
Authentication			x	
Error Control	Bit error control		x	x
	Packet error control			x

Table 4.1: Selected data link layer protocols and their protocol mechanisms

4.4 Internet's Network Layer

In the previous section we learned that the data link layer provides services for transferring a datagram across only an individual link between two direct connecting nodes (e.g. hosts or routers). To provide the communication between hosts connecting through several routers, services at the network layer need to be defined.

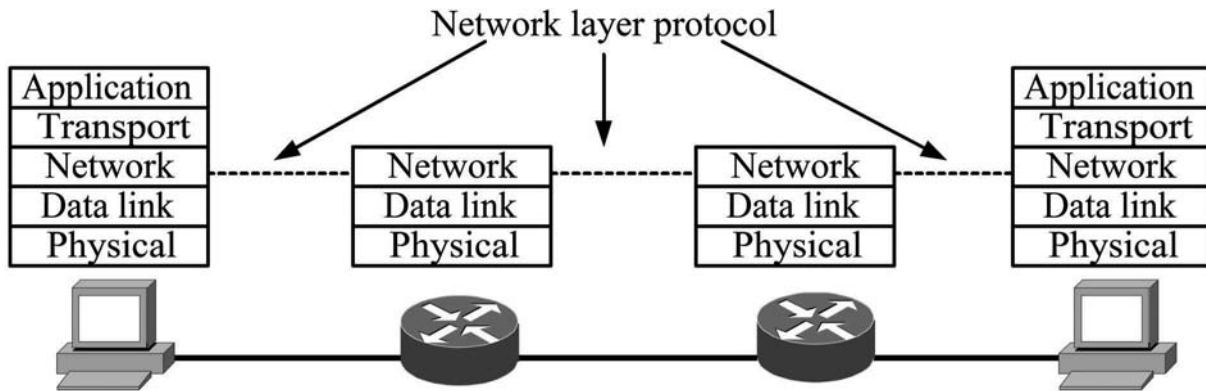


Figure 4-10: The network layer

The network layer provides services that enable the communication between hosts, between routers and between hosts and routers (figure 4-10). It provides logical communication between these devices. It is concerned with moving packets arriving from the transport layer from one host to another. In particular, the job of the network layer is to get the packets from the source to the destination and to pass the packets at the destination up the protocol stack to the upper layer protocol. This section describes the mechanisms and the selected network layer protocols for shifting packets from the transport layer of the source to the transport layer of the destination. We will see that unlike the data link layer, the network layer involves the whole communication path between two end nodes.

4.4.1 Internet's Network Layer Services

The network layer provides services to the transport layer. These main services at the network layer are addressing, best-effort delivery, connectionless service, routing, switching, IP input processing, IP output processing, multiplexing and demultiplexing, fragmentation and Reassembly, and error control. These services are summarized in the following.

4.4.1.1 Addressing

In order to transfer each datagram to its final destination, the network layer endpoint to which the packet should be delivered must be addressed with a unique address. There exist two Internet Protocols, the IPv4 and the IPv6. Therefore IPv4 addressing and IPv6 addressing will be described in the next coming section.

4.4.1.2 Unreliable services (best-effort delivery)

The Internet network layer protocols do not guarantee the handling of packet duplication, delayed or out-of-order delivery, corruption of data and packet loss. There is no receiving acknowledgement at the network layer protocols.

4.4.1.3 Connectionless service

The Internet network layer protocols do not provide connection establishment and connection teardown.

4.4.1.4 Routing

The network layer must be able to determine the communication path taken by packets as they travel from a sender to a receiver. The path determination is performed via routing described in chapter 3.7.

4.4.1.5 Switching (Packet Forwarding)

When a packet arrives at the input interface of a router, the router must move it to an appropriate output interface.

4.4.1.6 IP Input Processing

Before moving a packet from an input interface to an output interface the packet must be handled by the Internet protocol. This is done with the IP Input processing.

4.4.1.7 IP Output Processing

Before passing the packets to the appropriate data link layer, the packets must be processed by the Internet protocol. This is done with the IP output processing.

4.4.1.8 Error Control

Network layer protocols also provide mechanisms for recognizing the packet error but do not have mechanisms for the error correction because the network layer only provides unreliable services.

The above listed services are implemented in several Internet network layer protocols. These protocols will be detailed illustrated in the next coming session.

4.4.2 Internet's Network Layer Protocols

As summarized in the last session, the Internet's network layer offers unreliable and connectionless service, the so-called best-effort service delivery. When the network layer protocol instance receives a segment from the transport layer at the originating sender, it adds an adequate IP header to this segment to form an IP datagram, and passes the datagram to the next router on the path toward the destination. When the network layer protocol instance at a router (or at a host) receives an IP datagram, it uses its local routing tables and the destination address in the IP header of this datagram to decide where to pass the datagram. This process repeats until the IP datagram reaches its destination host. The Internet's network layer service neither guarantees that IP datagrams will arrive at their destinations within a certain time, nor assurance that datagrams will arrive in order sent. Indeed, the network layer service does not guarantee that a datagram will arrive at its destination.

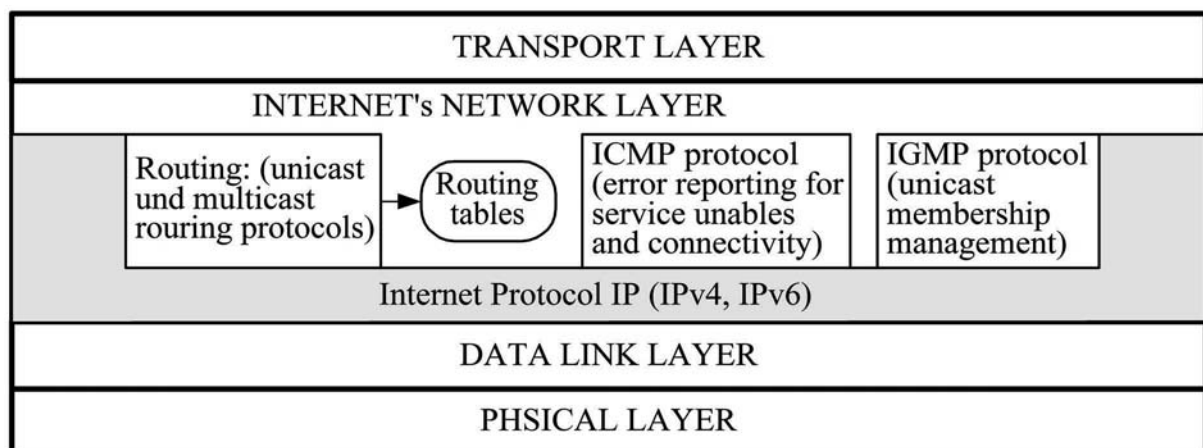


Figure 4-11: Internet network layer and its major components

Figure 4-11 shows four major components of the Internet's network layer – The Internet protocol IP, routing, ICMP and IGMP.

- *The Internet protocol IP* determines the assignment of IP addresses for network devices (e.g. end hosts, router, switches, mobile devices),

defining the format of IP datagrams and the actions taken by routers and end systems on the sending and receiving these IP datagrams over packet switched networks. There are two versions of IP protocol in use – the Internet Protocol version 4 (IPv4) and the Internet Protocol version 6 (IPv6). The most deployed version of the IP today is the IPv4 [RFC 791]. However, the IPv6 [RFC 2373; RFC 2460] is beginning to be supported. Its advantage is that IPv6 provides many more IP addresses than IPv4. Therefore, IPv6 is a key driver for new mobile/wireless applications and services in the future. IPv4 and IPv6 will be discussed in sessions 4.4.3 and 4.4.4..

- *Routing*. Routing is the path determination component that is described in session 3.7. Internet routing consists of unicast routing (e.g. OSPF, RIP and BGP) and of multicast routing (e.g. PIM, MOSPF). Most of routing protocols is built on top of the Internet protocol. These routing protocols will be illustrated in sessions 4.4.5 and 4.4.6..
- *ICMP (Internet Control Message Protocol)*. ICMP is used by network devices to send error messages indicating that the service is not available or that a host or router is not reachable. ICMP is built on top of Internet protocol. The ICMP messages are specified in the IP datagram with a protocol value of 1.
- *IGMP (Internet Group Management Protocol)*. IGMP operates between a host and its directly attached router. The protocol is used to manage the dynamic multicast group membership. In particular, IGMP enables a router to add and remove member to and from an IP multicast group. Moreover, IGMP allows a host to inform its attached routers that an application running on host wants to join a specific multicast group. Like ICMP, IGMP is built on top of Internet protocol. The IGMP messages are specified in the IP datagram with a protocol value of 2.

In the next sections, examples for Internet's network layer protocols will be discussed in more detail.

4.4.3 The Internet Protocol IPv4

The Internet Protocol (IPv4 and IPv6) provides unreliable service. In particular, the Internet Protocol does not provide service to handle problems of packet duplication, delayed or out-of-order delivery, data corruption and of packet losses. In contrast to circuit switching providing connection oriented services, the Internet protocol provides the connectionless service. No connection establishment, no connection termination procedures and no receiving acknowledgement are needed with the Internet Protocol. Thus, the

Internet Protocol does not maintain any state information about successive IP packets. Each IP packet is handled independently from all other packets. Four main functions of Internet protocol are IPv4 addressing, IP packet processing basic functions (multiplexing, demultiplexing, fragmentation and reassembly, encapsulation and decapsulation, Bit error recognition), IP input processing and IP output processing. These functions will be described in the following sub-session. This session provides a general detail on the operation of the Internet Protocol version 4. More about IPv4 can be founded in [RFC791, RFC3330, and RFC3171].

4.4.3.1 IPv4 Addressing

To provide seamless communication in the Internet, the Internet protocol must hide the details of physical networks and offer the facilities for abstracting the Internet as a large virtual network. Addressing is a critical component of the Internet abstraction. To transmit the user data across the Internet, a computer must know the address of the computer to which the packet is being sent and the address of computer from which the packet is sent. To give the appearance of a single, uniform system, all host computers and Internet devices must use a uniform addressing schema, and each address must be unique. Unfortunately, the physical network address (MAC address) discussed in session 4.2.1 do not be adequate because an Internet can include multiple network technologies and each technology defines its own address format.

To guarantee uniform addressing for all host computers and communication devices, the Internet Protocol software defines an addressing schema that is independent of the underlying physical address. To send a packet across an Internet, the sender places the destination address in the packet and passes the packet to the Internet Protocol instance for further delivery. The Internet Protocol software uses the destination address when it forwards the packet across the Internet to the destination. By this way, two applications can communicate without knowing either hardware addresses.

4.4.3.1.1 *The IPv4 Addressing Scheme*

An IPv4 address is a 32-bit number that uniquely identifies a device (such as computer, printer or router) on a TCP/IP network, and therefore there are a total of 232 possible IPv4 addresses. Each IPv4 address is typically written in so-called dotted-decimal notation (e.g. a.b.c.d), in which each byte of the address is written in its decimal form and separated by a dot from other byte in the address. For example the address 193.32.216.9 is binary notation is:

11000001 00100000 11011000 00001001

Each IPv4 packet sent across the Internet contains the IPv4 address of the sender (source address) as well as the IPv4 address of the receiver (the destination address).

4.4.3.1.2 The IPv4 Address Hierarchy

Before reaching the destination, each IP packet needs to travel through several routers in the Internet. In order to deliver an incoming IP packet, each router only needs the destination address of the physical network and not the destination host address. Therefore, each 32-bit binary number is divided into two parts –prefix and suffix. This two-level hierarchy is designed to make routing efficient. The address prefix identifies the physical network to which the destination device is attached, while the suffix identifies the destination device on that network. The prefix length is determined via adding the term “/n” to the IPv4 address, while the n indicates the number of significant bits used to identify a network to which this IP address belongs to. For example, 192.9.205.22/18 means, the first 18 bit are used to represent the physical network and the remaining 14 bits are used to identify the host.

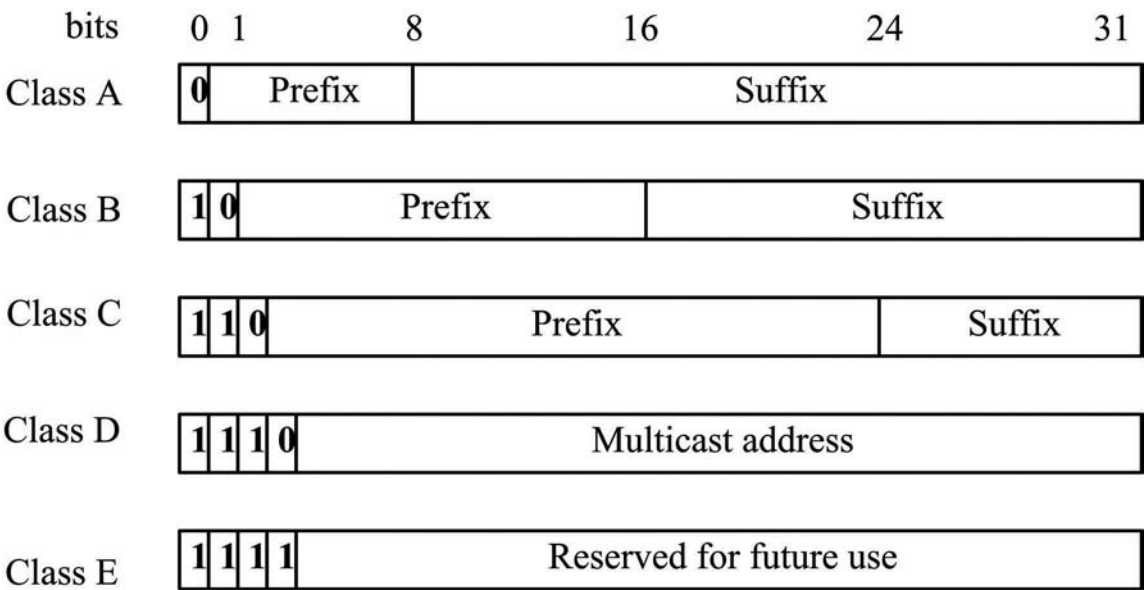


Figure 4-12: IPv4 address classes

4.4.3.1.2 IPv4 Address Classes

The original IPv4 addressing architecture defined five classes – class A, B, C, D and E (figure 4-12). The A, B and C class are used for host addresses and for end devices. Each of these classes has a different prefix and suffix size. Class D is used for multicasting that allows delivering packets to a group of receivers. The class E was reserved for the future.

4.4.3.2 IPv4 Datagram Format

The IPv4 datagram format is shown in figure 4-13. The key fields in the IPv4 datagram header are:

- *Version number.* This 4-bit field specifies the IP protocol version (e.g. 4 for IPv4 and 6 for IPv6)
- *Header length.* Because an IPv4 datagram can contain several options, these four bits are used to determine the length of the IPv4 header.
- *ToS.* This Type of Service (ToS) field is designed to carry information to provide quality of service for IP datagrams, such as prioritized delivery or drop precedence.
- *Datagram length.* This field indicates the total length of the IP datagram (header length plus data length).
- *Identifies, Flag, Fragmentation Offset.* These three fields are used by fragmentation and reassembly of a IP datagram. The Identifier field is used by the recipient to reassemble the IP packets without accidentally mixing fragments from different datagrams. The three bits of the flag field indicate whether the datagram should be fragmented or not. When fragmentation of a datagram occurs, the Fragment Offset field specifies the position in the overall fragments where the data in each fragment goes.
- *Time-to-Live (TTL).* The TTL field specifies how long the datagram is allowed to travel in the network, in term of router hops. Each router decrements the TTL value before transmitting the datagram. If TTL equals zero, the datagram is discarded.
- *Protocol.* This field identifies the higher-layer protocol, such as a transport layer protocol or a encapsulated network layer protocol, carried in the datagram. The values of this field were defined by RFC 1700, and are now maintained by IANA.
- *Header Checksum.* This field indicates a checksum computed over the header of the datagram. This value is used to provide basic protection against corruption in transmission
- *Source and destination IP address.* These fields carry the 32-bit address of the originator and of the recipient of the datagram. As discussed in 4.4.3.1, source and destination address are used by routers and end devices to delivery the datagram to its destination.
- *Options.* The Options field allows a datagram header to be included after the IPv4 standard header.
- *Data (payload).* The data field contains the user data to be sent over the Internet and PDUs of its upper layers

0	4	8	16	19	24	31
Version number	Header Length	Type of Service	Datagram length (bytes)			
Identifier			Flags	Fragment Offset		
Time to Live		Upper Layer Protocol	Header Checksum			
Source IP Address (32 bits)						
Destination IP Address (32 Bits)						
Options (max. 40 bytes)						
beginning of data						

Figure 4-13: IPv4 datagram format

4.4.3.3 IPv4 Basic Mechanisms

The IPv4 datagram processing basic mechanisms are multiplexing and demultiplexing, fragmentation and reassembly, error control, encapsulation and decapsulation. These mechanisms will be summarized in the following.

4.4.3.3.1 Multiplexing and Demultiplexing

The multiplexing service enables the Internet network layer protocol instance to convert the segment arrived from upper layer protocol (e.g. transport layer) into IP datagram and pass them into the network layer. At the receiving side the demultiplexing capability allows the network layer protocol instance to remove the IP header from the packets receiving from lower layer protocol and pass them to the right upper layer protocols.

4.4.3.3.2 Encapsulation and Decapsulation

When a message moves from an upper layer to the next lower layer of the TCP/IP protocol stack, the lower layer protocol instance attaches a header along with this message. The new PDU containing the header and the upper layer becomes the message that need to be passed to the next lower layer. The header contains control information placed at the beginning of each PDU. This header is used at the receiving side to extract the data from the encapsulated data packet. The packing of PDU at each layer is known as encapsulation. Decapsulation is the reverse process of encapsulation. This process occurs when

messages are received on the destination host. As a message moves up from the lower layer to the upper layer of TCP/IP protocol stack, each layer unpacks the corresponding header and uses the information contained in the header to handle and delivery the message to the next upper layer toward the exact network application waiting for the message.

IPv4 encapsulation is the process of packing each incoming PDU from an upper layer protocol into an IPv4 datagram. When the IPv4 protocol instance receives a PDU from the upper layer protocol (e.g. transport layer such as TCP, UDP or IPv4 sub layer such as ICMP, OSPF) it attaches an IP datagram header to this PDU. The result is a IPv4 datagram. The IP protocol instance then passes this datagram to the corresponding data link layer protocol. IPv4 protocol instance can get the messages from the lower-layer protocol such as Ethernet.

IPv4 decapsulation is the process of unpacking each incoming datagrams from lower layer protocol, removing the IPv4 header and passing the payload to the corresponding upper layer protocol.

4.4.3.3 Fragmentation and Reassembly

Each hardware technology specifies a maximal amount of data that a data link layer frame can carry. The limit is known as a maximum transmission unit (MTU). Thus, a datagram must be smaller or equal to the network MTU or it can not be transmitted. The fragmentation is needed if a network layer datagram is larger than MTU.

Because in an Internet a datagram can travel several heterogeneous networks before reaching its destination, MTU restrictions can cause problems. In particular, since a router can connect networks with different MTU values, the router can receive a datagram over one network that can not be sent over another.



Figure 4-14: an example of a router connecting two networks with different MTU

Figure 4-14 illustrates an example that a router interconnects two networks with MTU values of 1500 and 1000. In this example, the host H1 can transmit a datagram containing up to 1500 bytes. If H1 sends a 1500-byte datagram to H2, the router R will receive the datagram, but will not be able to send it across network 2. In this case, the datagram must be fragmented into small packets at

the sender and reassembled at the receiver. A router uses the MTU and the datagram header size to calculate the maximum amount of data can be sent in each fragment and the number of fragments will be needed. Three fields (FLAGS, Identifies and Fragment Offset) in the IPv4 datagram header are used for fragmentation and reassembly of the datagram. In particular, the FLAGS field is used to indicate whether this datagram is a fragment or not. The Identifies and Fragment Offset fields contain information that can be used to reassemble the fragments to the original datagram.

The reassembly is performed at the destination host. It deals with creating a copy of the original datagram from fragments. Because each fragment has a copy of the original datagram header, all fragments have the same destination address as the original datagram from which they were fragmented. Furthermore, the fragment carrying the last piece of the original datagram has the FLAGS field bit set to 0 whereas all other fragments have this flag bit set to 1. Thus the receiver performing the reassembly can verify whether all fragments have arrived successfully.

4.4.3.3.4 Error Control

IPv4 protocol instance provides bit error detection by using the Internet checksum mechanism that is described in section 3.1. On the receiving each IP datagram, the IPv4 protocol instance at the router or end host calculate the Internet checksum of the IP header and compare with the value in the header checksum field of this datagram. If the calculated checksum does not match the value in the header checksum field, the datagram is recognized as error and will be dropped.

4.4.3.4 IPv4 Input Processing

Figure 4-15 shows an overview of the IP datagram processing and IP routing within the Internet's network protocol. The IP packets arriving from lower layer protocols are held in the so called IP input queues located at the input ports. The IP input processing is responsible to handle the incoming packets arriving at the input ports, whereby it decides whether to drop them or to buffer them into these queues for waiting for service. It then removes the IP packets from these queues, verifying and processing them. Finally, if the packets are valid, the IP protocol instance passes the packets either to the appropriate transport layer protocols if the packets reached their destination or to the switching component. The mechanisms used here are packet error control (session 3.1), packet scheduling (session 3.4), active queue management (session 3.6) and packet switching (also called IP packet forwarding).

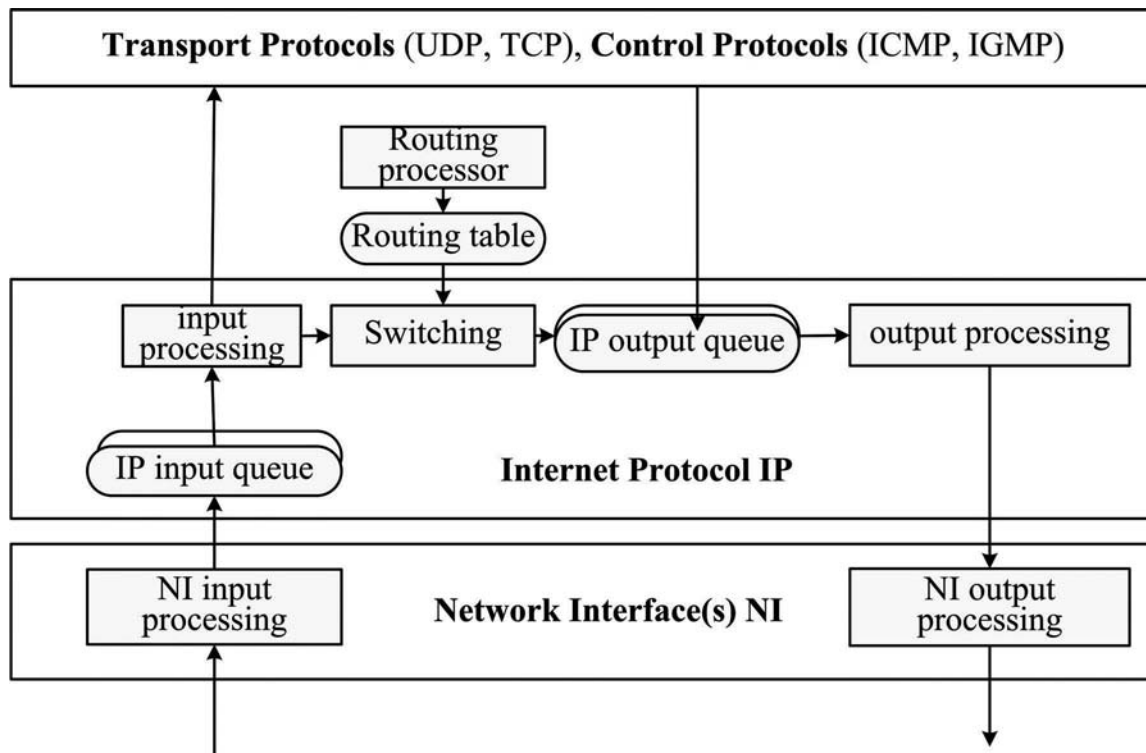


Figure 4-15: Overview of the Internet routing and IP datagram processing

For each incoming IP datagram, header length, checksum and version number must be verified. If the length of the IP datagram header (in bytes) is less than the length of a standard header (20 bytes), the datagram is invalid and is discarded. The IP protocol instance also drops the datagram if its header checksum is false or its version number is unrecognized.

If the IP datagram reaches its destination, the IP input processing reassembles the datagrams into the segments and passed them directly to the appropriate transport-level protocol by a function call. Otherwise, the IP input processing passes them to the IP forwarding if the host is configured to act as a router.

4.4.3.5 IPv4 Output Processing

IP output processing deals with handling the packets arriving at the output ports (interfaces). It decides whether to drop them or to buffer them into these queues for waiting for service. It then removes the IP packets from these queues for further verification and processing, it passes the packets to the appropriate data link layer protocol instance. The mechanisms used here are packet error control (session 3.1), packet scheduling (session 3.4) and active queue management (session 3.6). If the IP packet size is bigger than the MTU, the IP output

processing also fragments these packets before pass them to the data link layer protocol instance.

4.4.3.6 IPv4 Packet Forwarding

The IP packet forwarding is the process of taking a packet from an input interface and sending it out on an output interface. It includes the next hop selection and packet switching that will be discussed in this section.

4.4.3.6.1 Next Hop Selection

The IP packets traverse in Internet by following a path from their initial source to their final destination, possibly passing through several routers. Each router along the path receives the packet, uses the destination IP address in the header and the local routing table to determine a next hop to which the IP packet should be delivered. The router then forwards the packet to the next hop, either the final destination or the next hop.

On receiving an IP packet, a node (host or a router) first tries to find out, whether it is connected to the same physical network as the destination. To do it, it compares the prefix address of the destination address with the prefix of the address of each interface. If the match occurs, the destination host lies on the same physical network and the packet can directly delivered over that network.

If the node is not connected to the same physical network as the destination host, the node needs to determine the next router to which it need to send the packet to.

To select the next hop for each incoming IP packet, each router uses the destination IP address of this packet and the information in its routing table that contains a set of entries that each specifies a destination and the next hop used to reach that destination. The router than compares the destination IP address of this IP packet with the IP addresses of the next hop entries in its local routing table. This process performs a longest prefix match search in the routing table. The entry with the highest subnet mask matching the IP destination address in the header is selected as the next hop.

4.4.3.6.2 Packet switching

Transferring IP packets from a router's input port to a router's output port (see figure 4-16) is done with the switching. Switching can be accomplished in a number of ways: switching via memory, switching via a bus and switching via an interconnection network [Kur-2004].

- *Switching via shared memory.* This traditional switching is done under direct control of the routing processor. An input port with an arriving

packet first signals the routing processor via an interrupt. The packet is then copied into routing processor memory. The routing processor finds out an output port for this packet based on longest prefix match, and copies the packet to the output port's queue. Cisco Catalyst 8500 series (e.g. 8540, 8510) switch IP packet via shared memory

- *Switching via a shared bus.* In this technique, a packet is transferred directly from an input port to an output port over a shared bus without the involving of the routing processor. Since the bus is shared, only one IP packet at a time can be transferred over the bus. Thus, arriving packets are queued at the input port until the shared bus is free.
- *Switching via an interconnection network.* This technique enables an interconnection network consisting of $2N$ busses that connect N input ports to N output ports.

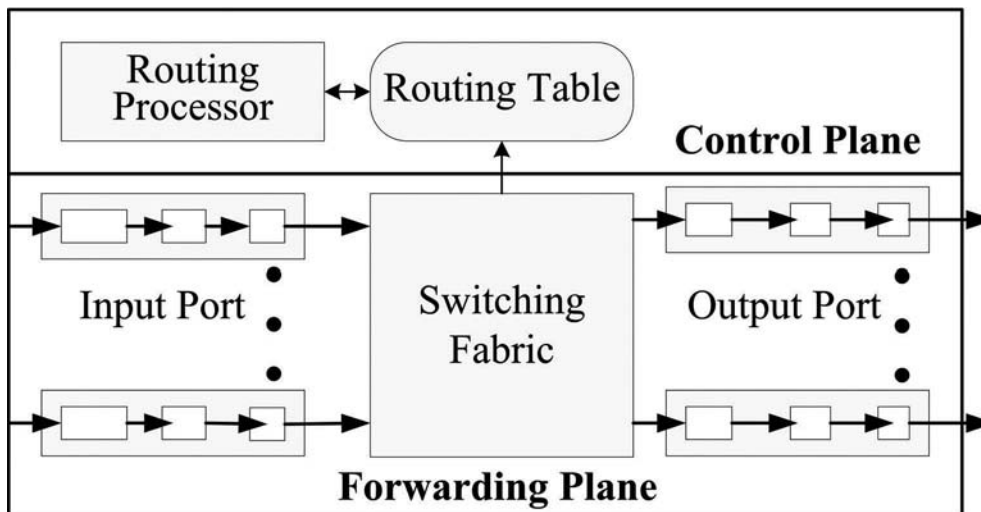


Figure 4-16: IP router architecture

4.4.4 The Internet Protocol IPv6

4.4.4.1 IPv4's Limitation

The Internet Protocol IPv4 has been used since the Internet was born and has worked very well until now, but it has many serious limits that the IPv6 has been designed by IETF [RFC2460, RFC2507, RFC3513, RFC5095] to overcome. The major limits of the IPv4 are as follows:

1. *Lack of address space.* The IPv4 uses 32-bit addresses that allow only an address space of 2^{32} addresses. This address space is too small for the current and future size of the Internet. This problem can only be

solved by moving to a larger address space. This was the primary motivating factor for creating the IPv6.

2. *Security problem.* Encryption, authentication and data integrity safety are not provided in IPv4. There exist particular firm solutions for security but there are no standards. IPv6 supports authentication and encryption possibilities.
3. *The management complexity of IPv4 is enormous.* With IPv4, each node in a network must be specially configured as IPv4 address, DNS-server and default router. This is done mostly still manually. Companies are bound with ISPs via IP addresses. Therefore changing of ISPs will be expensive because all computers must be new manually configured. In IPv6, this task is configured to be done automatically.
4. *Quality of Service (QoS).* QoS is a major keyword for multimedia and for wireless applications, but it is very restricted by IPv4. Only 8 priority classes can be defined within 3 bits in Type-of-Service (ToS) field in IPv4 header. This was an important motivation for designing the IPv6.
5. *Route optimization via elimination of “triangle routing”.* As illustrated in session 3.11, the routing in the mobile IPv4 is based on so-called triangle routing that operates between home agent, foreign agent and mobile node. The data packets addressed to the mobile node are intercepted by the HA (home agent), which tunnels them to the FA (foreign agent) towards the mobile node. Nevertheless, data sent from mobile IPv4 node to a wired node can be routed directly. Triangle routing problem delays the delivery of the datagrams and places an unnecessary burden on networks and routers. This problem is solved in the IPv6 (see section 3.11).

The mechanisms used in the IPv6 input and output processing as well as the IPv6 datagram forwarding are the same as in IPv4 except that the IPv6 addresses are used and thus need to be verified in all of these processes. Because of this only IPv6 addressing, IPv6 datagram format and IPv6 basic processing will be illustrated in this section.

4.4.4.2 IPv6 Addressing

Like IPv4, unique IPv6 address is needed to be assigned for each interface between a node (e.g. host or router) and a physical network. Therefore if a node (e.g. router) connects to two physical networks, the node is assigned two IPv6 addresses. Also like IPv4, each IPv6 address is separated into a prefix that identifies the network and a suffix that identifies a particular node on the

network. In spite of adopting the same approach for assigning IP addresses, IPv6 addressing differs from IPv4 addressing in significant ways.

- An IPv6 address is a 128-bit number that uniquely identifies a device (such as computer, printer or router) on a TCP/IP network, and therefore there are a total of 2^{128} possible IPv6 addresses. Each 128-bit IPv6 address is written in 8 groups of four hexadecimal digits with colons between the groups, e.g. 8000:0000:0000:0000:0123:4567:89AB:CDEF. Since many addresses will have many zero, leading zeros with a group can be omitted and one or more groups with zeros can be replaced by a pair of colons. For example the IPv6 address 8000:0000:0000:0000:0123:4567:89AB:CDEF can be written as 8000::0123:4567:89AB:CDEF.
- IPv6 addresses do not have defined classes. Instead, the boundary between prefix and suffix can be anywhere within the address and can not be funded out from the address alone. Thus, the prefix length must be associated with each address. An IPv6 address is therefore a combination of an IPv6 address and a prefix length. For example, the IPv6 address fe80::10:1000:1a4/64 contain information that the prefix length of this address is 64, and that the first 64 bits form the network part of the address and the last 64 bits form its host part.
- Ipv6 does not include a special address for broadcasting on a given network. Instead, each IPv6 address is one of three basic types: unicast, multicast and anycast.

4.4.4.3 IPv6 Datagram Format

Figure 4-17 shows the IPv6 datagram format with 40-byte fixed header. The most important changes introduced in IPv6 are addressed in the datagram format. First, the IPv6 increases the address space from 32 to 128 bits. Second, IPv6 adds the traffic class and the flow label field that allow to specify 28 priority classes instead of 23 by IPv4, and to label the packets belonging to a particular flow. Packets of up to 220 flows can be labeled with IPv6. Finally, a number of IPv4 header fields have been made optional. The resulting 40-byte fixed-length header allows for faster processing of IP datagrams at the routers and end hosts.

The following fields are defined in IPv6:

- *Version*. this four-bit field identifies the Internet Protocol as version 6
- *Traffic class*. The 8-bit IPv6 traffic class is used to identify different traffic classes or different priorities of IPv6 packets.
- *Flow label*. This 20-bit flow label can be used by source to label those packets for which the source requests a special handling by IPv6 routers.

- *Payload length.* This 16-bit field indicates the number of bytes in the IPv6 datagram following the 40-byte packet header.
- *Next header.* This field specifies the protocols to which the content in the data field will be delivered.
- *Hop limit.* The value of this field is decremented by one at each router that forwards the datagram. If the value of the hop limit field reaches zero, the datagram is discarded.
- *Source and destination address.* The 128-bit IPv6 addresses for source and destination.
- *Data.* This field contains the payload portion of the IPv6 datagram. This payload will be removed at the destination from IPv4 datagram and passed to the protocols specified in the next header field.

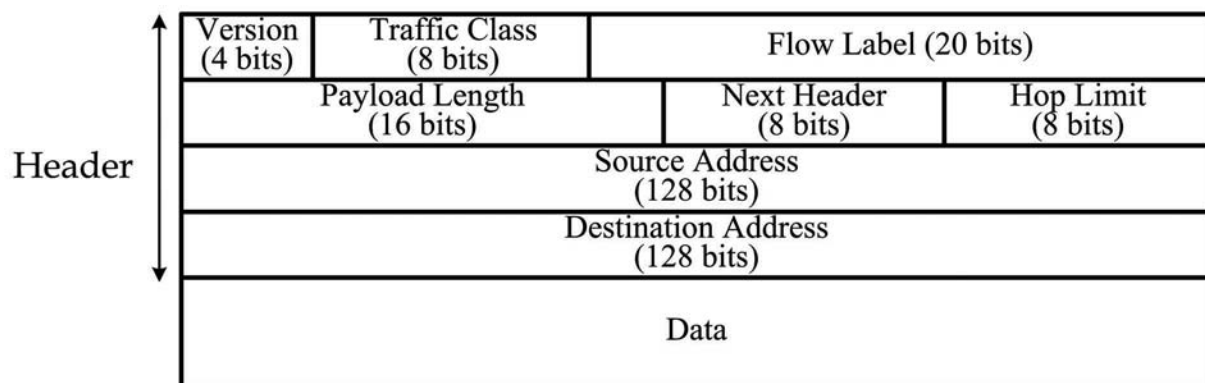


Figure 4-17: IPv6 datagram format

4.4.4.4 IPv6 Basic Mechanisms

Like IPv4, IPv6 supports multiplexing, demultiplexing, encapsulation and decapsulation. These mechanisms work as those specified for IPv4.

Unlike IPv4, the IPv6 fragmentation is only performed at the end host and not at the routers along the path toward destination. All IPv6 conformant hosts and routers must support packets of maximum 1280 bytes. When a host sends a IPv6 packet that larger than 1280 bytes, the router that is unable to forward it sends back an error message telling the host to fragment all future packets to that destination. Another major difference to IPv4 is that no checksum field is specified in IPv6 datagram format. Therefore IPv6 does not provide packet error control.

4.4.5 Unicast Routing Protocols in Internet

The fundamental mechanisms specified in each unicast routing protocols are illustrated in section 3.7.1. Based on these mechanisms this section addresses four standard unicast IP routing protocols – RIP, OSPF, IS-IS and BGP. The RIP (Routing Information Protocol), OSPF (Open Shortest Path First) and IS-IS (Intermediate System to Intermediate System) are used within an autonomous system (AS), and the BGP (Border Gateway Protocol) is used between ASs.

4.4.5.1 Routing Information Protocol Version 1

The Routing Information Protocol (RIP) is one of the oldest routing protocols, which works within an AS. RIP is an implementation of the distance vector routing algorithm described in section 3.7.1. The original RIP (also called RIP-1) is defined in the RFC 1058 [Hed-1988]. This RIP uses a special routing packet format to collect and to share information about distances to known destinations. Each RIP packet contains the following fields (figure 4-18):

- *Command (1 byte)* indicates whether the RIP packet was generated as a request or as a response to a request. While a request packet asks a router to send all or a part of its routing table, a response packet contains routing table entries that are to be shared with other RIP routers in the network. A response packet can be generated either in response to a request or as an update.
- *Version number (1 byte)* contains the version of RIP that was used to generate the RIP packets.
- *Zero fields (2 bytes)* are contrived as a means of providing backward compatibility with older RIP-like protocols.
- *Address Family Identifier (AFI) (2 bytes)* describes the address family which is presented by the IP address field.
- *Another Zero Field (2 bytes)* is used for providing backward compatibility with older RIP-like protocols.
- *Internetworking address (4 bytes)* contains an IP address of a host, a network or a default gateway. In a single request packet, this field contains address of the packet's originator. In a multiple response packet, this field contains the IP addresses stored in the originator's routing table.
- *Another zero field (4 bytes)* is used for providing backward compatibility with older RIP-like protocols.
- *Metric (4 bytes)* contains the packet's metric counter. This value is incremented as it passes through a router. The valid range of metrics for this field is between 1 and 15.

The routing information, which is contained in a RIP packet, is stored in a routing table. Each RIP routing table entry contains the following fields:

- *Destination IP addresses*: specifies the IP address of a known destination.
- *Distance vector metric*: represents the total cost of moving a packet from this router to its specified destination. Thus, the metric field contains the sum of the cost associated with the links building the end-to-end path between the router and its specified destination.
- *Next hop IP address*: contains the IP address of the next router in the path to the destination IP address.
- *Router change flag*: is used to indicate whether the route to the destination IP address has changed recently.
- *Router timers*: Two timers associated with each router are the router timeout timer and the router-flush timer. These timers work together to control and maintains the validity of each router stored in the routing table.

Command (1 byte)	Version Number (1 byte)	Zero field (2 bytes)	AFI field (1 byte)	Zero field (2 bytes)	IP address (2 bytes)	Zero field (4 bytes)	Zero field (4 bytes)	Metric field (4 bytes)

Figure 4-18: RIPv1 packet format

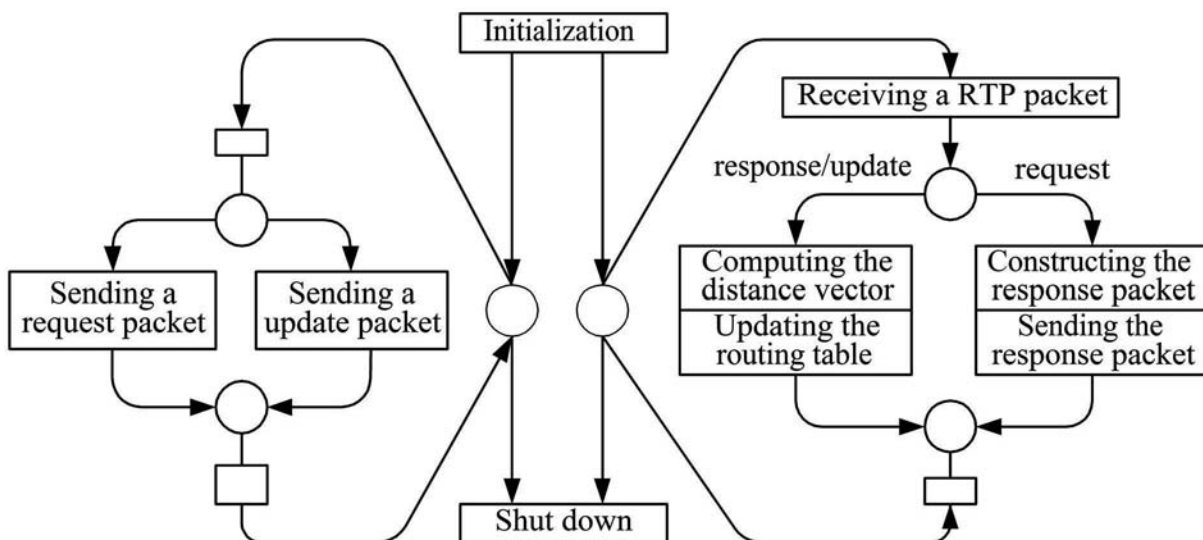


Figure 4-19: The RIPv1 basis principle

The basis principle of the original RIP is shown in figure 4-19. Each RIP router periodically copies a part of their routing table into RIP response packets and passes them to its neighbours. An RIP router can also send RIP request packets to a particular router to ask this router to send it all or a part of the routing table. On receiving a RIP response packet, the router recalculates its

distance vector and updates its routing table. On receiving a RIP request packet from a router, the router immediately sends its routing table or a part of its routing table to the requested RIP router.

4.4.5.1.1 Computing the distance vector

In the RFC 1058, there is a single distance vector metric – the hop count. The default hop metric in RIP is 1. Therefore, for each router that receives and forwards a packet, the hop count in the RIP packet metric field is incremented by one.

4.4.5.1.2 Updating the routing table

RIP requires all active RIP routers to broadcast their routing tables to neighbour RIP routers at a fixed interval by using of timers. Each RIP router timers are activated independently of the other RIP routers. Three timers are used to maintain and update the routing table. The first one is the update timer used to locally initiate the routing table update at the router level. The second one is the timeout timer (180 seconds) which is used for identifying invalid routes. Routes can become invalid if one of two events arrives: a route can expire or a router receives a notification from another router of a route's unavailability. In both events, a RIP router needs to modify its routing to reflect the unavailability of a given route. The third timer is the router flush timer which is used for purging invalid routes. When a router recognizes that a route is invalid, it initiates the flush timer (90 seconds). If this route is still not received after 270 seconds ($=180+90$), this route is removed from the routing table.

RIP-1 is a simple interior routing protocol in the Internet. Nevertheless it has several limitations. Some of greatest limitations are:

- Impossible to support path longer than 15 hops. Each time a packet is forwarded by a router, its hop counter is incremented by one. If the hop counter is 15, and the packet does not reach its destination, the destination is considered unreachable and the RIP packet is dropped.
- Reliance on fixed metrics to calculate routes. The next fundamental problem of RIP is its fixed cost metrics. These cost metrics are manually configured by administrators, and RIP cannot update them in real-time to accommodate network changes.
- Network intensity of table updates. A RIP node broadcasts its routing tables every 30 seconds. This can consume a lot of bandwidth in a large network with a lot of routers.
- Lack of support for dynamic balancing. It is impossible by RIP to dynamically load balance. For example, if a router has two serial

connections with the same link cost to another router, RIP would forward all its traffic over one of these two connections even though the second connection was available for use.

4.4.5.2 Routing Information Protocol Version 2

The Routing Information Protocol version 2 (RIP-2) was proposed as an extension of RIP that provided additional functionalities. These extensions were focused on the RIP packet format and on the new protocol mechanisms such as subnetting, authentication, and multicasting. The RIP-2 is specified in RFC 1723 [Mal-1994].

4.4.5.2.1 RIP-2 Packet Format

RIP-2 packet format is shown in the figure 4-20.

Command (1 byte)	Version Number (1 byte)	Unused field (2 bytes)	AFI field (1 byte)	Route tag (2 bytes)	IP address (4 bytes)	Subnet mask (4 bytes)	Next hop (4 bytes)	Metric field (4 bytes)

Figure 4-20: RIPv2 packet format

Each RIP-2 packet contains following fields:

- Command (1 byte): The command field remains unchanged from RIP.
- Version (1 byte): RIP-2 sets this field equal to 2.
- Unused field (2 bytes): The content of this field is ignored by RIP-2 routers and must be set to zeros by RIP routers.
- Address Family Identifier (AFI) (2 bytes): The AFI field is used for several purposes. The value in the AFI field indicates the network protocol address architecture contained in the network address field. A value of 2, for example, indicates that IPv4 is the network address architecture. Furthermore, setting the AFI field to 1 indicates that the receiving router should send a copy of its routing table to the requesting router. Moreover, the AFI field can also contain a special character string, 0xFFFF, which identifies the content of AFI row as the authentication information, and not as the routing information.
- Route tag (2 bytes): This field is for differentiating internal and external routes. Internal routes are routes that were discovered by the RIP-2 protocol. The external routes are those that were learned from other routing protocols, such as Border Gateway Protocol (BGP).
- Network Address (4 bytes): The network address field remains unchanged from RIP.
- Subnet Mask (4 bytes): contains the subnet mask of the network address.

- Next hop (4 bytes): contains the IP address of the next hop in the route to the destination which is specified in the network address field.
- Metric (4 bytes). This field remains unchanged from RIP

4.4.5.2.2 RIP-2 New Features

In comparison with RIP, RIP-2 additionally provides four significant new mechanisms

- *Authentication.* The RIP-2 supports the authentication of the router that initiates response messages. The reason for this is that the routers use response messages to propagate the routing information throughout a network and to update the routing tables. Authenticating the initiator of a response packet was proposed to prevent routing tables from being corrupted routing tables from a fraudulent source. A RIP-2 packet with authentication activated has the following structure. The content of first three fields (command, version and unused field) of RIP-2 packet remains unchanged. The AFI field of the first record in an authenticated message would be set to 0xFFFF. The Route Tag field following the AFI in this authentication entry is converted to the Authentication Type field that identifies the type of authentication being performed. The last 16 bytes of the RIP-2 packet – normally used for network address, subnet mask, next hop and metrics field – are used to carry password.
- *Subnet masks.* RIP-2 allocated a 4-bytes field to correlate a subnet mask to a destination IP address. This field lies directly behind the IP address field. Therefore, the 8-bytes of the RIP-2 routing entry are used to identify a destination.
- *Next hop identification.* This field makes RIP-2 more efficient than RIP by preventing unnecessary hops.
- *Multicasting RIP-2 messages.* Multicasting enables a RIP router to simultaneously advertise routing information to multiple RIP routers. This reduces overall network traffic and reduces the processing load of the routers.

4.4.5.3 Open Shortest Path First

The fundamental limitations of distance vector routing were become increasingly in the 1980's. An attempt to improve the scalability of networks was to define routes based on the link states rather than hop counts or other distance vectors. A link is a connection between two routers. The cost of that link can include attributes such as its transmission speed, delay or geographical distance. Open Shortest Path First (OSPF) [Moy-1998] is a routing protocol that was developed by IETF in response to the increased need for building of large

IP networks. OSPF is an interior gateway routing protocol that runs direct over IP and bases on the link state routing algorithm described in section 3.7.1. The protocol number in IP header for the OSPF is 89. Moreover OSPF packets should be sent with IP ToS field set to zero and the IP precedence field for OSPF is set equal to the value for control packets.

OSPF uses five packet types: hello, database description, link-state request, link-state update, link-state acknowledgement). These packets share a common header, known as the OSPF header. This header is 24 bytes long and has the following fields:

- *Version number (1 byte)*. The current version is 2, although older router may still run RFC 1131 (the OSPF version 1). RFC 1247, 1583, 2178, and 2328 [Moy-1991, Moy-1994a, Moy-1997, Moy-1998] all specify backward compatible variations of OSPF version 2.
- *Type (1 byte)*. There are five OSPF packet types that are identified numerically.
- *Packet Length (2 bytes)*. This field is used to inform the router receiving the packet of its total length. The total length includes the payload and header of the OSPF packet.
- *Router ID (4 bytes)*. Each OSPF router in an AS is assigned with a unique 4-byte identification number. Before transmitting any OSPF packets to other routers, an OSPF router populates the router ID field with its identification number.
- *Area ID (4 bytes)*. This field is used to identify the area identification number.
- *Checksum (2 bytes)*. The checksum field is used to detect bit error of each received OSPF packet. The Internet checksum is used as the bit error detection method in OSPF.
- *Authentication Type (2 bytes)*. OSPF can guard against the types of attacks that can result in spurious routing information by authenticating the originator of each OSPF packet. This field identifies which of the various forms of authentication is being used on this packet.
- *Authentication (9 bytes)*. This field is used to carry the authentication data that can be needed by the recipient to authenticate the originator of the OSPF packet.

As mentioned above, five different packet types are implemented in OSPF. Each of these packets is designed to support a particular routing function [RFC 2328]:

- *Hello packet (Type 1)*. Hello packets are sent periodically on all interfaces in order to establish and maintain neighbor relationships.

- *Database description packet (Type 2)*. These packets describe the content of link state database. DD packets are exchanged between two OSPF routers when they initialize an adjacency.
- *Link state request packet (Type 3)*. After exchanging DD packets with a neighbor router, a router may find that a part of its link state database is out of date. The link state request packets (LSRs) are used to request pieces of a neighbor's link state database that are more up-to-date.
- *Link state update packet (Type 4)*. LSU packets are sent to all routers within a AS via flooding. These packets are used to carry the LSA packets to neighboring routers. There are five different LSA packet types: Router LSA; Network LSA; Summary LSA-IP network, Summary LSA-Autonomous System Boundary Router; and AS-external LSA. These packet types are described in RFC 2328.
- *Link state acknowledgement packets (Type 5)*. OSPF features a reliable flooding of LSA packets. This means that receipt of the LSA packet must be acknowledged. The link state acknowledgement (LSACK) is designed for this purpose.

The basis principle of the OSPF is described in the figure 4-21. Each OSPF router periodically broadcasts hello packets to its neighbor. Two OSPF routers also exchange the database description (DD) packets as they initialize an adjacency. On receiving the OSPF packets, each OSPF protocol instance verifies the checksum value in the checksum field of this packet. If the checksum is failed, this packet is dropped. Otherwise, the OSPF protocol instance tries to authenticate this packet. If the router can not authenticate this packet, it drops the packet. Otherwise, the router processes the packet and takes action according to the packet type. If the incoming packet is a hello packet, the router compares its old neighbor list with the new one and updates its neighbor list (NBL). If the packet is a LSU packet, the router compares the LSA packets of this LSU packet with the LSA packets in its LSA database and updates its LSA database. Because LSU packets are reliable flooded, the router then acknowledges the initiators of the newly LSA packets. However, acknowledgements can also be accomplished implicitly by sending LSU packets.

By receiving LSACK packets, the router makes many consistent checks before it passes them to the flooding procedure. In particular, each LSACK packet has been associated with a particular neighbor. If this neighbor is in a lesser state than exchange, the LSACK packet is discarded. By receiving a DD packet, the router compares it with the last received DD packets and makes decision of sending LSR packets. When a router receives a LSR packet, it processes the packet and sends a LSU packet as the response.

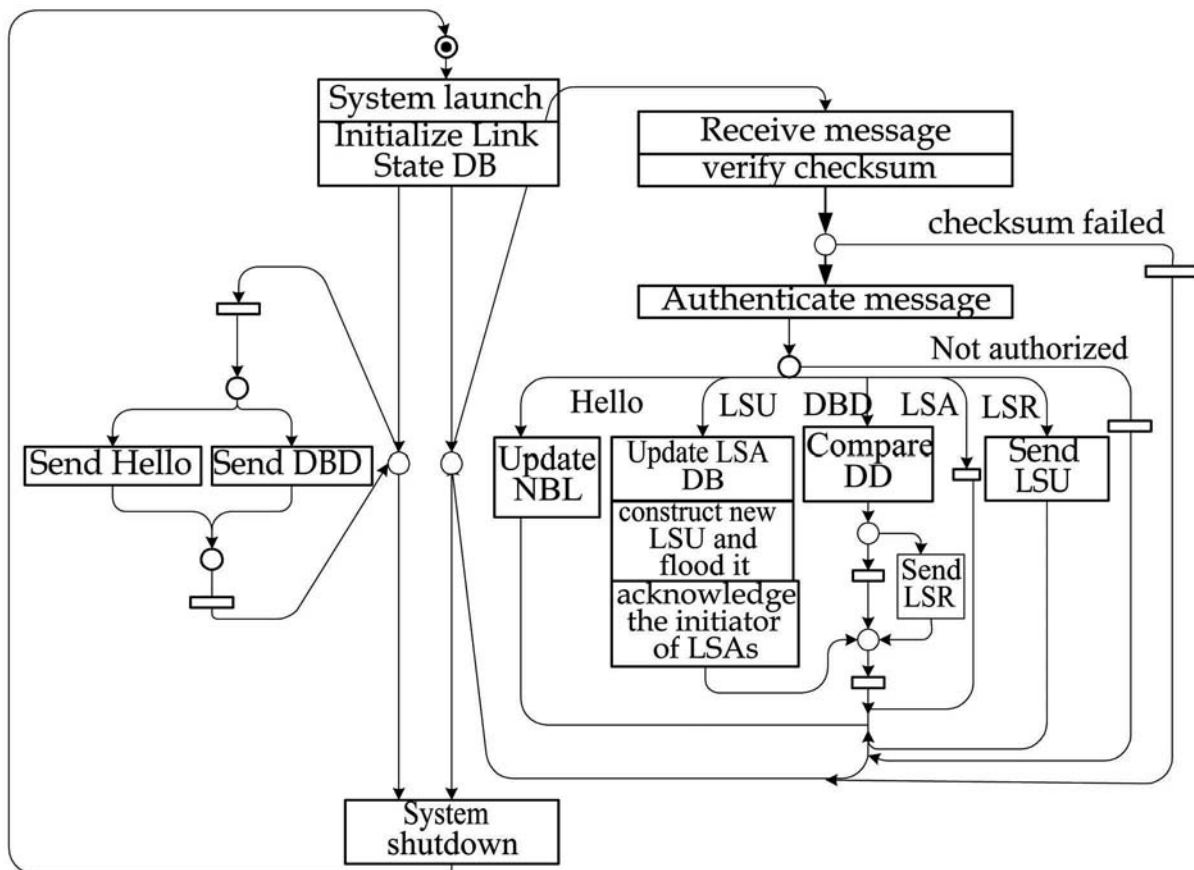


Figure 4-21: a simple modelling of the OSPF

4.4.5.4 Border Gateway Protocol

Today's Internet consists of an interconnection of multiple autonomous systems (ASs), which are connected to each other in arbitrary ways. Border Gateway Protocol (BGP) is used to exchange routing information and is the protocol that operates between ASs. Customer networks usually employ an Interior Gateway Protocol such as OSPF or RIP for exchange their routing information within their networks. To exchange ISP routes, BGP is used by customers connecting to ISPs and by ISPs use BGP.

BGP is in its fourth version at the time of writing (BGP-4). It is an Internet standard that is specified in RFC4271 [RLH-2006]. BGP does not belong to either of two fundamental routing algorithms (distance vector and link state vector). Unlike these routing algorithms, BGP advertises more complete paths as an enumerated list of ASs from itself to reach a particular network, and not advertises cost information to its neighbors. Special path attributes describe the characteristics of paths, and are used in the process of route selection. BGP routers use the TCP on port 179 to perform a reliable communication with each other, instead of layering the routing message direct over IP, as done by OSPF

and RIP. In particular, BGP neighbors exchange full routing information when the TCP connection between neighbors is first established. When routing table changes are detected, the BGP routers send to their neighbors only those routes that have changed. BGP routing information updates advertise only the optimal path to a destination network, and periodic routing information updates are not sent by BGP routers.

BGP is very robust and scalable routing protocol employed in the Internet. At the time of this writing, the Internet BGP routing tables contain 325.087 active BGP entries [BGP-2010]. To achieve scalability at this level, BGP uses many route parameters, called attribute, to define routing policies and maintain a stable routing environment. In addition to BGP attributes, classless inter-domain routing (CIDR) is used by BGP to reduce the size of routing table.

4.4.5.4.1 BGP Message Header Format

The BGP message header is specified in RFC4271 [RLH-2006]. Each message has a fixed-size header (figure 4-22) consisting of three fields – Marker, Length and Type.

- Marker. This 16-byte Marker field is used for compatibility and must be set to all ones.
- Length. The value of this 2-byte field indicates the total length of the BGP message. This value must be at least 19 bytes, which is the fixed-size header, and no greater than 4096 bytes.
- Type. This 1-byte field specifies the type code of the BGP message. Depending on the message type, there may or may not be a data portion following the fixed header in figure 4-22. There are four type codes
 1. OPEN
 2. UPDATE
 3. NOTIFICATION
 4. KEEPALIVE

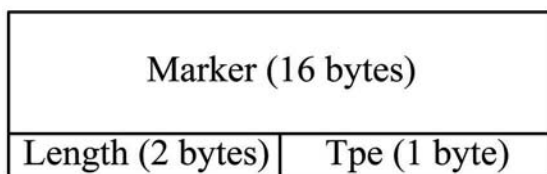


Figure 4-22: BGP message header format

4.4.5.4.2 BGP Messages

As described in the previous section that BGP supports four types of messages – OPEN, UPDATE, NOTIFICATION and KEEPALIVE. These messages and their utilization will be illustrated in this section.

OPEN Message

An OPEN message is the first BGP message sent by each side after the TCP three-way handshake is completed. The BGP OPEN message is used to open a BGP session. In addition to the fixed-size BGP header, the OPEN message contains information about the BGP neighbors initiating the session, and the information about the supported and negotiated options including the BGP version, AS number, hold down time value, BGP Identifier, Optional Parameter and Optimal Parameters Length.

UPDATE Message

BGP routers use the UPDATE message to exchange the routing information with BGP peer routers. When a BGP session is established, UPDATE messages are transferred between the peers until the complete BGP routing table has been exchanged. Each BGP router uses the information contained in the UPDATE message to construct a graph that describes the relationships between various autonomous systems in order to update the BGP routing information base and BGP routing table. Furthermore, BGP routers use UPDATE message to advertise feasible routes that share common path attributes to a peer, or to withdraw multiple unfeasible routes from services.

In addition to the fixed-size BGP header, the UPDATE message may include information about withdrawn routes length, withdrawn routes, total path attribute length, path attribute and network layer reach-ability information.

NOTIFICATION Message

The NOTIFICATION messages are sent to signal a peer when an error is detected in a BGP session. In addition to the fixed-size BGP header, the NOTIFICATION message contains three fields – Error code, Error Subcode and data. The first field indicates the type of NOTIFICATION, whether fixed-size message header error, OPEN message error, UPDATE message error, hold timer expired or finite state machine error. The second field gives more specific information about the report error. The third field is used to identify the reason for this notification message.

KEEPALIVE Message

A positive confirmation of an OPEN message is a KEEPALIVE message. A BGP router sends KEEPALIVE messages at an interval specified by the KEEPALIVE interval timer in the BGP configuration to determine if its peers are reachable. A KEEPALIVE message consists of only the fixed-size message header that has a length of 19 bytes.

4.4.5.4.3 BGP Attributes

BGP uses a set of attributes in the route selection process in order to determine the best route to a destination network when multiple paths exist for a particular destination. These attributes specified in RFC4271 are

- Origin
- AS_path
- Next hop
- Multi-exist discriminator
- Local preference
- Atomic aggregate
- Aggregator

4.4.5.4.4 Basic Principle of BGP

So far we have discussed about the BGP message format, BGP messages and the BGP attributes. Based on this fundament, this section will describe how the BGP routing protocol works.

When a BGP router comes up on the Internet, it first setup a TCP connection with each of its BGP neighbor routers. After TCP three-way handshake is completed, the BGP router establishes BGP session with its BGP neighbor routers by sending the OPEN messages. At the beginning, the BGP router then uses the UPDATE message to download the entire routing table of each neighbor router. After that it only exchanges shorter update message with other BGP routers.

BGP routers send and receive UPDATE messages to indicate a change in the preferred path to reach a network with a given IP address. If the BGP router decides to update its own routing tables because this new path is better, then it will subsequently propagate this information via sending UPDATE messages to all of the other neighboring BGP routers to which it is directly connected, and these BGP neighbors will in turn decide whether to update their own routing tables and propagate the information further.

Each BGP router maintains a Routing Information Base (RIB) that contains the routing information. Three parts of information are contained in RIB [RLH-2006]:

- *Adj-RIBs-In*. Adj-RIBs-In stores unprocessed path information received from neighbouring BGP routers (also called peer).
- *Loc-RIB*. Loc-RIB contains the actual path information that has been selected by the BGP router. The routing Information in Loc-RIB is processed by Adj-RIBs-In.
- *Adj-RIBs-Out*. Adj-RIBs-Out contains the path information the BGP router chooses to send to neighbouring BGP routers in the next UPDATE messages.

BGP routers exchange the path information using four BGP messages (OPEN, UPDATE, KEEPALIVE and NOTIFICATION) described above. After receiving an UPDATE message from a neighboring router, the BGP router first verifies each field of this message. If the UPDATE message is valid, the BGP router performs following three steps:

- *Update*. If the path information for an IP address in the update message is different from the previously path information received from this router, then the Adj-RIBs-In database is updated with the newest path information. Once the BGP router updates the Adj-RIBs-In, the router shall run its Decision process.
- *Decision*. If it was new path information, then a decision process is performed. This process determines which BGP router, of all those presently stored in the Adj-RIBs-In, has the best routing path for the IP address in the UPDATE message. If the best path selected is different from the one currently recorded in the Loc-RIB, then the Loc-RIB is updated.
- *Propagation*. If the decision process found a better path, then the Adj-RIBs-Out is updated, and the BGP router sends out UPDATE messages to all of its neighbouring BGP routers to tell them about the better path. Each BGP router must run their own update and decision process in turn to decide whether or not to update their RIB, and then propagates any new and improved paths to neighbour BGP routers in turn.

4.4.6 Multicast Routing Protocols in Internet

Multicast routing protocols can be classified via their operation areas or via methods used for forwarding the multicast packets. Based on operation areas, multicast routing protocols are classified into intra domain routing protocols operating within an AS, and inter domain routing protocols functioning between

ASs. For example, Distance Vector Multicast Routing Protocol (DVMRP), Multicast Open Shortest Path First (MOSPF) and Protocol Independent Multicast (PIM) belong to the intra domain routing protocol. The Border Gateway Multicast Routing Protocol (BGMP) and Multicast Source Discovery Protocol (MSDP) are the inter domain multicast routing protocols.

Based on the forwarding methods, the multicast routing protocols are categorized into three classes - sparse mode, dense mode and sparse-dense mode protocols. Sparse mode multicast protocols use a pull mode to delivery the traffic. This means that multicast traffic will only be forwarded to such networks that have active receivers. Therefore, these protocols tend to use shared tree techniques and a host need to subscribe to a multicast group to become its member to receive the multicast data. In contrast to the sparse mode multicast, dense mode multicast protocols use a push mode to flood traffic to every corner of the network. These protocols tend to use the source-based tree technique and include by default all multicast routers in the multicast distribution trees. Thus, multicast routers need to send prune message if they don't want to receive the data. Since using the push mode, the dense mode protocols are optimized for such networks where most hosts are member of multicast groups. A combination of sparse mode and dense mode is called spare-dense mode protocols. Routers running spare-dense model protocols can switch between spare mode and dense mode.

In the rest of this section, some selected multicast routing protocols will be described. These protocols are DVMRPv3 (Distance Vector Multicast Routing Protocol), MOSPF (Multicast Extension to OSPF) and PIM (Protocol Independent Multicast). The illustration of these protocols bases on the fundamental mechanisms used to develop multicast routing protocols addressed in section 3.7.2.

4.4.6.1 Distance Vector Multicast Routing Protocol

Distance vector multicast routing protocol version 3(DVMRPv3) [WPD-1988, Her-2000, Pus-2004] is an interior routing protocol, which uses the dense mode to forward the multicast traffic. The basis principle of DVMRPv3 is the use of the distance vector algorithm to discover the topology information and of the IGMPv3 for exchanging the routing protocol packets.

The common DVMRP packet header is shown in figure 4-23. DVMRP packets consist of two portions: a fixed length IGMP header and a stream of tagged data. The tagged data portion is used to carry for example the router ID, IP addresses of neighbours, group addresses, source host addresses and prune life time for each interface. The common packet header consists of following

fields: The type field described the DVMRPv3 packet type and is defined as hexadecimal 0x13. The major version of 3 and minor version of 0xFF are used to indicate a compliance with the version 3 specification. Checksum field is used for bit error control of the whole DVMRP packet. The code field defines the DVMRP types shown in table 4-2.

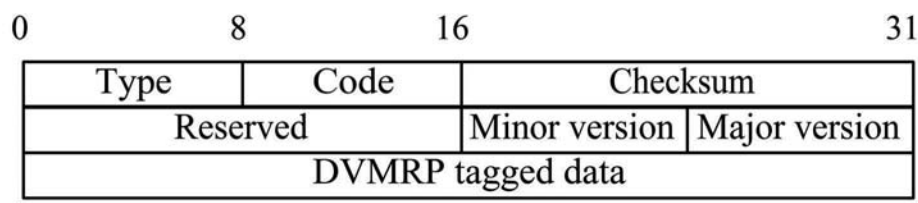


Figure 4-23: Format of DVMRPv3 packets [Pus-2004]

Code	DVMRP packet type	Description
1	DVMRP probe	For discovering the neighbors
2	DVMRP report	For exchanging the routing atble
7	DVMRP prune	For printing multicast delivery trees
8	DVMRP graft	For grafting multicast delivery trees
9	DVMRP graft ack	For acknowledging graft packet

Table 4-2: DVMRPv3 packet types [Pus-2004]

DVMRPv3 implements four fundamental mechanisms (source-based tree, reverse path forwarding, pruning and grafting) discussed in the previous paragraphs.

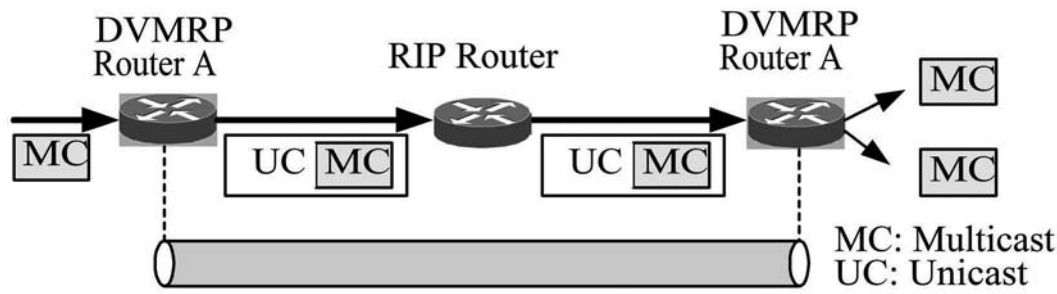


Figure 4-24: Tunneling by DVMRPv3

Since only a small fraction of the Internet routers is multicast-capable, DVMRPv3 additionally supports tunneling that enables the forwarding of multicast packets in respect of this heterogeneity. If one router is multicast-capable (for example router A in figure 4-24) but all of its immediately neighbors are not, the router then encapsulates the multicast packets inside a standard IP unicast packets and addresses them to the routers

that do support native multicast routing (for example router B in figure 4-24). Between these two native multicast routers, the unicast routers forward the IP packets using IP unicast routing and other IP services. When the multicast packets arrive at the destination multicast router (router B), this router extracts the multicast packets and forwards it to the attached networks.

The multicast forwarding is performed on the basis of DVMRPv3. This process is illustrated in the following. Each multicast sender floods multicast packets along the pre-configured source-based tree to all interested routers by using of the reverse path forwarding (RPF) rules. These packets arrive at the intermediate routers, which receive the same multicast packets more time over different routes and use the RPF rules to discard or to forward these packets. If leaf routers do not have any group members on their subnets, these routers send prune messages to upstream router to stop unnecessary multicast traffic. The DVMRP prune message contains a prune lifetime that determines how long a prune branch will remain pruned. When the prune lifetime expires, the pruned branch is joined back onto the multicast delivery tree. When a router has received a prune message from all its dependent downstream routers for a given group, it will propagate a prune message upstream to the router from which it receives the multicast traffic for that group. When new members of a pruned leaf router want to join a multicast group, the router sends graft message to its upstream neighbour to add the pruned branch back on the multicast tree.

The main issue of DVMRP is the scalability because of the periodically flooding of the multicast traffic occurring when prune states expire. In this case, all DVMRP routers will receive unwanted traffic until they have sent the prune messages. Furthermore, the routers must maintain and update prune states per source, per interface and per multicast group within each multicast routing table and forwarding table. This leads to scalability problems with a large number of multicast groups.

4.4.6.2 Multicast extension to Open Shortest Path First

The multicast extension to Open Shortest Path First (MOSPF) is specified in RFC 1584 [Moy-1994b]. This protocol operates in an autonomous system (AS) that uses the OSPF protocol. MOSPF implements the link state routing algorithm as OSPF and is a spare mode multicast routing protocol. The basis idea for extending OSPF to multicast is to enable each router to listen to LSA messages in order to detect the multicast groups. By this way, routers obtain locations of the receivers and thus routers can build a multicast tree for each group.

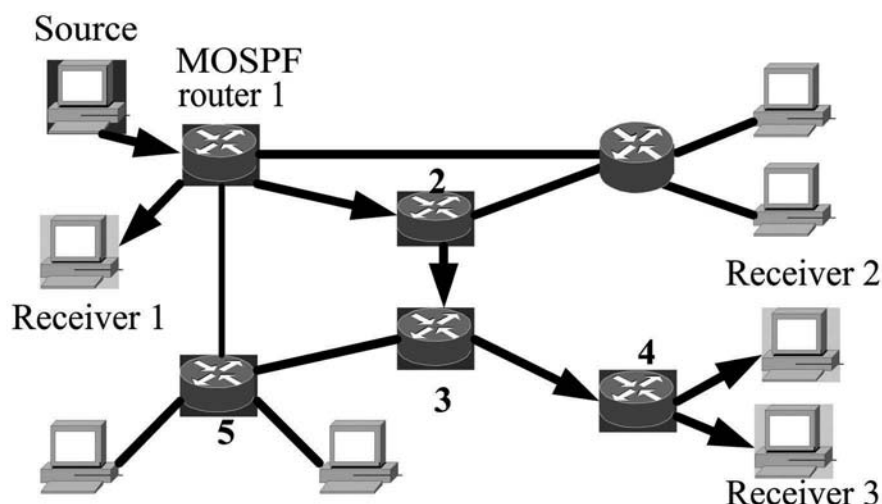


Figure 4-25: Multicast delivery tree by MOSPF

Each MOSPF router uses the link state advertisement (LSA) database built by OSPF [Moy-1998] to determine a shortest path tree for each pair of source and group. To inform the routers about the multicast memberships, the so called multicast capable bit is added to link state advertisement (LSA) packets that are flooded by routers as a part of the OSPF. Thus, the routers know the topology and their membership, so that the multicast tree spans only MOSPF routers and subnets that have multicast receivers. Because of this, MOSPF is a sparse mode multicast protocol. That means the multicast delivery tree for each group G only spans the MOSPF routers that have interest in receiving multicast traffic of G . An example for this is shown in figure 4-25: the shared tree for group G spans only the MOSPF routers 1, 2, 3 and 4; it does not span the MOSPF router 5 and 6 because these routers do not have any membership hosts.

In addition to the regular OSPF routing table and the LSA database, each MOSPF router maintains a group membership table describing the group membership on all attached networks for which this router operates either as a designed router or as a backup designed router. Within each subnet, these group memberships are maintained through one or two MOSPF routers in a local group database. Updating this local state database is performed via interaction with the IGMP protocol through the following steps. The MOSPF designed router (DR) periodically issues IGMP queries on each subnet and the DR and backup designed router listen to the IGMP host membership reports. Based on the receiving membership reports, the designed router constructs the group membership LSAs (the OSPF LSA with additionally multicast capable bits) and floods them within entire OSPF area. Other multicast routers in the domain receive these group membership LSAs so that they can learn the topology and the membership. Thus, a multicast tree, which spans only the MOSPF routers

and subnets that have group members, can be determined using all pair shortest path first algorithm, so that pruning and grafting do not need to be implemented in MOSPF. This is the big difference to DVMRP.

Using MOSPF, hosts can join and leave a group without pruning and grafting, but at the expense of a much large LSA database, since the database must contain one entry for every group on every link in the network. Moreover, the all pair shortest path computation must be performed separately for every source, which results in a expensive operation.

4.4.6.3 Protocol Independent Multicast

Protocol Independent Multicast (PIM) is a collection of three multicast routing protocols, which are optimized for different environment. These protocols are PIM Sparse Mode (PIM-SP), PIM Dense Mode (PIM-DM) and Bi-directional PIM. The third PIM protocol is known as Sparse-Dense Mode. This protocol is less widely used and therefore it will not be discussed in this section. Generally, either PIM Sparse Mode or PIM Dense Mode will be used within a multicast domain. However, they also may be used together within a single domain, whereby the Sparse Mode can be used for some multicast groups and Dense Mode for others.

The main difference to DVMRP and MOSPF is that PIM is not dependent on the routing algorithms (distance vector routing, link state routing) provided by any particular unicast routing protocol. However, any implementation supporting PIM requires the presence of a unicast routing protocol to provide routing table information and to adapt to topology changes.

This section starts with a description of the PIM packet format. After that, the protocol functionalities of PIM-SM and PIM-DM are discussed.

4.4.6.3.1 PIM Packet Format

All PIM control messages have the IP protocol number 103. PIM messages are either unicast (e.g. registers and register-stop) or multicast with TTL of 1 to the “ALL-PIM-ROUTERS” group (e.g. join, prune message).

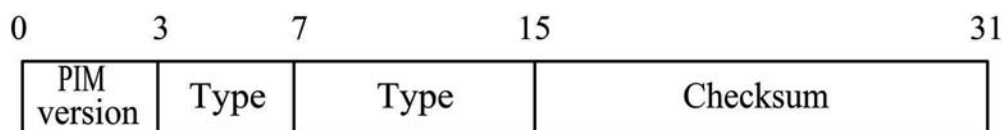


Figure 4-26: The PIM common header [FHH-2006]

All PIM messages have a common header described in figure 4-26. In that figure, the field “PIM version number” is 2 and the checksum is for the whole PIM message. The type field is specified for specific PIM messages and is shown in table 4-3 [FHH-2006].

Message Type	Description
0 = Hello	Multicast to ALL-PIM-ROUTERS
1 = Register	Unicast to RP
2 = Register-stop	Unicast to source of Register packet
3 = Join/Prune	Multicast to ALL-PIM-ROUTERS
4 = Bootstrap	Multicast to ALL-PIM-ROUTERS
5 = Assert	Multicast to ALL-PIM-ROUTERS
6 = Graft (used in PIM-DM only)	Unicast to RPF of each source
7 = Graft-Ack (used in PIM-DM only)	Unicast to source of graft packet
8 = Candidate-RP-Advertisement	Unicast to domain's BSR

Table 4-3: PIM message

These packet types are summarized as follows:

- *Hello*. Hello messages are sent periodically on each PIM enabled interface. These messages allow a router to learn about its neighbouring PIM routers and thus all PIM routers in its network.
- *Register*. A Register packet is the first packet sent by unicast from a DR to a root of shared tree (such as RP). Register packets are used by PIM-SM and will be discussed the next subsection in more detail.
- *Register-stop*. This packet is sent from a shared tree root to the DR to tell the DR to stop sending encapsulated multicast packets.
- *Join/Prune*. While join packets are sent to build source-based tree (by PIM-DM) or shared tree (by PIM-SM), Prune packets are sent to move a branch from the multicast tree.
- *Bootstrap*. In order to map a particular multicast group address to the same RP, a bootstrap packet is needed by every PIM-SM router.
- *Assert*. PIM Routers generate Assert messages within a shared LAN to elect a designed router for a multicast group. This DR is then responsible to forward the multicast packets to all sources in this LAN and thus the duplication can be avoided.
- *Graft and Graft-ack*. A Graft packet is sent only by PIM-DM to inform the upstream router of interesting in receiving the multicast data. Graft-ack is the acknowledgement for the graft message.
- *Candidate-RP-Advertisement*. This message is sent by PIM-SM routers to advertise each multicast group an RP.

4.4.6.3.2 PIM Spare Mode

Version 1 of PIM-SM was created in 1995 and is now considered obsolete. PIM-SM version 2 was standardized in RFC 2117 (in 1997), updated by RFC 2362 (in 1998) and is now again obsolete. The new PIM-SM protocol is specified in RFC 4601 (in August 2006) [FHH-2006] that obsoletes the RFC 2362. This section describes an overview of the PIM-SM specified in RFC 4601.

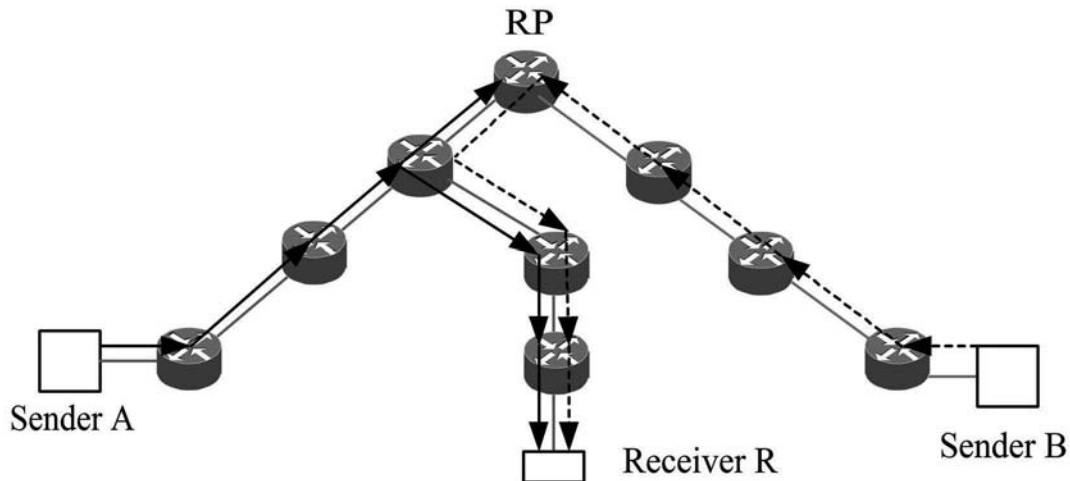


Figure 4-27: Packet forwarding by PIM-SM through shared tree and RP

PIM Spare Mode (PIM-SM) protocol is based on shared tree and rendezvous point (RP). The basis principle of this protocol is that a multicast sender sends multicast stream to the RP, which then forwards this traffic to the active receivers through the shared tree. In PIM-SM, this shared tree is rooted at the selected router called RP and used for all sources sending to the multicast group. This principle is illustrated in figure 4-27 in which the sender A and B send multicast traffic to the RP that again sends this traffic to the active receiver R. In order to send the first data to RP, the sources send it per multicasts to the designed router (DR), which then encapsulates data in PIM-SM control messages and send it by unicast to the RP. Based on this first data, a shared tree for each multicast group is then built, so that the DR sends multicast datagrams via multicast and does not need to encapsulate them and send them per unicast communication.

Each PIM protocol uses an underlying topology-gathering protocol to populate the so-called multicast routing information base (MRIB). This MRIB can be determined directly from the unicast routing table. The primary role of the MRIB is to determine the next hop router along a multicast shared tree for each destination subnet. Furthermore, the MRIB is used to define the next-hop

(b) Registering

A source starts sending data destined for a multicast group. The local DR takes this data and encapsulates in unicast PIM register packets and sends them to RP. The RP receives these packets, decapsulates them, and sends them into the RP tree (RPT) built in the previous step. These packets then reach all receivers for that multicast group. The process of encapsulating the multicast packets to the RP is called registering, and the encapsulation packets are called PIM register packets. This registering process is illustrated as a UML sequence diagram in the figure 4-28.

(c) Register-stop.

Encapsulating the packets at the DR, sending them to the RP and decapsulating those at the RP may result in expensive operations for a router that performs these operations. Moreover, sending encapsulated packets to RP, and then sending them back down the shared tree may result in the packet traveling a long distance to reach receivers which may be closer to the sender than the RP. To solve this problem, RP will switch from the registering phase to the native multicast forwarding. To do it, when RP receives a PIM register packet from the source S to the group G, it initiates an (S, G) source specific Join toward S. This Join message is forwarded hop-by-hop toward S, and instantiates a (S, G) multicast shared tree state in the routers along the path. This (S,G) tree is then used to deliver packets for group G if these packets are generated from source S. When Join message reaches S's subnet, the routers along the path all have (S, G) multicast tree state. Therefore, packets from S start to travel following the (S,G) tree toward the RP. When packets from S begin to arrive as natively multicast packets at RP, the RP will receive two copies of each multicast packets (one from PIM register packet and one from natively multicast packet). At this point, the RP starts to discard the encapsulated copy of these packets and send a Register-stop message back to S's DR. By receiving the Register-stop message, the DR stops encapsulating the multicast packets sent from S. This process is called register-stop and illustrated in figure 4-28 b).

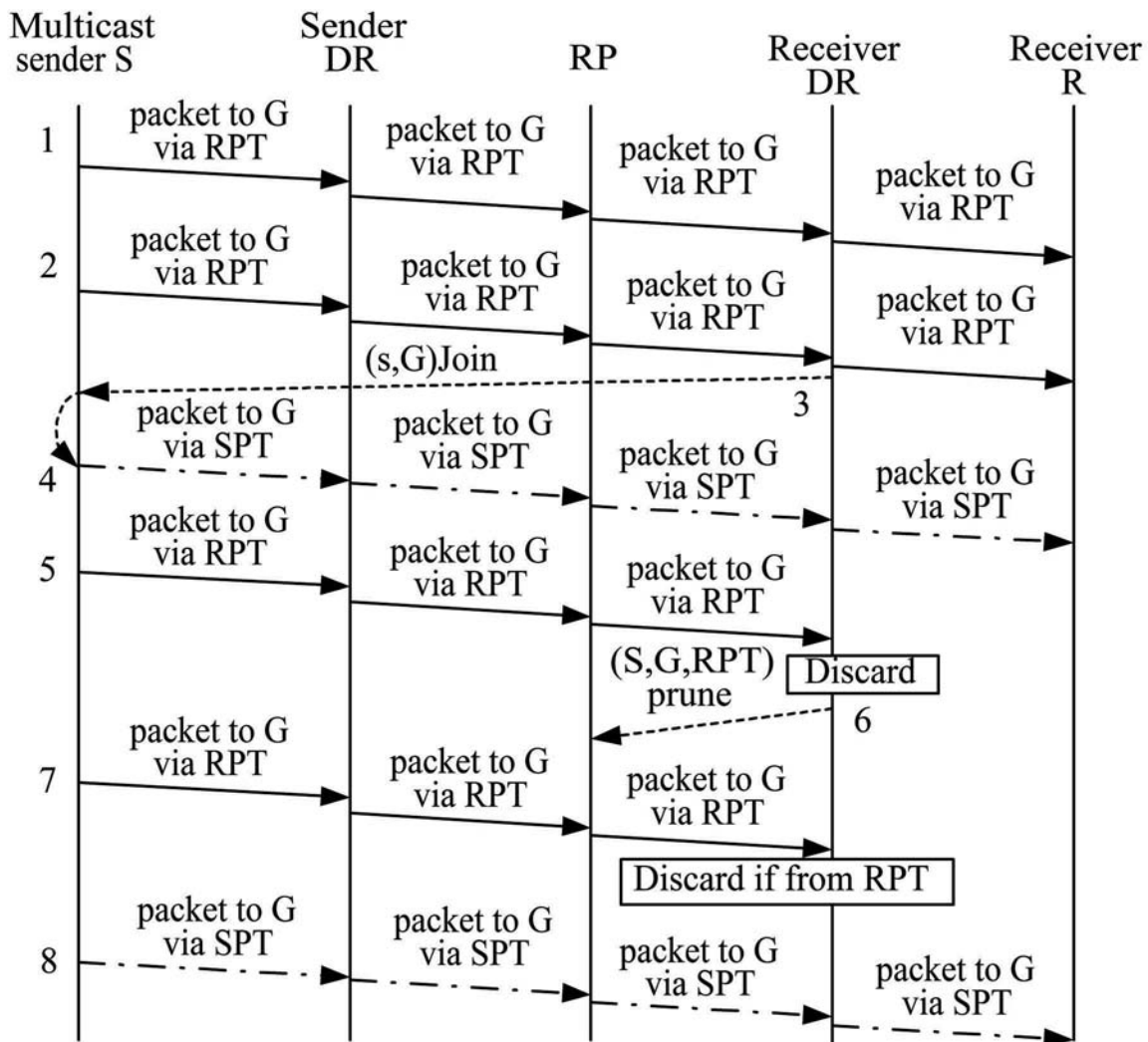


Figure 4-29: UML sequence diagram for switching from RPT to SPT

(d) Phase shortest path tree

For many multicast receivers, the path via RP may be longer than the shortest path from the source to the receiver. To improve the network efficiency and latency, the DR on the receiver's subnet (typical LAN) may initiate a transfer from the shared tree (the RP tree above, RPT) to a source-specific shortest path tree (SPT). To do this, the DR sends an (S, G) Join message towards the source S that instantiates the states in the routers along the path to S. When this Join message arrives at S, the shortest path tree (SPT) from the source to the receiver for the multicast group G is built and is also used to forward the multicast packets of the group G toward receivers. At this point, the DR at the receiver's subnet will receive two copies of the data: one from SPT and one from the RPT. When the DR (or an upstream router) receives the first packet from the SPT, the DR (or a upstream router) starts dropping packets sent from S for group G that

arrive via the RPT. The DR (or an upstream router) additionally sends an (S, G) Prune toward the RP. This prune message is forwarded hop-by-hop, instantiating the state in routers along the path toward RP and indicating that traffic from S to G should not be forwarded in this direction [FHH-2006]. This Prune message is propagated until it reaches the RP or a router that still needs the traffic from S. When this prune message reaches the RP, the multicast traffic from the RP tree still arrives at the RP but the RP does not forward this traffic to the subnet of the receiver S. The switch from RPT to SPT is shown in figure 4-29.

4.4.6.3.3 PIM Dense Mode

PIM Dense mode (PIM-DM) is designed with an opposite assumption to PIM-SM. Namely that the multicast receivers for any multicast group are densely distributed through the network. For using the PIM-DM, it is assumed that most subnets have expressed an interest in receiving any given multicast traffic. The development of PIM-DM has paralleled that of PIM-SM. Version 1 was created in 1995 and is now considered obsolete. But this version is still supported by Cisco and Jupiter routers. PIM-DM version 2 was created in 1998, but was never standardized. The actual PIM-DM protocol is specified in RFC 3973 [ANS-2005], which is summarized in this section.

PIM-DM differs from PIM-SM in two fundamental features: 1) PIM-DM only uses source-based tree through explicitly triggered prunes and grafts, and no periodic join messages are transmitted. 2) There is no Rendezvous Point (RP). These features make PIM-DM simpler than PIM-SM to implement and deploy. PIM-DM is an efficient protocol when most receivers are interested in the multicast traffic, but not scale well in a large network in which most receivers are not interested in receiving the multicast data. Each PIM-DM protocol implements source-based tree, reverse path forwarding (RPF), pruning and grafting. By this protocol, the multicast traffic is initially sent to all hosts in the network, and the routers that do not have any receiver hosts then send PIM-DM prune messages to remove themselves from the tree. Main functions of a PIM-DM protocol are: (a) maintaining the state of all source-based trees; (b) determining packet forwarding rules; (c) detecting other PIM routers in the domains; (d) issuing and processing prune, graft and join messages; (e) refreshing the state of all source-based trees. These functions are described in [ANS-2005] and summarized as follows.

(a) Maintaining the state of all source-based tree

The protocol state describing the multicast route and the state information associated with each pair of source S and group G is stored in the so called tree

information base (TIB). This TIB holds the state of all multicast source-based trees and thus it must be dynamically maintained as long as any timer associated with that (S, G) entry is active. To do that, each router stores the non-group-specific state for each interface and the neighbor state for each neighbor in its TIB. Furthermore, each router stores the (S, G) state for each interface. For each interface, the (S, G) state involves the local membership information, the (S, G) prune state and the assert winner state. Each router also stores the graft/prune state and the originate state of the upstream interface-specific. Using the state defined in the TIB, a set of macros is defined for each router. These macros can be used for the following purposes:

- Describing the outgoing interface list for relevant states,
- Indicating the interfaces to which traffic might or might not be forwarded,
- Returning the reverse path forwarding (RPF) for each source S,
- Discovering the members on a given interface.

(b) Packet forwarding rules

Multicast packet delivering is performed at each PIM-DM router by using the packet forwarding rules specified in pseudo code [ANS-2005]. According to the rules in this pseudo code, a router first performs RPF check for each incoming multicast packet to determine whether the packet should be forwarded. If the RPF check has been passed, the router constructs an outgoing interface list for the packet. If this list is not empty, the router forwards this packet to all listed interfaces. If the list is empty, then the router will issue a prune message for the pair (S, G).

(c) Detecting other PIM routers in the domains

A detection of other PIM routers is done through generating and processing the hello messages, which are periodically sent on each PIM enable interface. When a hello packet is received at a router, this router records the receiving interface, the sender and the information contained in the hello message and retains this information for a given hold time in its TIB. The hello messages are also used at routers to dynamically update the tree information base (TIB) and the multicast forwarding information base (MFIB).

(d) Issuing and processing prune, graft and join messages

Prune messages are sent toward the upstream neighbours for a source S to indicate that traffic from this source addressed to a group G is not desired. When a router wishes to continue receiving multicast traffic, a join message is sent from this router to its upstream routers. Finally, a graft message is sent to re-join a previously pruned branch to the multicast delivery tree. These

messages can be sent from or received by a PIM-DM router. The sending and receiving process are described below.

- **Sending prune, graft and join messages.** For each source S and a multicast group G , the upstream (S, G) interface state machine for sending prune, graft and join at each PIM-DM router is shown in figure 4-30. There are three states: forwarding, pruned and AckPending. Forwarding is the starting state of the upstream (S, G) state machine. The router is in this state if it just started or if the outgoing interface list ($olist(S, G)$) is not empty. The router goes into the pruned state if the $olist(S, G)$ is empty and the router stops to forward the traffic from S addressed to the group G . If the $olist$ is not empty, then the router moves from Pruned state to the AckPending state, sending a graft message to indicate the traffic from S addressed to G should again be forwarded. The router stays in this AckPending state if it has sent a graft message but does not receive a Graft-Ack message. By receiving the Graft-Ack, or a state refresh or a direct connect message, the router goes to the Forwarding state.

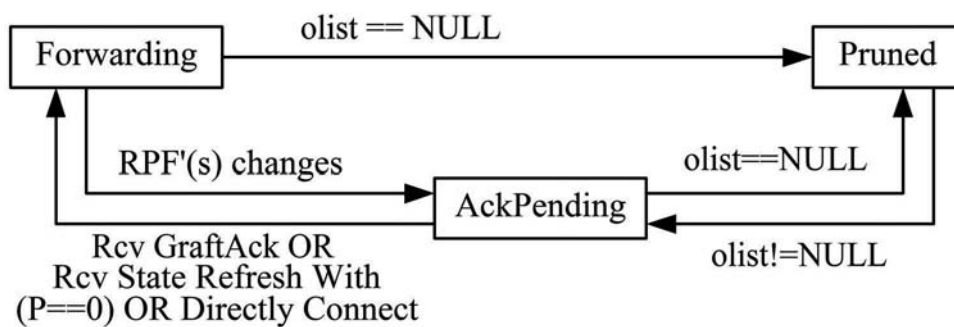


Figure 4-30: Upstream interface state machine [ANS-2005]

- **Receiving prune, graft and join messages.** For each source S and multicast group G , the downstream (S, G) interface state machine at each router is described in figure 4-31 below. This state machine contains three states: NoInfo, PrunePending and Pruned. The router is in the NoInfo state if it has no prune state for (S, G) , and neither the prune timer nor the PrunePending timer is running. The router moves from this state into the PrunePending state if it receives a prune message. The router stays in this state when it is waiting to see whether other downstream router will override the prune. The router moves from prunePending state to the pruned state and stays there until it receives join/graft messages or the prune timer expires.

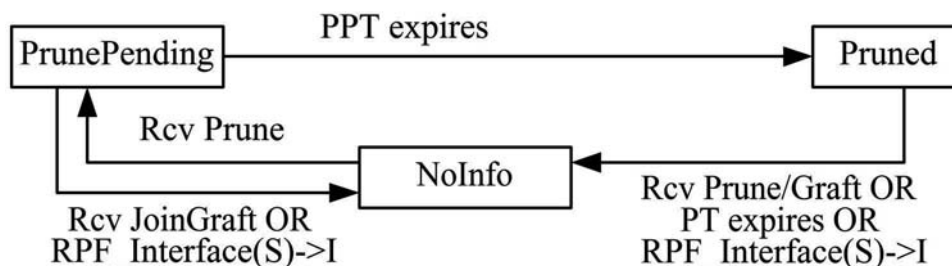


Figure 4-31: Downstream interface state machine [ANS-2005]

(e) Refreshing the state of all source-based trees

For each source-based tree (S, G), the refresh messages are generated periodically by a PIM-DM router that directly connect to the source S. The controlling of this refresh message is done via two timers: the state refresh timer and the source active timer. By receiving the refresh messages the router refresh the multicast delivery tree and forwards the message using the rules defined in [ANS-2005].

4.4.7 Summary

The Internet network layer protocols described in this section and their mechanisms are summarized in the table 4.4 below. Mechanisms, which are not described in this section, are founded in the following sections:

- Bit error control and packet error control: section 3.1
- Classification of routing protocol and mechanisms: section 3.7.
- Queuing and packet scheduling mechanisms: section 3.4
- Active queue management mechanisms: section 3.6.

Protocol mechanisms		IPv4	IPv6	RIP	OSPF	BGP	DVMRP	MOSPF	PIM-SM	PIM-DM
Addressing	MAC									
	IP	x	x	x	x	x	X	x	x	x
	Port					x				
Connection management	connectionless	x	x	x	x		X	x	x	x
	connection-oriented					x				
Multiplexing/demultiplexing		x	x							
Encapsulation/decapsulation		x	x							
Bit error control		x			x		x	x	x	x
Packet error control						x				
Queuing/Packet scheduling		x	x							
Active queue management		x	x							
Explicit Congestion Notification ECN		x	x							
Packet switching		x	x							
Authentication			x		x	x	x	x	x	x
Multiple higher layer protocol support		x	x							
Fragmentation/Reassembly	at end hosts	x	x							
	at routers	x								
Unreliable service		x	x	x	x		x	x	x	x
Reliable service						x				
Routing	Unicast			x	x	x				
	Multicast						x	x	x	x
	Distance vector routing			x			x			
	Link state routing				x			x		
	Path routing					x				
	Flooding									
	Shared trees								x	
	Source-based tree						x			x
	Reverse path forwarding						x			
	Pruning and Grafting						x			
	Joint								x	x
	Rendezvous point								x	

Table 4.4: Selected Internet network layer protocols and their mechanisms

4.5 Transport Layer

In the previous session we learned that the network layer offers services that allow logical communications between network devices, such as host and

routers. To provide logical communications between the application processes running on different hosts, services at the transport layer need to be determined. Application processes use logical communications to send messages to each other and to receive messages from each other, without knowing of detail of the underlying infrastructure used to transmit these messages (figure 4-32).

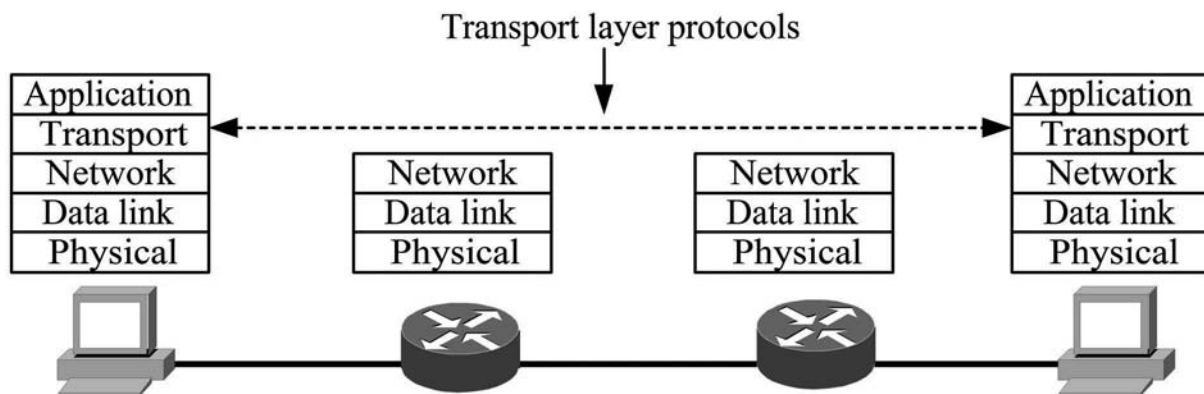


Figure 4-32: The transport layer

Thus, the job of the transport layer is to provide services that enable the logical communication between application processes. The transport layer at the sending side encapsulates the messages it receives from application layer into transport layer protocol data units (T-PDUs), passing them to the network layer protocol instance. On the receiving side, it received the T-PDUs from the network layer, removes the transport header from these PDUs, reassembles the messages and passes them to the appropriate receiving application processes.

This chapter describes the fundamental transport layer services and selected transport layer protocols for shifting packets from the application layer of the source to the application layer of the destination. We will see that unlike the network layer that provides logical communication between the hosts, the transport layer offer logical communications between processes running on these hosts.

4.5.1 Transport Layer Services

The complexity of the transport layer services depends on the one hand on the services provided by the Internet's network layer, and on the other hand on the services the application layer protocols need. As illustrated in the last section, the Internet network layer only provides connection less and unreliable transport services. But not all Internet applications only need these connectionless and unreliable transport services offered by the network layer, several applications

needs connection oriented services, e.g. Email- or FTP-application. Moreover numerous applications require reliable transport services, such as web-applications and email-applications. In addition, real-time audio/video applications need the real-time services that can guarantee the timing, the bandwidth and the data loss. The services, which are not provided by the Internet network layer, must be made available in the transport layer or in the application layer. The transport layer provides following services to the application layer:

- Addressing
- Multiplexing and demultiplexing
- Unreliable service and reliable service
- Connection less service and connection-oriented service
- Error control
- Flow control and congestion control

4.5.1.1 Addressing

Several application layer protocols may use a same transport layer protocol, for example http and ftp both use the transport protocol TCP. In order to correctly deliver transport layer segments to their corresponding application processes, each transport layer protocol must be able to address each segment when sending it. The addressing for the transport layer segment is performed via the so called source port number and destination port number in the header of each transport layer segment, while a port number is a 16-bit integer. The source port number and destination port number are analogous to the source address and destination address in the IP header, but at a higher level of detail. The source port number identifies the originating process on the source machine, and the destination port identifies destination process on the destination machine.

In comparison with the network layer address that identifies a host, the transport layer address identifies a user process running on a host.

4.5.1.2 Multiplexing and demultiplexing

At the destination host, the transport layer receives segments from the network layer just below. The transport layer has the responsibility of delivering the data in these segments to the appropriate application process running in this host.

Each transport-layer segment has a field that contains information that is used to determine the process to which the segment's data is to be delivered. At the receiving end, the transport layer can then examine this field to determine the receiving process, and then directs the segment to that process. This job of delivering the data in a transport-layer segment to the correct application process

is called demultiplexing. The job of gathering data at the source host from different application processes, enveloping the data with header information to create segments, and passing the segments to the network layer is called multiplexing.

UDP and TCP perform the demultiplexing and multiplexing jobs by including two special fields in the segment headers: the source port number field and the destination port number field. These fields contain information used to indicate the process from which the segment was sent and to which the segment's data is to be delivered. At the receiving end, the transport layer can then examine the destination port number field to determine the receiving process, and then direct the segment to that process.

4.5.1.3 Unreliable service and reliable service

While the Internet network layer only provides unreliable service, the transport layer protocols support both unreliable transport service and reliable transport service. The reason for this is because different applications need different reliability, e.g. the internet telephony and the video conference need the unreliable delivery, while the electronic mail and the web need the reliable service

By unreliable transport service we mean that the transport layer does not guarantee the handling of segment duplication, segment loss, corruption of data and delayed or out-of-order delivery. This unreliable service is provided by the transport protocol UDP (User Datagram Protocol), which will be discussed in the next section.

The reliable service enables the transport layer to deliver all data sent without error. The reliable transport service is implemented in the TCP (Transmission Control Protocol) that will be addresses in the next section.

4.5.1.4 Connection-oriented service

In general, transport protocols can be characterized as being either connection-oriented or connectionless. Connection-oriented services must first establish a connection with the desired service before passing any data. A connectionless service can send the data without any need to establish a connection.

If reliability is paramount, then connection-oriented transport services (COTS) is the better choice. For protocols with unreliable transport service, only connection less transport service is required.

The connection-oriented transport service enables the communication processes to perform handshaking to set up an end-to-end connection. The

handshaking process may be as simple as synchronization or as complex as negotiating communications parameters. To negotiate a connection, both sides must be able to communicate with each other. This will not work in a unidirectional environment. In general, connection-oriented services provide some level of delivery guarantee, whereas connectionless services do not.

4.5.1.5 Connectionless service

Connectionless service means that no effort is made to set up a dedicated end-to-end connection before sending the data. Connectionless communication is usually achieved by transmitting information in one direction, from source to destination without checking to see if the destination is still there, or if it is prepared to receive the information.

While the connection-oriented service is provided by TCP, the connectionless transport service is implemented in the UDP protocol.

4.5.1.6 Error control

All transport layer protocols provide the bit error recognitions by using the Internet checksum method described in section 3.1.1. But only transport layer protocols providing the reliable services such as TCP support the error correction at the packet level error control addressed in section 3.1.2.

4.5.1.7 Congestion control

Congestion control issue is generally addressed in section 3.5. The design of congestion controls at the transport layer depends on types of the user applications. Different applications have different needs for congestion controls. Elastic applications, such as email or ftp, need faster transfer but do not care about the actual sending rate. These applications require the feedback-based and window-based congestion control described in the section 3.5.1. In comparison to elastic applications, the inelastic applications, such as Internet telephony or video conference, must be able to transmit data at a certain rate in order to be effective and care about the actual sending rate. Because of this, instead of feedback-based and window-based congestion control, the rate-based controls discussed in 3.5.1 are suitable for the inelastic applications.

4.5.2 Transport Layer Protocols

In this section the transport protocols TCP and UDP will be illustrated. Other transport protocols (RTP, RTCP, SCTP, DCCP) for audio and video applications are demonstrated in section 3.12.

4.5.2.1 User Datagram Protocol

The User Datagram Protocol (UDP) is a unreliable and connectionless transport protocol, which is defined in RFC 768. In this section, the segment format and the fundamental protocol mechanism of UDP as well as its application will be illustrated.

4.5.2.1.1 UDP Segment Format

The header format of a UDP datagram is illustrated in figure 4-33 below.

Source port number (16 bits)	Destination port number (16 bits)
Length (16 bits)	Checksum (16 bits)
Application data (if any)	

Figure 4.33: The header format of a UDP segment

The UDP header consists of four fields

- *Source Port number and destination port number.* These fields are used to address the UDP packets for delivering them to a given application process. A UDP sender uses the source port as a service access point (SAP) to indicate the application process on the local sender that originated the packet. It also uses the destination port as a service access point to indicate the service required from the remote receiver. UDP packets from a UDP receiver carry the UDP sender SAP in these port number fields.
- *Length.* This field specifies the UDP segment length (in bytes) including the header and the payload.
- *Checksum.* This field indicates the Internet checksum calculated over the header of a UDP datagram. The checksum field is used to verify that the end to end data has not been corrupted by routers or bridges in the network or by the processing in an end system. The algorithm to compute the checksum is the Standard Internet Checksum algorithm described in section 3.1. This allows the receiver to verify that it was the intended destination of the packet, because it covers the IP addresses, port numbers and protocol number.

4.5.2.1.2 UDP Protocol Mechanisms

UDP is a very simple protocol that provides following services:

- Addressing
- Connectionless and unreliable service

- Multiplexing and demultiplexing
- Bit error control.

Addressing, multiplexing and demultiplexing mechanisms of the UDP are the same as described section 4.5.1. The connectionless, unreliable and error control service for the UDP will be described in more detail in this section.

4.5.2.1.2.1 Connectionless and Unreliable Service

With UDP as the transport protocol, there is no initial handshaking phase between sending and receiving transport layer instances before sending a UDP segment. The UDP operation principle is illustrated in figure 4-34.

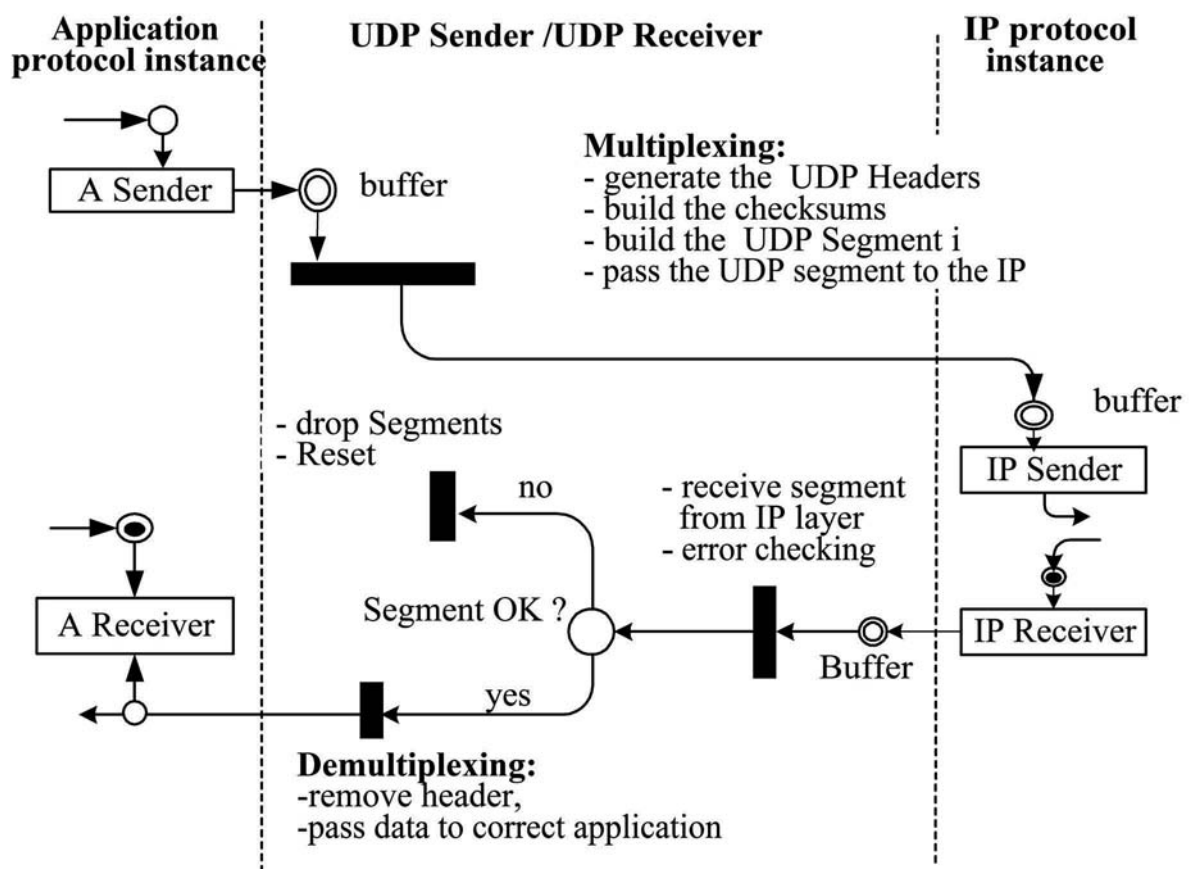


Figure 4.34: A simple modelling of the UDP protocol

UDP simply takes messages from the application process, attaching source and destination port number fields, adding the length and the checksum fields, and passing the resulting UDP segment to the Internet network layer. The Internet network layer encapsulates the UDP segment into an IP datagram and uses its services to deliver this segment to the destination host. If the UDP segment arrives at its destination host, UDP uses the destination port number to deliver the segment to correct application process.

Because of the unreliable service, the UDP does not guarantee the handling of segment duplication, segment loss, corruption of data and delayed or out-of-order delivery. Thus, the UDP segments may be lost or may be delivered out of order to the applications.

4.5.2.1.2.2 Bit error control

The bit error control is performed via the Internet checksum. But the UDP only provides the error detection; it does not do any thing to recover the error. Some UDP implementations simply discard the damaged segment (see figure 4-34); other implementations pass the damaged segment to the application with a warning.

4.5.2.1.3 Applications of UDP

UDP is useful for applications that prefer timeliness to reliability, such as Voice-over-IP, video streaming, conferencing and broadcasting. Also traditional data applications use the UDP for data transport, such as DNS (Domain Name Server), BOOTP (Bootstrap Protocol), DHCP (Dynamic Host Configuration Protocol), SNMP (Simple Network Management Protocol), RIP (Routing Information Protocol) and NFS (Network File System).

4.5.2.2 Transmission Control Protocol

The Transmission Control Protocol (TCP) is a reliable and connection-oriented transport protocol that is defined and specified in RFC 793, RFC 1122, RFC 1323, RFC 2018 and RFC 2581.

Like UDP, TCP provides addressing, multiplexing and demultiplexing and bit error control. In contrast, the TCP additionally provides the connection-oriented, reliable transport services, the packet error control and the congestion control.

4.5.2.2.1 TCP Segment Format

A TCP segment consists of a TCP header and the TCP payload containing the user data created at the application layer. The TCP header format is illustrated in figure 4-35.

- *Source Port number and destination port number.* Like the UDP, TCP also uses these fields to identify the sender and the receiver application process. These port numbers are also necessary needed for multiplexing and demultiplexing discussed in section 4.5.1.
- *Sequence number and acknowledgement number.* These fields are used for implementing a reliable data-transfer service. While the sequence number identifies the byte in the streams of data from a TCP sender process to a

TCP receiver process, the acknowledgement contains the next sequence number that the TCP sender of this acknowledgement expects.

- *Header length.* This field specifies the length of the TCP header in 32-bit word.
- *Window size.* The window size field indicates the number of bytes that a TCP receiver is willing to accept. This field is used for the congestion control in the TCP.
- *Checksum.* This field indicates the Internet checksum calculated over the header of a TCP datagram. The checksum field is used to verify that the end to end data has not been corrupted by routers or bridges in the network or by the processing in an end system. The algorithm to compute the checksum is the Standard Internet Checksum algorithm described in section 3.1. This allows the receiver to verify that it was the intended destination of the packet, because it covers the IP addresses, port numbers and protocol number.
- *Flag field.* This field contains 6 bits: URG, ACK, PSH, RST, SYN und FIN. The acknowledgement bit (ACK) is used to indicate that the value carried in acknowledgement field of this TCP header is valid. If the push bit (PSH) is set, the data must be pushed up to the application layer process immediately. The reset flag is used to reset the TCP connection. The synthesis bit (SYN) and finish bit (FIN) are used to setup and to teardown the TCP connection, as we will discuss in this section. The urgent bit (URG) specifies that the sending-side upper layer entity has marked the data in this TCP segment as “urgent”.
- *Urgent Pointer.* This field specifies that the TCP must inform the upper-layer entity when urgent data exists and passes the data to the application process quickly.
- *Options.* The options field is specified to use in future.

Source Port (16 bits)					Destination Port (16 bits)				
Sequence Number									
Acknowledgement Number									
Header length	Reserved	URG	ACK	PSH	RST	SYN	FIN	Window	
Checksum					Urgent Pointer				
Options (if created)									

Figure 4-35: The TCP segment format

4.5.2.2.2 TCP Protocol Mechanisms

Like UDP, TCP provides addressing, multiplexing and demultiplexing and bit error control. In contrast, the TCP additionally provides the connection-oriented, reliable transport services, the packet error control and the congestion control.

Addressing, multiplexing and demultiplexing mechanisms of the TCP are the same as described in section 4.5.1. Therefore, in this section, only the following protocol mechanisms will be detailed:

- TCP connection-oriented Service
- TCP Reliable transport service
- TCP Error Control (Bit error control and packet error control)
- TCP Congestion Control
- TCP Time Management

4.5.2.2.2.1 Connection-Oriented Services

Connection oriented services require that a logical connection must be established between two devices before transferring data between them. This is generally accomplished by following a specific set of rules that specify how a connection should be initiated, negotiated, managed and eventually terminated. Usually one device begins by sending a request to open a connection, and the other responds. They pass control information to determine if and how the connection should be set up. If this is successful, data is sent between the devices. When they are finished, the connection is broken.

The TCP Connection-oriented service involves two phases: connection establishment and connection termination.

- *Connection establishment.* TCP uses a three-way handshaking procedure to establish a connection. A connection is established when the initiating side sends a TCP segment with the SYN bit set and a proposed initial sequence number in the sequence number field (i in the figure 4-36). The receiver then returns a segment (segment 2 in figure 4-36) with both the SYN and the ACK bits set. In this second segment, the sequence number field is set to its own assigned value for the reverse direction (j in the figure 4-36) and the acknowledgement number field is set equal to the sequence number in the first segment plus 1 – it is the next sequence number that the TCP instance at the `www.ira.uka.de` expects. On receipt of this, the initiating side returns a segment (segment 3 in figure 4-36) with just the ACK and SYN bits set and the acknowledgement field is set equal to the sequence number in the second segment plus 1. Figure 4-36 illustrates a TCP connection setup example between

mai.hpi.uni-potsdam.de and www.ira.uka.de, whereby the initiating side is mai.hpi.uni-potsdam.de and www.ira.uka.de is a web server.

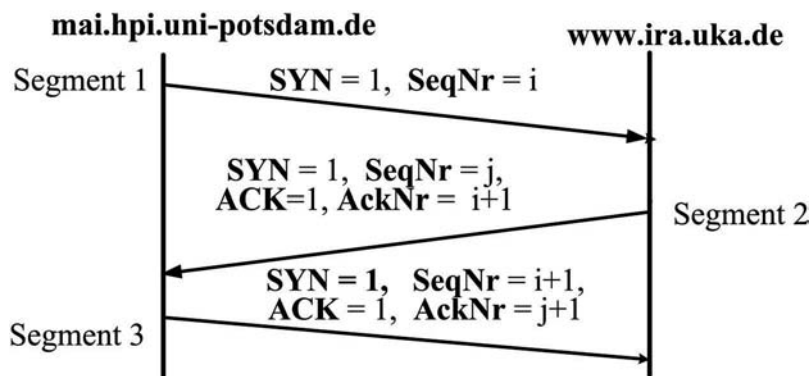


Figure 4-36: TCP connection setup via three-way handshaking

- Connection termination.** The TCP connection termination is performed via four-way handshaking. When the user has sent all its data and wishes to close the connection, it sends a CLOSE primitive to TCP protocol instance which then sends a segment (segment 1 in figure 4-37) with the FIN bit set. The sequence number k of this segment indicates the byte in the data streams from a TCP sender process to a TCP receiver process. On receipt of this, the peer TCP issues a CLOSING primitive flag to the user and returns an ACK segment (segment 2 in figure 4-37) with FIN bit set. In this second segment, the acknowledgement number ($k+1$) specifies the next sequence number that the TCP protocol instance at www.ira.uka.de expects. If the peer TCP user has finished send all its data it sends a CLOSE primitive. If the user still has some data to send, it sends the data in a segment (segment 3 in figure 4-37) with the sequence number l and the FIN bit set. On receipt of this segment, the initiating TCP issues a TERMINATE primitive to the user and returns an acknowledgement (segment 4 in figure 4-37) for the data just received. When the peer TCP receives the ACK, it returns a TERMINATE primitive to the user. In the case where the user wants to abort the connection, the user can send an ABORT primitive to the TCP entity which sends a segment with the RST flag set. On receipt of this segment, the peer TCP closes both sides of the connection and issues a TERMINATE primitive with a reason code to the peer TCP user.

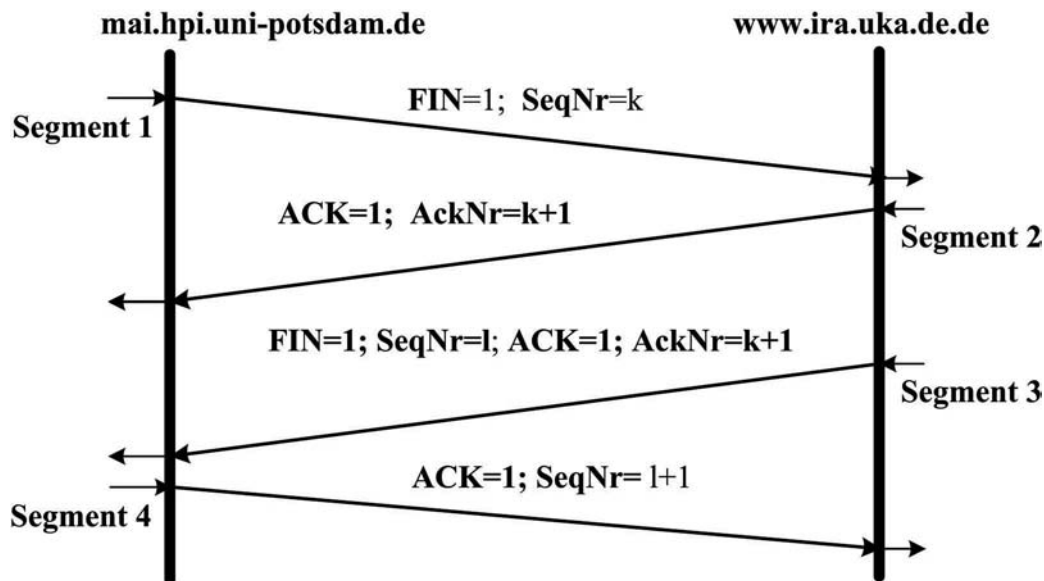


Figure 4-37: TCP connection termination via four-way handshaking

4.5.2.2.2 Reliable Transport Service

The TCP reliable data transmission is performed via sequence numbering, acknowledgement and packet retransmission that illustrated in 3.1.2.1, 3.1.2.2 and 3.1.2.4. During data transfer, the TCP connection-oriented services will monitor the lost packets in order to regulate its transmission rate. This is performed via the TCP congestion control discussed in 3.5.2.

4.5.2.2.3 TCP Error Control

TCP provides both bit error control and packet error control. Like UDP, TCP also uses the Internet checksum method to detect the bit error. The Checksum field is found in the TCP segment header. The TCP bit error correction is carried out via the retransmission mechanism of the TCP. In particularly, TCP recognizes a TCP segment with the bit error as a packet error. It drops the segment and tries to resend this segment in the next transmission.

TCP detects the packet error by using the sequence numbering and the acknowledgement numbering mechanism for each TCP segment. Also timers are used to detect packet losses. The packet error correction is performed via the TCP retransmission. Detail about mechanisms for bit error control and packet error control is found in 3.1.

4.5.2.2.2.4 TCP Time Management

The TCP time management is used in both connection management (connection setup and teardown) and the data transfer phase. TCP maintains seven timers for each TCP connection [RS-1994]:

- *Connection establishment timer.* This timer starts when a SYN segment is sent to setup a new connection. If the initiator of the SYN segment doesn't receive an ACK within a predefined timeout value (default is set to 75 seconds), the connection establishment is aborted.
- *Retransmission timer.* The timer is set when TCP sends a data segment. If the other end does not acknowledge the data segment when this timer expires, the TCP retransmits the data. The retransmission timer is dynamically calculated based on the round-trip time.
- *Persist timer.* The timer is set when the other end of a connection advertises a zero window but it still has data to send. In this case, the sender uses a persist timer to query the receiver to see if the window has been increased.
- *Keepalive timer.* This time enables a TCP side (e.g. server) to know whether the other side (e.g. client) has either crashed and is down, or crashed and rebooted. If the connection is idle for 2 hours, the keepalive timer expires and a special segment is sent to the other end. If other end is down, the sender will receive a RESET and the connection will be closed. If there is a segment exchange during 2 hours, the keepalive timer is set to 2 hours again.
- *Reconnection timer.* This timer is set when TCP sends data. If the other end does not acknowledge the data when the reconnection timer expires, TCP retransmits the data. This timer is calculated dynamically based on the RTT (round-trip time).
- *Delayed ACK timer.* This timer is set when TCP receives data that must be acknowledged but need not be acknowledged immediately.
- *FIN_WAIT_2 timer.* As illustrated in the figure 4-37 for the TCP connection termination, the server sends the client a segment with a "FIN" bit set (segment 1 in figure 4-37). The client gets the FIN segment and goes into CLOSE_WAIT state, and sends an acknowledgment segment back to the server. When the server gets that acknowledgement segment (segment 2 in figure 4-37), it goes into FIN_WAIT_1. If the server receives the FIN segment (segment 3 in figure 4-37) from client, it enters FIN_WAIT_2 state. A FIN_WAIT_2 timer is started when there is a transition from the FIN_WAIT_1 state to the FIN_WAIT_2 state. The value of this timer is 10 minutes. A TCP segment with a FIN bit set is

expected in the `FIN_WAIT_2` state. If a packet with a FIN bit set is received, the timer is cancelled. On expiration of the timer, it is restarted with a value of 75 seconds. The connection is dropped if no packet with the FIN bit arrives within this period.

4.5.2.2.4 TCP Congestion Control and Explicit Congestion Notification

The TCP congestion control may include 4 mechanisms: slow start, congestion avoidance, fast retransmit and fast recovery. These algorithms and how the TCP congestion control works are discussed in 3.5.2. While the TCP congestion control operates at the end hosts, the Explicit Congestion Notification (ECN) operates at the routers by using the active queue management and at the end hosts by using the TCP congestion control. The ECN is illustrated in 3.5.3.

4.5.2.2.3 TCP Implementations

The most popular TCP implementations are TCP Tahoe, TCP Reno and TCP Sack [APS-1999, MMF-1996, PF-2001]. These TCP implementations only differ in their congestion controls.

- *TCP Tahoe*. TCP Tahoe supports slow start, congestion avoidance and fast retransmit for the congestion control.
- *TCP Reno*. Reno adds fast recovery mechanism to the TCP Tahoe.
- *TCP Sack*. The Sack adds the selective acknowledgement to the Reno. The disadvantage of the Reno is that, when there are multiple losses, it can retransmit only one lost segment per round-trip-time. The selective acknowledgement of the TCP SACK enables the receiver to give more information to sender about the received packets. This allows sender to recover from multiple packet losses faster and more efficiently.

4.5.2.2.4 Applications of the TCP

Applications, such as Simple Mail Transfer Protocol (SMTP) used by electronic mail, file transfer protocol (FTP), the Hypertext Transfer Protocol (HTTP) used by the World Wide Web (WWW), remote host access, web document transfer and financial applications require fully reliable data transfer, that is, no data loss. Such applications use TCP as their transport protocol. These applications don't care about loss of a small amount of performance to overhead. For example, most applications that transfer files or important data between machines use TCP, because loss of any portion of the file renders the entire thing useless.

4.5.3 Summary

The transport layer protocol described in this section and their mechanisms are summarized in the table 4.5 below. Mechanisms, which are not described in this section, are founded in the following sections:

- Bit error control and packet error control: section 3.1
- TCP congestion control: 3.5.2
- Explicit congestion Notification (ECN): 3.5.3

Protocol mechanisms		Protocols	
		UDP	TCP
Addressing	MAC		
	IP		
	Port	x	x
Connection management	Connectionless	x	
	Connection-oriented		x
Multiplexing/demultiplexing		x	x
Encapsulation/decapsulation		x	x
Bit error control	Error detection	x	x
	Error recovery		x
Packet error control	Error detection		x
	Error correction		x
TCP Congestion control			x
Explicit Congestion Notification (ECN)			x
Multiple higher layer protocol support		x	x
Unreliable service		x	
Reliable service			x
Time management			x

Table 4.5: Transport layer protocols and their mechanisms

4.6 Application Layer

In the previous section we illustrated several aspects of the Transport Layer and its protocols. We learned a great deal of information; covering transport services, protocol mechanisms, UDP and TCP. Now we will take a look at the fourth and final layer of the TCP/IP stack: the Application Layer.

A lot of newcomers to TCP/IP wonder why an Application Layer is needed, since the Transport Layer handles a lot of interfacing between the network and applications. While this is true, the Application Layer focuses more on network services, APIs, utilities, and operating system environments. If you know the TCP/IP stack and OSI model well enough, you'll know that there are three OSI model layers (application layer, presentation layer and session layer) that

correspond to the TCP/IP Application Layer. The OSI equivalent to the TCP/IP application layer is described as follows:

- *OSI Application Layer.* Network access and providing services for user applications are supported at this layer.
- *OSI Presentation Layer.* This layer is responsible to translate data into a format that can be read by many platforms. With different operating systems, programs, and protocols floating around, this is a good feature to have. It also has support for security encryption and data compression.
- *OSI Session Layer.* The function of this layer is to manage the communication between applications on a network, and is usually used particularly for streaming media or using web conferencing.

Thus, the job of the application layer is to enable the services that provided by OSI/ISO application, presentation and session layer. Before going to discuss the application layer services and protocols, it is important to explain following basic terms:

- *Application layer protocols and Network applications.* A network application consists of many interacting software components running at processes, which are distributed among two or more hosts and communicate with each other by exchanging messages across the Internet. An application layer protocol is only one component of a network application. For example, the web is a network application consisting of several components: web browsers, web servers, a standard for document format such as HTML and the application layer protocol HTTP [FGM-1999], which defines the message formats exchanged between browser and web server and the actions taken by the browser and web server by sending and receiving these HTTP messages.
- *Clients and servers.* A network application protocol has typically two sides, client and server. The client initiates contact with the server. The server provides requested service to client via replies. Let's look at the web application discussed above. A web browser implements the client part of the HTTP and the web server implements the server part of the HTTP.
- *Processes and process communication.* A process is a program running within a network device (e.g. end host). While two processes within the same host communicate with each other using inter-process communication defined by the operating system, processes running on different hosts communicate with each other by using an application layer protocol. An application involves two or more processes running in different hosts that communicate with each other over a network. These processes communicate with each other by sending and receiving messages through their sockets.

- *Sockets.* A socket is an interface between application process and the underlying transport protocol (figure 4-38). Two processes communicate with each other by sending data into socket and reading data out of socket.
- *Process addressing.* The communication end point at the receiving process is identified via the IP address of the destination host and the port number for the receiving process at the destination host. Also the communication end point of the sending process is identified via the IP address of the source host and the port number for the sending process at this source host. While the source IP address and destination IP address are carried in the IP header, the source port number and destination port number are carried in the transport header of the messages exchanged between source process and destination process.

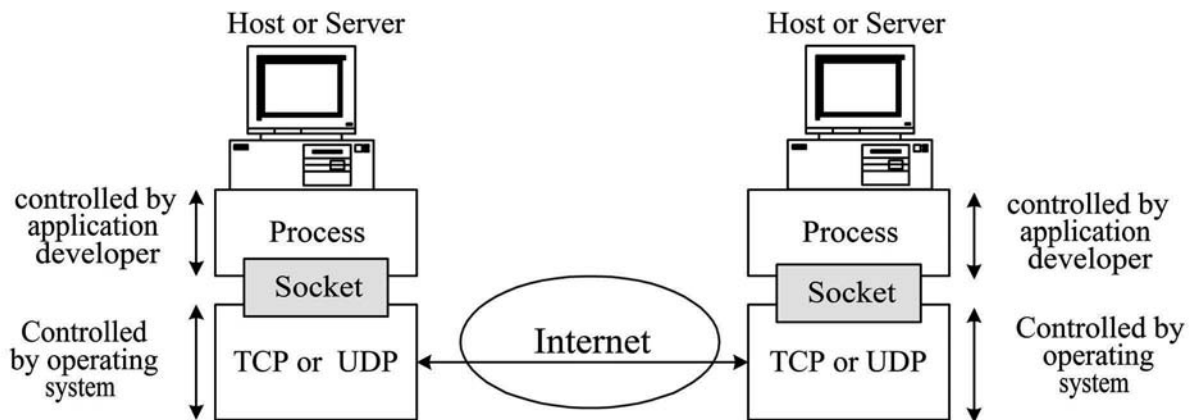


Figure 4-38: Process communication and sockets

4.6.1 Application Layer Services

The network applications can be classified into elastic and inelastic applications. Depending on which services a network application may need from the transport layer, the developer will select whether TCP or UDP is used. Application service requirements can be classified into three categories: data loss, bandwidth and timing. It depends on whether the application is elastic or inelastic.

- *Data loss related requirement.* The elastic applications, such as electronic mail, file transfer or remote host access, require reliable data transfer, that is, no data loss. For these applications, TCP is used as underlying transport protocol. The inelastic applications, such as real-time audio/video or stored audio/video, can tolerate some data losses. The effects of data losses on the application quality and the requirement for the amount of packet losses depend strongly on the application and used coding scheme. For loss-tolerant applications, UDP is used as their

transport protocol. In order to guarantee the data loss rate for inelastic applications, mechanisms for data loss report and monitoring must be offered at the application layer.

- *Bandwidth related requirement.* Some inelastic applications require a certain minimum level of bandwidth to be effective. For example, if Internet telephony uses the codec G.711, it encodes voice at 64 kbps. Therefore it must be able to send data into the network and have data delivered to the receiver at this rate. The 64 kbps is the bandwidth this application needs. If this amount of the bandwidth is not available, the application should give up, because receiving bandwidth below the required bandwidth is not used for such bandwidth-sensitive applications. In order to guarantee the required bandwidth, the applications must support bandwidth negotiation and reservation, QoS monitoring and reporting and congestion control. By contrast, elastic applications can take advantage of however much or little bandwidth is available.
- *Timing related requirement.* In addition to bandwidth, some inelastic applications also require a certain maximum latency to be effective. Interactive real-time applications, such as Internet telephony, virtual environments, teleconferencing and multiplayer game require tight timing restrictions on data delivery in order to be effective. Many of these applications require that the end-to-end delays must be only few hundred milliseconds or less. Long delays in Internet telephony tend to result in unnatural pauses in the conversation. In multiplayer games, a long delay between taking an action and seeing the response from environment makes the applications feel less realistic. In order to guarantee the timing, time report and monitoring as well as congestion control are needed. For elastic applications, lower delay is always preferable to higher delay, but no tight timing constraint is required.

Applications	Data loss	Bandwidth	Timing	Transport protocol
File transfer	No loss	No	No	TCP
World Wide Web (WWW)	No loss	No	No	TCP
Real-time audio	Loss-tolerant	Few Kbps-1Mbps	Yes	UDP
Real-time video	Loss-tolerant	10 kbps-5Mbps	Yes	UDP
Internet Games	Loss-tolerant	Few Kbps-10Kbps	Yes	UDP
Financial Applications	No loss	No	yes and no	UDP

Table 4-6: Service requirements for some selected applications

Table 4-6 summarizes the loss, bandwidth and timing requirements of some popular applications as well as the transport protocols used by these applications.

As discussed in the previous section, the transport layer protocols TCP and UDP provide following services to the application layer:

- Addressing
- Multiplexing and demultiplexing
- Unreliable service
- reliable service
- Connection less service
- connection-oriented service
- Error control
- Flow control and congestion control

Because most inelastic applications use UDP as the transport protocol and the UDP does not provide bandwidth and timing guarantee as well as the controlling for inelastic application, in addition to the services provided by the TCP and UDP, following services and mechanisms must be developed at the application layer:

- *Controlling the media applications.* Controlling the multimedia applications such as session setup, session teardown and codec negotiation, is done by SIP and H323, which are illustrated in section 3.9.
- *Congestion control for inelastic applications.* Because the congestion control is only provided by TCP and not by the UDP, inelastic applications using UDP do not have congestion control mechanisms provided from transport layer. In order to guarantee the QoS for such applications, congestion control mechanisms must be added into the application layer. The congestion control mechanisms for inelastic applications are discussed in section 3.5.4 and 3.5.5.
- *Monitoring and reporting the data loss and timing.* Because UDP is used as transport protocol for inelastic applications and UDP does not support any mechanisms that enable the applications to regulate the transmission rate and to guarantee the QoS. In order to regulate the data rate as well as jitter and delay, mechanisms for monitoring and reporting the packets sending between a source and a destination as well as time stamp and jitter must be provided at the application layer. These mechanisms are implemented in the RTP and RTCP protocol that are addressed in section 3.12.

4.6.2 Selected Application Layer Protocols

In this section, some selected application layer protocols will be summarized.

4.6.2.1 Simple Mail Transfer Protocol

SMTP (Simple Mail Transfer Protocol) is an application protocol used to transfer electronic mail through the Internet. SMTP is an IETF standard specified in RFC 5321. SMTP uses TCP as its transport protocol. Therefore SMTP is a connection oriented and reliable data transport protocol. The basis architecture of SMTP is shown in figure 4-39 [RFC 5321]. In SMTP, the client is the computer that is sending email, and the server is the computer that is receiving it. The responsibility of a SMTP client is to transfer mail messages to one or more SMTP servers or report its failure. An SMTP server may be either the destination or an intermediate relay or gateway. SMTP commands are generated by the SMTP client and sent to the SMTP server. SMTP server sends replies to the SMTP client in response to the commands.

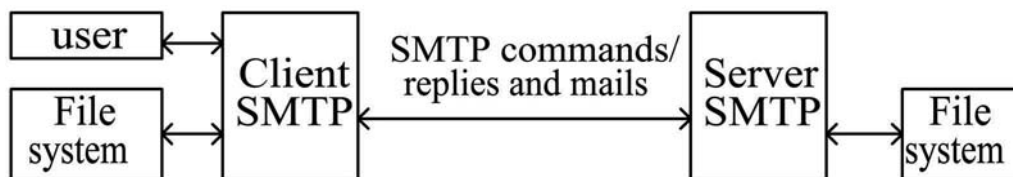


Figure 4-39: SMTP basic architecture

SMTP uses TCP as the underlying transport protocol. Therefore, the SMTP protocol support connection oriented, reliable data transport, congestion control and error control. When an SMTP client (the SMTP sender) has a message to transmit, it establishes a two-way TCP transmission channel to an SMTP server (SMTP receiver). Once the transmission channel is established and initial handshaking is completed, the SMTP client initiates a mail transaction, which consists of a series of command to specify the originator and destination of the mail and the message content. If the message is sent to multiple recipients, SMTP encourages the transmission of only one copy of the data for all recipients at the same destination.

The SMTP offers following services [RFC5321]:

1. *Salutation.* After the TCP communication channel is established, the SMTP server salutes the SMTP client by sending the 220 message (220 <domain name of receiver> Ready) to inform the SMTP client that the SMTP server is ready. After receiving this message from the SMTP server, the client sends the HELO

message (HELO <sender's domain name><CRLF>) to the SMTP server. The server prepares the potentially upcoming mail transactions by assigning available empty buffers for storing the email related data (sender's email address, receiver's email address and the textual content) and state tables to this particular connection. As soon as the SMTP receiver is ready to begin the email transaction he replies to the HELO message by sending the 250 message. This message may also be enriched by a message to inform the sender about e.g. location restrictions for certain SMTP commands.

2. *Email transactions.* After client and server have introduced each other, the SMTP client is able to start the transmission of the email. It is initiated by sending the command: MAIL FROM:<reverse-path><CRLF>. Usually, the reverse-path contains only the sender's absolute mail address. But if the email can not be sent directly to the final receiver who administrates the addressed mailbox, the email has to be relayed. In this case, every SMTP server which relays it inserts his domain name into the reverse-path. Thus, the whole route, which has been passed by the email, is always reversible and in case an error occurs the original sender may be informed by just using the reverse-path as forward-path. The email transaction has been started and by now, the receiver has to accept at least one valid email address to send the email. Therefore the client uses the command "RCPT TO:<forward-path><CRLF>" one forward-path for each usage of this command. The forward-path consists of one absolute email address and an optional list of domain names of servers which are to be used to relay the email to the SMTP server which administrates the addressed mailbox. After receiving this command including the forward-path, the SMTP server has to check whether the address mailbox is administrated by himself or at least if it knows where to relay it to. In the first case, the server replies with the 250 status code and the forward-path is saved. Otherwise, if the email address is not local, but the SMTP server knows where to relay it, the server replies with the 251 status code (251 user not local; 251 will forward to <forward-path>). After at least one email address, including its optional forward route, was accepted, the SMTP client may commence the transfer of email's textual content. To signal this to the SMTP server, the command "DATA<CRLF>" is sent. Both SMTP client and SMTP server are now in the plain text transfer mode. All lines, still

terminated by <CRLF>, are considered to be textual content of the email itself and, step by step, the SMTP client transmits the whole content to the SMTP server. After the SMTP client has finished the plain text transfer, the SMTP server acknowledges the whole email transaction by replying the message with the 250 status code. If there are no mail transactions left, the QUIT command is to be sent from the SMTP client to the SMTP server to close the TCP connection.

3. *Relaying*. Relaying is the process of retransmitting an email until it arrives at the addressed domain's SMTP server.
4. *Other important services*. To investigate whether or not an SMTP server directly administrates a specific mailbox, the command "VRFY <search string>" is used. The command EXPN is used to request whether or not a certain string is used to address a mailing list in the SMTP receiver's domain.

4.6.2.2 Simple Network Management Protocol

SNMP (Simple Network Management Protocol) is a management protocol that facilitates the exchange of management information between network devices and a network management station. The main difference to SMTP is that SNMP operates over UDP. And thus, SMTP supports a connection-less and unreliable data transmission.

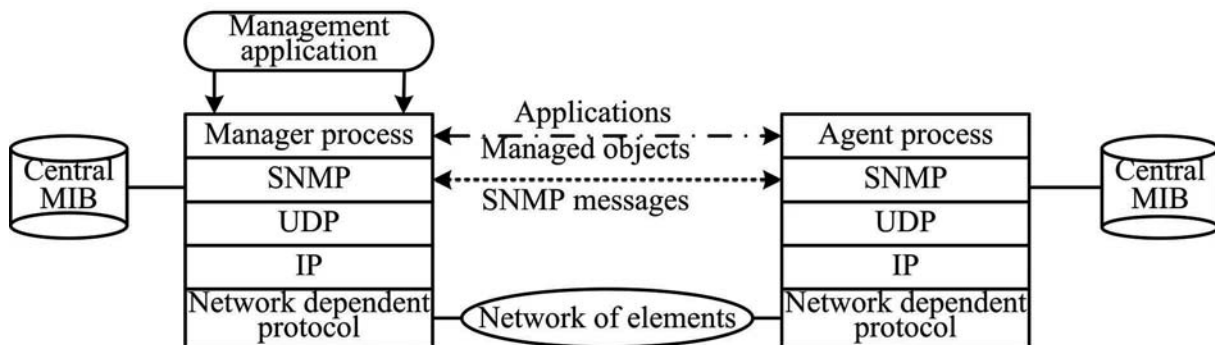


Figure 4-40: Internet Management Architecture

SNMP is a part of the Internet management framework standardized at the IETF. It is used in the Internet management system. Figure 4-40 shows the architecture of the Internet management [hoa-2005]. In this architecture, a manager process controls the access to a central MIB (Management Information Base) at the management station and provides an interface to the management application. Furthermore, a manager may control many agents, whereby each agent interprets the SNMP messages and control the agent's MIBs.

The IETF network management framework consists of [hoa2005]

- *SNMP*. SNMP is a management protocol for conveying information and commands between a manager and a agent running in a managed network device [KR01]
- *MIB*. Resources in networks may be managed by representing them as objects. Each object is a data variable that represents one aspect of a managed device. In the IETF network management framework, the representation of a collection of these objects is called the management information base (MIB) [RFC1066, RFC1157, RFC1212]. A MIB object may be a counter such as the number of IP datagrams discarded at a router due to errors, descriptive information such as generic information about the physical interfaces of the entry, or protocol-specific information such as the number of UDP datagrams delivered to UDP users.
- *SMI*. SMI [RFC1155] allows formal specification of data types that are used in a MIB and specifies how resources within a MIB are named. The SMI is based on the ASN.1 (Abstract Syntax Notation 1) [ASN90] object definition language. However since many SMI-specific data types have been added, SMI should be considered with monitoring and controlling access to managed networks and access to all or part of management information obtained from each nodes.
- *Security and administration* are concerned with monitoring and controlling access to managed networks and access to all or part of management information obtained from network nodes

In the following sections, an overview of several SNMP versions (SNMPv1, SNMPv2, SNMPv3) with respect to protocol operations, MIB, SMI, and security is given.

4.6.2.2.1 SNMPv1

The original network management framework is defined in the following documents:

- RFC 1155 and RFC 1212 define SMI, the mechanisms used for specifying and naming managed objects. RFC 1215 defines a concise description mechanism for defining event notifications that are called traps in SNMPv1.
- RFC 1157 defines SNMPv1, the protocol used for network access to managed objects and event notification.
- RFC 1213 contains definitions for a specific MIB (MIB I) covering TCP, UDP, IP, routers, and other inhabitants of the TCP/IP world.

4.6.2.2.1.1 SMI

The RFCs 1155, 1212 and 1215 describe the SNMPv1 structure of management information and are often referred to as SMIv1. Note that the first two SMI documents do not provide definitions of event notifications (traps). Because of this, the last document specifies a straightforward approach toward defining event notifications used with SNMPv1.

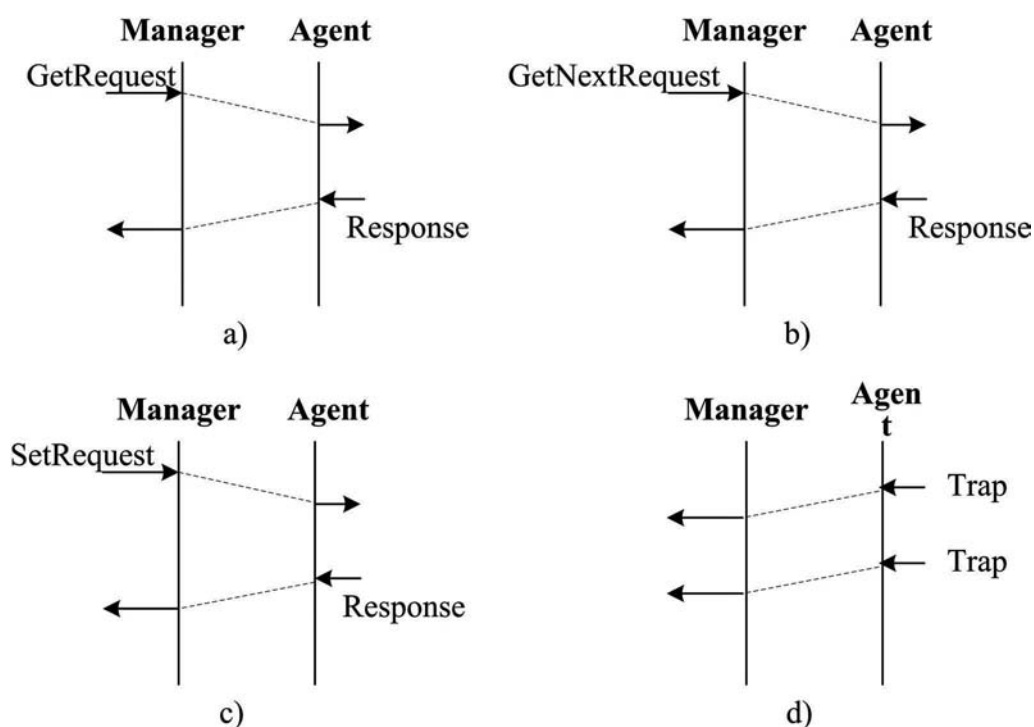


Figure 4-41: Initiative from manager (a, b, c) and from agent (d)

4.6.2.2.1.2 Protocol Operations

In SNMPv1, communication between manager and agent is performed in a confirmed way. The manager at the network management station takes the initiative by sending one of the following SNMP protocol data units (PDUs): GetRequest, GetNextRequest or SetRequest. The GetRequest and GetNextRequest are used to get management information from the agent; the SetRequest is used to change management information at the agent. After reception of one of these PDUs, the agent responds with a response PDU, which carries the requested information or indicates failure of the previous request (figure 4-41). It is also possible that the SNMP agent takes the initiative. This happens when the agent detects some extraordinary event such as a status change at one of its links. As a reaction to this, the agent sends a trap PDU to the manager. The reception of the trap is not confirmed (figure 4-41 (d)).

4.6.2.2.1.3 MIB

As noted above, the MIB can be thought of as a virtual information store, holding managed objects whose values collectively reflect the current state of the network. These values may be required or set by a manager by sending SNMP messages to the agent. Managed objects are specified using the SMI discussed above.

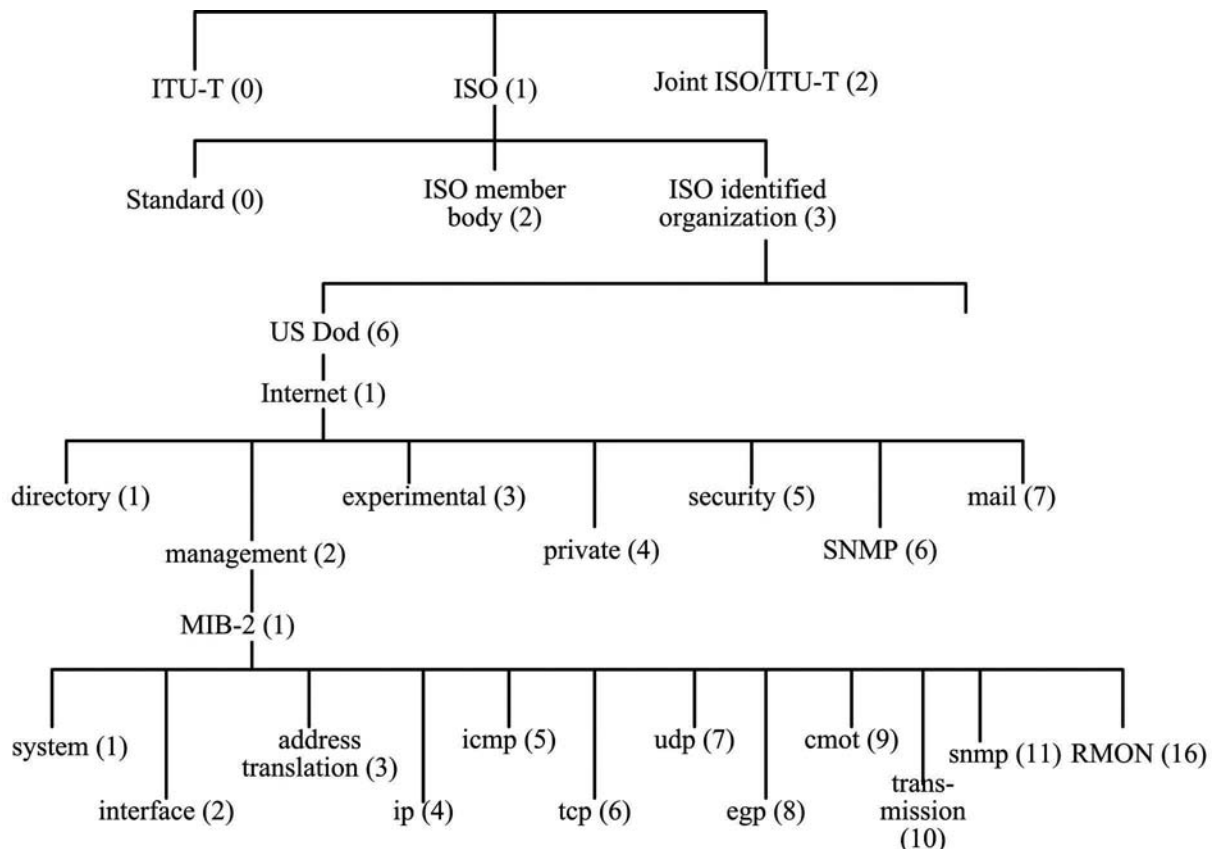


Figure 4-42: ASN.1 object identifier tree

The IETF has been standardizing the MIB modules associated with routers, hosts, switches and other network equipments. This includes basic identification data about a particular piece of hardware and management information about the devices, network interfaces and protocols. With the different SNMP standards, the IETF needed a way to identify and name the standardized MIB modules, as well as the specific managed objects within a MIB module. To do that, the IETF adopted ASN.1 as a standardized object identification (naming) framework. In ASN.1, object identifiers have a hierarchical structure, as shown in figure 4-42.

The global naming tree illustrated in the figure 4-42 allows for unique identification of objects, which corresponds to leaf nodes. Describing an object identifier is accomplished by traversing the tree, starting at the root, until the intended object is reached. Several formats can be used to describe an object

identifier, which integer values separated by dots being the most common approach.

As shown in figure 4-42, ISO and the telecommunication standardization sector of the international telecommunication union (ITU-T) are at the top of the hierarchy. Under the Internet branch of the tree (1.3.6.1), there are seven categories. Under the management (1.3.6.1.2) and MIB-2 (1.3.6.1.2.1) branches of the object identifier tree, we find the definitions of the standardized MIB modules. The lowest level of the tree shows some of the important hardware-oriented MIB modules (system and interface) as well as modules associated with some of the most important Internet protocols. RFC 2400 lists all standardized MIB modules.

4.6.2.2.1.4 Security

The security capabilities deal with mechanisms to control the access to network resources according to local guidelines so that the network cannot be damaged (Intentionally or unintentionally) and persons without appropriate authorization have no access to sensitive information.

SNMPv1 has no security features. For example, it is relatively easy to use the SetRequest command to corrupt the configuration parameters of a managed device, which in turn could seriously impair network operations. The SNMPv1 framework only allows the assignment of different access right to variables (READ-ONLY, READ-WRITE), but perform no authentication. This means that anybody can modify READ-WRITE variables. This is a fundamental weakness in the SNMPv1 framework.

Several proposals have been presented to improve SNMPv1. In 1992, IETF issued a new standard, SNMPv2.

4.6.2.2.2 SNMPv2

Like SNMPv1, SNMPv2 network management framework [RFC1213, RFC1441, RFC1445, RFC1448, RFC1902] consists of four major components:

- RFC1441 and RFC1902 define the SMI, the mechanisms used for describing and naming objects for management purpose.
- RFC1213 defines MIB-2, the core set of managed objects for the Internet suite of protocols.
- RFC1445 defines the administrative and other architectural aspects of the framework.
- RFC1448 defines the protocol used for network access to managed objects.

The main achievements of SNMPv2 are improved performance, better security, and a possibility to build a hierarchy of managers.

4.6.2.2.2.1 Performance

SNMPv1 includes a rule that states if the response to a `GetRequest` or `GetNextRequest` (each of which can ask for multiple variables) would exceed the maximum size of a packet, no information will be returned at all. Because manager can not determine the size of response packets in advance, they usually take a conservative guess and request just a small amount of data per PDU. To obtain all information, managers are required to issue a large number of consecutive requests. To improve the performance, SNMPv2 introduced the `GetBulk` PDU. In comparison with `get` and `GetNext`, the response to `GetBulk` always returns as much information as possible in lexicographic order.

4.6.2.2.2.2 Security

The original SNMP had no security features. To solve this deficiency, SNMPv2 introduced a security mechanism that is based on the concepts of parties and contexts. The SNMP party is a conceptual, virtual execution environment. When an agent or manager performs an action, it does so as a defined party, using the party's environment as described in the configuration files. By using the party concept, an agent can permit one manager to do a certain set of operations (e.g. read, modify) and another manager to do a different set of operations. Each communication session with different manager can have its own environment. The context concept is used to control access to various parts of a MIB; each context refers to a specific part of MIB. Context may be overlapping and are dynamically configurable, which means that contexts may be created, or modified the network's operational phase.

4.6.2.2.2.3 Hierarchy of Managers

Practical experience with SNMPv1 showed that in several cases managers are unable to manage more than a few hundred agent systems. The main cause for this restriction is due to the polling nature of SNMPv1. This means that the manager must periodically poll every system under his control, which takes time. To solve this problem, SNMPv2 introduced the so-called intermediate-level manager concept, which allows polling to be performed by a number of intermediate-level managers under control of top-level managers (TLMs) via the `InformRequest` command provided by SNMPv2.

Figure 4-43 shows an example of hierarchical managers: before the intermediate-level managers start polling, the top manager tells the intermediate-level managers which variable must be polled from which agents. Furthermore, the top-level manager tells the intermediate-level manager of the events he wants to be informed about. After the intermediate-level managers are

configured, they start polling. If an intermediate-level manager detects an event of interest to the top-level manager, a special Inform PDU is generated and sent to TLM. After reception of this PDU, the TLM directly operates upon the agent that caused the event.

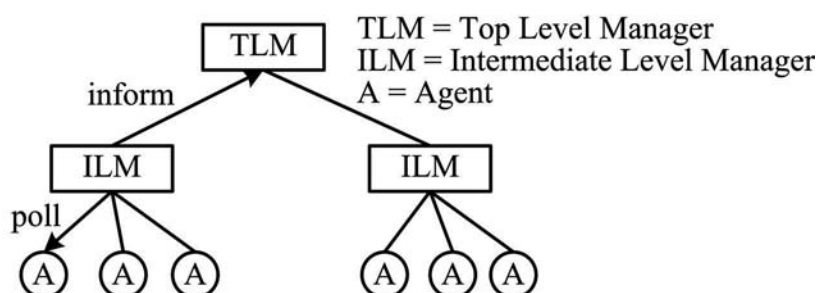


Figure 4-43: Hierarchy of managers

SNMPv2 dates back to 1992, when the IETF formed two working groups to define enhancements to SNMPv1. One of these groups focused on defining security functions, while the other concentrated on defining enhancements to the protocol. Unfortunately, the group tasked with developing the security enhancements broke into separate camps with diverging views concerning the manner by which security should be implemented. Two proposals (SNMPv2m and SNMPv2*) for the implementation of encryption and authentication have been issued. Thus, the goal of SNMPv3 working group was to continue the effort of disbanded SNMPv2 working group to define a standard for SNMP security and administration.

4.6.2.2.3 SNMPv3

The third version of Simple Network Management Protocol (SNMPv3) was published as proposed standards in RFCs 2271 to 2275 [RFC2271, RFC2272, RFC2273, RFC2274, RFC2275], which describe an overall architecture plus specific message structure and security features, but do not define a new SNMP PDU format. This version is built upon the first two versions of SNMP, and so it reuses the SNMPv2 standard documents (RFCs 1902 to 1908). SNMPv3 can be thought of as SNMPv2 with additional security and administration capabilities [RFC2570]. This section focuses on the management architecture and security capabilities of SNMPv3.

4.6.2.2.3.1 The Management Architecture

The SNMPv3 management architecture is also based on the manager-agent principle. The architecture described in RFC 2271 consists of a distributed, in-

interacting collection of SNMP entities. Each entity implements a part of SNMP capabilities and may act as an agent, or a combination of both.

The SNMPv3 working group defines five generic applications (figure 4-44) for generating and receiving SNMP PDUs: command generator, command responder, notification originator, notification receiver, and proxy forwarder. A command generator application generates the GetRequest, GetNextRequest, GetBulkRequest, and SetRequest PDUs and handles Response PDUs. A command responder application executes in an agent and receives, processes, and replies to the received GetRequest, GetNextRequest, GetBulkRequest, and SetRequest PDUs. A notification originator application also executes within an agent and generates Trap PDUs. A notification receiver accepts and reacts to incoming notifications. And a proxy forwarder application forwards request, notification, and response PDUs.

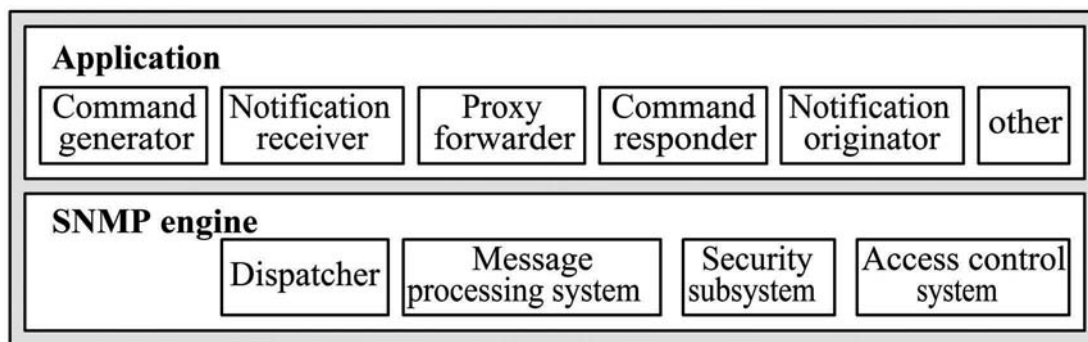


Figure 4-44: SNMPv3 entity

The architecture shown in figure 4-44 also defines an SNMP engine that consists of four components: dispatcher, message processing subsystem, security subsystem, and access control system. This SNMP engine is responsible for preparing PDU messages for transmission, extracting PDUs from incoming messages for delivery to the applications, and doing security-related processing of outgoing and incoming messages.

4.6.2.2.3.2 Security

The security capabilities of SNMPv3 are defined in RFC 2272, RFC 2274, RFC 2275, and RFC 3415. These specifications include message processing, a user-based security model, and a view-based access control model.

The message processing can be used with any security model as follows. For outgoing messages, the message processor is responsible for constructing the message header attached to the outgoing PDUs and privacy functions, if required. For incoming messages, the message processor is used for passing the appropriate parameters to the security model for authentication and privacy

processing and for processing and removing the message headers of the incoming PDUs.

The user-based security model (USM) specified in RFC 2274 uses data encryption standard (DES) for encryption and hashed message authentication codes (HMACs) for authentication [sch95]. USM includes means for defining procedures by which one SNMP engine obtains information about another SNMP engine, and a key management protocol for defining procedures for key generation, update and use.

The view-based access control model implements the services required for an access control subsystem [RFC2275]. It makes an access control decision that is based on requested resource, the security model and the security level used for communication the request, the context to which access is requested, the type of access requested, and the actual object for which access is requested.

4.6.2.3 Hypertext Transfer Protocol

The well-known web application includes several software components: web browser, web server, standard document formats (HTML) and the application layer protocol HTTP (see figure 4-45). A web browser is used to display the requested web page and provides numerous navigational and configurationally features. A web server houses Web objects, each is addressable by a URL. The application layer protocol HTTP defines the format and the order of the PDUs exchanged between a Web browser and a Web server, as well as actions taken on the transmission and/or received of messages or other event. Popular Web server includes Apache, Microsoft Internet Information Server, and the Netscape Enterprise Server.

4.6.2.3.1 HTTP features

The services supported by HTTP are:

- Using TCP transport services,
- Stateless,
- Using both non-persistent and persistent connections, and
- Authentication.

These services will be discussed in the following sub sections

4.6.2.3.1.1 Using the TCP transport services

As an application layer protocol, HTTP uses the TCP transport services to enable the reliable data transfer between a web browser and a web server. Web browser (also called HTTP client) first initiates a TCP connection to a web server (also called HTTP server) on port 80. The Web server accepts the TCP connection request from the web browser with a response message containing

the resource that was requested from the web browser. Once the connection is established, the web browser and the web server access a TCP connection through their socket interfaces. The client sends the HTTP request message into the socket interface and receives HTTP responses from its socket interface. Similarly, the HTTP server receives the request messages from its socket interface and send responses messages into its socket interface. Once a message is sent into the socket interface, the message is treated by TCP. Recall from section 4.5.2.2 that TCP provides a reliable data transmission service. This implies that each HTTP request message sent out from a HTTP client eventually arrives intact at the server; similarly, each HTTP response message sent out from a HTTP server eventually arrives intact at the client. HTTP does not need to take care about data lost or reordering of data. That is the job of TCP and the protocols in the lower layer of the TCP/IP protocol stack.

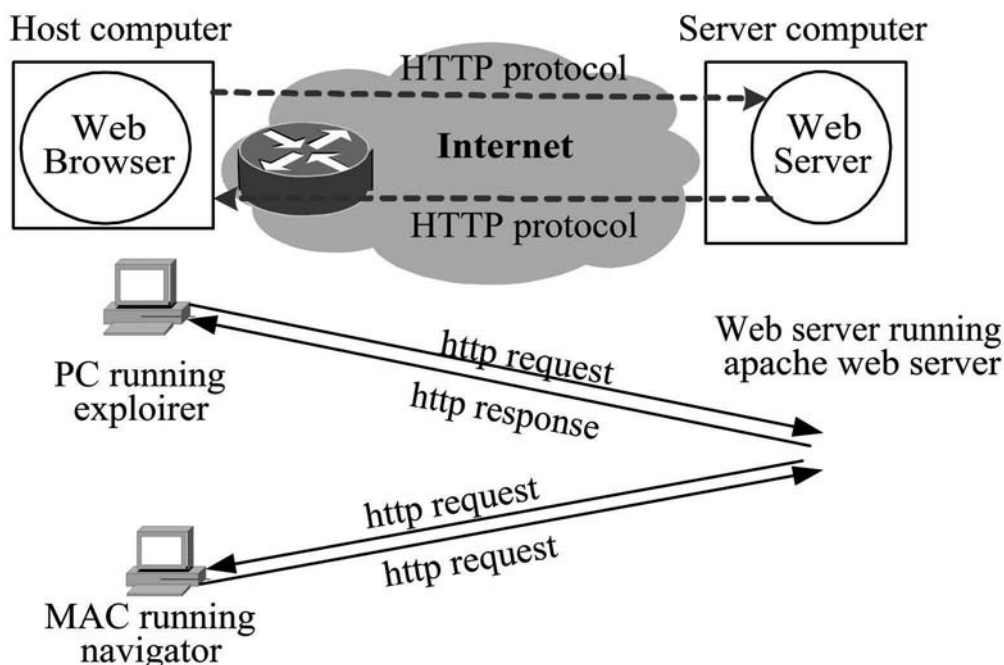


Figure 4-45: The HTTP protocol behaviour (soll figure 4-44 sein ???)

4.6.2.3.1.2 Stateless

The HTTP protocol is stateless, because the HTTP server does not maintain any connection information about past client requests. When a client requests some information (say, click on a hyperlink), the browser sends a request message to the HTTP server for the requested objects. The server receives the requests and sends the response message with the objects. After the server has sent the requested objects to the client, the server does not store any state information about the client, and if the client asks for the same object again, the server

resends the object, and does not reply by saying that it just served the object to the client.

4.6.2.3.1.3 Using both non-persistent and persistent connections

HTTP can use both non-persistent and persistent connections.

- *Non-persistent connections.* A non-persistent connection is the one that is closed after the server sends the requested object to the client. In other words, each TCP connection is used exactly for one request and one response. Each TCP connection is closed after the server sends the object – the connection is not persist for other objects. Thus, when a user requests a web page with 10 JPEG objects, 10 TCP connections are generated for 10 JPEG objects. HTTP 1.0 uses non-persistent connections as its default mode [RFC 1945]. Non-persistent connections have following main limitations. First, a new TCP connection must be established and maintained for each requested object; For each TCP connection, TCP buffers must be allocated and TCP variables (discussed in section 4.5.2.2) must be kept in both the client and the server. This can lead to a serious burden on the web server, which may be serving requests from hundreds of different clients simultaneously. Second, as mentioned in section 4.5.2.2, each object suffers two RTTs – one RTT to establish the TCP connection and one RTT to request and receive an object. This leads to increase the end to end delay. Finally, each object experiences TCP slow start because every TCP connection begins with a TCP slow start phase, which slows down the TCP throughput.
- *Persistent connections.* With persistent connections, the server leaves the TCP connection open after sending the responses and hence the subsequent requests and responses between the same client and server can be sent. The HTTP server closes the connection only when it is not used for a certain configurable amount of time. There exist two versions of HTTP persistent connections: HTTP persistent without pipelining and HTTP persistent with pipelining. In persistent HTTP without pipelining, the HTTP client first waits to receive a HTTP response from the HTTP server before issuing a new HTTP request. In this version, each of the requested objects (e.g. 10 JPEG objects) experiences one RTT in order to request and receive the object. It is an improvement over non-persistent's two RTTs, but depending on network latencies and bandwidth limitation, this can result in a significant delay before the next request is seen by the server. Another limitation of no pipelining is that after the server sends an object over the persistent TCP connection, the connection suspends – it does nothing while waiting for another request to arrive. This hanging

wastes resources of the HTTP server. In persistent HTTP with pipelining, the browser issues multiple HTTP requests into a single socket as soon as it has a need to do so, without waiting for response messages from the HTTP server. This pipelining of HTTP requests leads to a dramatic improvement in page loading time. Since it is usually possible to fit several HTTP requests in the same TCP segment, HTTP pipelining allows fewer TCP packets to be sent over the network, reducing the network load. Pipelining was added to HTTP 1.1 as a means of improving the performance of persistent connections in common cases. With persistent connection, the performance is improved, because Persistent connections are the default mode for HTTP 1.1 [RFC 2616].

4.6.2.3.1.4 Authentication and Cookies

HTTP offers two mechanisms to help a server identify a user: authentication and cookies.

- *Authentication:* HTTP supports the use of several authentication mechanisms to control the access to documents and objects housed on the server. These mechanisms are all based around the use of the 401 status code and the WWW-Authenticate response header. The most widely used HTTP authentication mechanisms are basic, digest and NTLM as follows:
 - *Basic.* The client sends the user name and password as unencrypted base64 encoded text. It should only be used with HTTPS, as the password can be easily captured and reused over HTTP.
 - *Digest.* The client sends a hashed form of the password to the server, since the password cannot be captured over HTTP, it may be possible to relay requests using the hashed password.
 - *NTLM.* A secure challenge/response mechanism is used to prevent password capture or replay attacks over HTTP. However, the authentication is per connection and will only work with HTTP/1.1 persistent connections. For this reason, it may not work through all HTTP proxies and can introduce large numbers of network roundtrips if connections are regularly closed by the web server.
- *Cookies:* A cookie is a piece of data issued by a server in an HTTP response and stored for future use by the HTTP client. The HTTP client only needs to re-supply the cookie value in subsequent requests to the same server. This mechanism allows the server to store user preferences and to identify individual user.

4.6.2.3.2 HTTP Message Format

The HTTP message format is defined in the HTTP specification 1.0 [RFC1945] and HTTP specification 1.1 [RFC2616]. There are two types of HTTP messages – the request messages and the response messages. The format of these messages is illustrated below.

4.6.2.3.2.1 HTTP Request Message

A typical HTTP request message sent from a web browser if a user requested a link (e.g. www.uni-paderborn.de) is shown in figure 4-46 below.

```
GET / HTTP/1.1
Host: www.uni-paderborn.de
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; de;
rv:1.9.1.13) Gecko/20100914 Firefox/3.5.13 (.NET CLR 3.5.30729)
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: de-de,de;q=0.8,en-us;q=0.5,en;q=0.3
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 300
Connection: keep-alive
```

Figure 4.-46: HTTP request message

The message is written in ASCII text and consists of one request URI and several request-header fields [RFC2616]. The request URI is the first two lines of the HTTP request. It consists of three fields shown below: the method field, the HTTP version field and the URI:

```
GET / HTTP/1.1
Host: www.uni-paderborn.de
```

The method field can take on several different values, including GET, POST, and HEAD. The most common form of Request-URI is that it is used to identify a resource on an origin server or gateway. In this case the absolute path of the URI must be transmitted as a request-URI, and the network location of URI (authority) must be transmitted in a host header field.

The request-header fields allow the client to pass additional information about the request and about the client itself to the server. The request-header fields in the HTTP request shown in figure 4-46 are as follows:

```
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; de;
rv:1.9.1.13) Gecko/20100914 Firefox/3.5.13 (.NET CLR 3.5.30729)
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: de-de,de;q=0.8,en-us;q=0.5,en;q=0.3
```

```

Accept-Encoding: gzip, deflate
Accept-Charset: ISO-8859-1, utf-8; q=0.7, *; q=0.7
Keep-Alive: 300
Connection: keep-alive

```

By including the “connection: keep-alive” header line, the browser is telling the server that it wants to use persistent connection. The “User-agent:” header line specifies the user agent, which is the browser type making the request to the server. Here the user agent is Mozilla/5.0. The “accept-language: header” is one of many content negotiation headers available in HTTP.

4.6.2.3.2 HTTP Response Message

A typical HTTP response message sent from a HTTP web server in response to a HTTP request is shown in figure 4-47 below.

```

HTTP/1.1 200 OK
Date: Mon, 11 Oct 2010 12:39:54 GMT
Server: Apache
Set-Cookie: fe_typo_user=a31a0d859d8d670eb427c6d813183ffe; path=/
Vary: Accept-Encoding
Content-Encoding: gzip
Content-Length: 5338
Keep-Alive: timeout=15, max=100
Connection: Keep-Alive
Content-Type: text/html; charset=iso-8859-1
(data data data ....)

```

Figure 4-47: HTTP response message

A typical HTTP/1.1 response message has three sections: a status line, 9 header lines, and then the message body. The message body contains the requested object itself (represented by data data data ...). The status line “HTTP/1.1 200 OK” has three fields: the protocol version field (HTTP/1.1 in figure 4.46), a status code (200 in figure 4.46) and a corresponding status message (OK in figure 4.46). In the example in figure 4.46, the status line indicates that the HTTP server agrees to use HTTP/1.1 for communicating with the client and responds with 200 OK indicating that it has successfully processed the client’s request.

The first header line “Date:” indicates the time and date the HTTP response message was created and sent by the HTTP server. The HTTP server then sends the data and identifies itself as an apache HTTP server. By including the “connection: keep-alive” header line, the server is telling the client that it agrees to use persistent connections.

4.6.2.3.3 HTTP Methods

HTTP/1.0 and 1.1 allow a set of methods to be used to indicate the purpose of a request. Three most often used methods are GET, HEAD and POST.

- *GET*. The GET method is used to request for a document. When one click on a hyperlink, GET is being used.
- *HEAD*. The HEAD method is used to request only for information about a document, not for the document itself. HEAD is much faster than GET, as a much smaller amount of data transferred.
- *POST*. The POST method is used for transferring data from a client to a server. Its goal is to allow a uniform method to cover functions like: annotation of existing resources; posting a message to a bulletin board, new group, mailing list; providing a block of data to a data-handling process; extending a database through an append operation.

4.6.2.4 Real Time Transport Protocol

The Real Time Transport Protocol (RTP) [SCF-2003] developed within the IETF is the most widely used application layer protocol for real time audio applications. Most of the VoIP applications, Video conferencing applications and Audio conferencing applications support RTP. Moreover, the standards proposed for Internet telephony, such as H.323 or SIP, define RTP as the application level transport protocol for the data. Detail of RTP is addressed in section 3.12 (Audio and Video transport).

4.6.3 Summary

The application protocols described in this section and their mechanisms are summarized in the table 4.6 below.

Protocol Mechanisms and Transport services		Selected Application Protocols			
		SMTP	SNMP	HTTP	RTP/RTCP
Used transport protocol	UDP		x		x
	TCP	x		x	
Authentication and Cookies		x	x	x	
Connection Management	Connectionless		x		x
	Connection oriented	x		x	
Addressing	MAC				
	IP	x	x	x	x
	Port number	x	x	x	x
Unreliable service			x		x
Reliable service		x		x	x
Monitoring and reporting the data loss and timing					x

Table 4-6: Selected application layer protocols and their mechanisms

5. Next Generation Networks and the IP Multimedia Subsystem

5.1 Introduction

A Next Generation Network (NGN) is a packet-based network that enables on the one hand the deployment of access independent services over converged fixed and mobile networks, and on the other hand the use of multiple broadband and QoS-enabled transport technologies in which service-related functions are independent from underlying transport-related technologies [TR-180.000]. NGN is one of four current solutions (GAN – cellular integration; 3GPP – WLAN internetworking; Femtocells; NGNs) for the Fixed Mobile Convergence (FMC), which is the convergence technology offering a way to connect a mobile phone to a fixed line infrastructure so that operators can provide services to their users irrespective of their location, access technology and end terminal.

Next Generation Networks are based on Internet technologies including Internet Protocol (IP) and Multiprotocol Label Switching (MPLS) as the transport technology, and Session Initiation Protocol (SIP) at the application layer. Based on these technologies, NGNs allow the transport of various types of traffic (voice, video, data and signalling). Triple play services (Voice, Internet and TV) are available via Cable and xDSL already. The NGN brings mobility in to the picture and the opportunity for further bundling of high revenue services for customers.

At the core of a NGN is the IP Multimedia Subsystem (IMS), which is defined by 3GPP and 3GPP2 standards and organisations and is based on Session Initiation Protocol (SIP). IMS is a framework consisting of a set of specifications that describe the NGN architecture for implementing Voice over IP (VoIP) and multimedia services. The IMS standard defines architecture and concepts that enables the convergence of data, voice, video, fixed network technologies and mobile network technologies over an IP based infrastructure. IMS provides an access independent platform for any type of access technologies such as a fixed line, CDMA, WCDMA, GSM/EDGE/UMTS, 3G, WIFI or WiMax. IMS allows features such as Presence, IPTV, Messaging, and Conferencing to be delivered irrespective of the network in use. IMS is anticipated that we are moving into an era where rather than having separate networks providing us with overlapping services, it is the relationship between the user and service that is important and the infrastructure will maintain and manage this relationship regardless of technology. The most obvious overlap

currently is between fixed and mobile networks, and the IMS has been identified as a platform for the FMC technology.

This chapter will first describe the next generation network architecture and its fundamental mechanisms. After that, the chapter will discuss about the IMS – the core of each NGN and the main platform for the fixed mobile convergence.

5.2 Next Generation Network

Next Generation Networking (NGN) refers to a framework for developing packet-based networks that are able to provide telecommunication services and to make use of multiple broadband and QoS-enable transport technologies in which service-related functions are independent from underlying transport related technologies [Y.2001, Y.2011]. In particular, NGNs promise to be multiservice, multiprotocol, multi-access and IP-based networks with secure, reliable and trusted. NGNs incorporate real-time multimedia communications and service quality management functionality, providing high-quality video, video conferencing, and high-reliability communication services for enterprises in addition to existing fixed and mobile telephone services.

Standardization of NGNs is specified by ITU-T through regional standard development organizations such as ETSI and ATIT. The ETSI's Telecommunications- and Internet-converged Services and protocols for Advanced Networking (TISPAN) technical committee deals with fixed networks and migration from circuit switched (CS) networks to packet switched (PS) networks. The TISPAN technical committee focuses on all aspects of standardization for present and future converged networks, providing implementations that cover NGN service aspects, architectural aspects, QoS approaches, security related approaches, and mobility aspects within fixed networks.

Active standardization development organizations involved in defining the NGNs are Internet Engineering Task Force (IETF), 3rd Generation Partnership Project (3GPP), 3rd Generation Partnership Project 2 (3GPP2), American National Standards Institute (ANSI), CableLabs, MultiService Forum (MSF), and Open Mobile Alliance (OAM).

Much of the NGN activities has been documented by the ITU-T NGN group and will be addressed in this section [TS181.001, TS181.002, TS181.005, TS181.018, TS188.001].

5.2.1 NGN Architecture

The general architecture of NGNs is typically described in terms of functional blocks that are used in combination to allow service providers to support a range of multimedia services. The NGN overall functional architecture developed by the ITU-T is shown in figure 5-1.

The NGN architecture defines a Network-Network Interface (NNI), User-Network Interface (UNI), and an Application Network Interface (ANI). This architecture is structured according to the service stratum and transport stratum. Whereby, the term stratum is intended to signify a set of one or more layers, as conventionally described in the ISO reference model. This architecture enables new subsystems to be added over time to cover new demand and service classes. It also provides the ability to import subsystems defined by other standardizations. Each subsystem is specified as a set of functional entities and related interfaces.

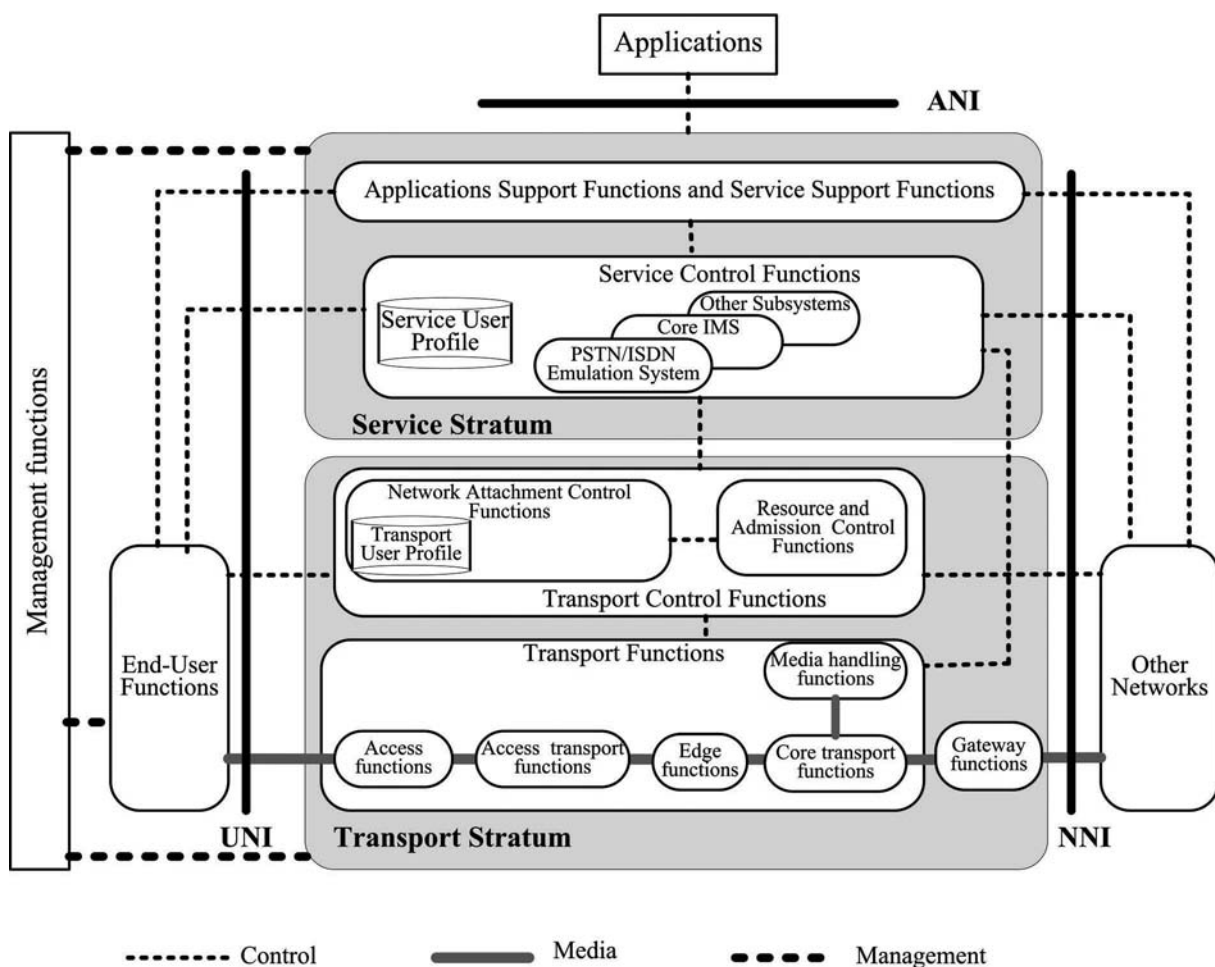


Figure 5-1: The NGN Architecture Overview [ITU-T Y.2012]

The 2-layered NGN architecture model incorporates the separation between service related and transport-related functions, allowing them to be offered separately and to involve independently.

- *Service Stratum*: The service stratum includes the control functions and the application layer functions. The start of this service stratum is denoted by layer 4 of OSI reference model and the end is denoted by layer 7 of OSI reference model. Thus, the NGN transport layer can involve all functions defined from layer 4 to layer 7 of OSI reference model. The NGN service stratum comprises the following:
 - PSTN/ISDN emulation subsystem
 - IMS core
 - Other multimedia subsystems (e.g. streaming subsystem, content broadcast subsystem)
 - Common components used by several subsystems (e.g. subsystems for charging functions, user profile management)
- *Transport stratum*: The transport stratum provides the IP connectivity for NGN users. The transport stratum functions are intended to include all those functions that are responsible for forwarding and for routing of IP packets, including those functions needed to provide the required QoS capabilities for any given service. The end of the NGN transport layer is indicated by the layer 3 of the OSI reference model. The main feature of the NGN protocol reference model shown in figure 5-2 is the use of IP as the common packet mode transfer protocol, which is virtually in all technology configurations.

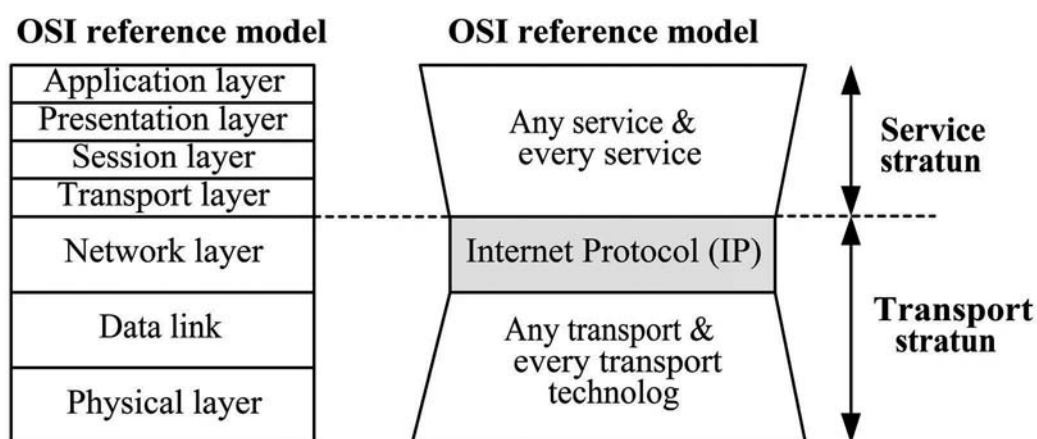


Figure 5-2: The NGN protocol stack architecture

5.2.2 NGN Functions

The NGN functions shown in figure 5-1 are classified into service stratum functions, transport stratum functions, end-user functions, and management functions.

5.2.2.1 Transport Stratum Functions

The Transport Stratum Functions provides IP connectivity services to NGN users under the control of Transport functions and Transport control functions, including the Network Attachment Control Functions (NACF) and Resource and Admission Control Functions (RACF).

5.2.2.1.1 Transport Functions

The transport functions defined in the ITU-T Recommendation Y.2012 are responsible for transmitting media data, control information and management information. These transport functions refer to all functions that are concerned with forwarding and routing the IP packets, including those needed to provide the end-to-end QoS for any given service. The transport functions defined in the ITU-T recommendation include access functions, access transport functions, edge functions, core transport functions, gateway functions and media handling functions (see figure 5-1).

- *Access Functions (AFs)*. The access functions address the mechanisms to manage the end-user access to the network. The access functions are access-technology-dependent, such as wideband code-division multiple access (W-CDMA) and digital subscriber line (xDSL). These access functions provide mechanisms related to cable access, DSL technology, wireless technology, Ethernet technology, and optical access.
- *Access Transport Functions (ATFs)*. These functions are responsible for delivering data across the access network. They offer traffic management and QoS control mechanisms dealing directly with user traffic, including buffer management, packet queuing and scheduling, packet filtering, traffic classification, marking, policing and shaping (as discussed in chapter 3).
- *Edge Functions (EFs)*. These functions are used for processing the access traffic when this traffic is merged into core network.
- *Core Transport Functions (CTFs)*. These functions include the mechanisms for ensuring data transport throughout the core network. They provide the means to differentiate the transport quality in the network, according to interactions with the transport control functions. The Core

Transport Functions also provide QoS mechanisms dealing directly with gate control, firewalls, user traffic management, including buffer management, traffic classification, traffic marking, packet policing and shaping (as described in chapter 3).

- *Gateway Functions.* These functions offer capabilities to internetworking with other networks, such as PSTN/ISDN/PLMN-based networks and the Internet. These functions also support internetworking with other NGNs belonging to other administrators.
- *Media Handling Functions.* These functions address the mechanisms for processing the media resource, such as tone signal generation, transcoding and conference bridging.

5.2.2.1.2 Transport Control Functions

In contrast to the transport functions, the transport control functions do not provide the transfer of data and control information. The transport control functions include resource and admission control functions (RACF), network attachment control functions (NACF) and Transport User Profiles Functions. While the RACFs take into account the capabilities of transport networks and the associated transport subscription information for subscribers in support of the resource control, NACFs provide identification and authentication, managing the IP address space of access networks, and authenticating access sessions. Terminals that talk to the NGN will authenticate with the Network Attachment Control Functions (NACF), receiving an IP address, getting configuration information, etc. Once attached to the network, terminals will communicate directly or indirectly with the Resource and Admission Control Functions (RACF) in order to get desired QoS for communication, and to get permission to access certain resources, etc.

- *Resource and Admission Control Functions (RACFs).* RACF acts as the arbitrator between Service Control Functions and Transport Functions to provide applications with a mechanism for requesting and reserving resources from the access network. The RACFs involve the admission control and gate control mechanisms, including control of network address and port translation (NAPT) as well as differentiated services code points (DSCP). Admission control deals with mechanisms that check whether admitting a new connection would reduce the QoS of existing connections, or whether the incoming connection's QoS requirements can not be met. If either of these conditions holds, the connection is either delayed until the requested resources are available or rejected. It also involves authentication based on user profile, taking into account operator-specific policy rules and resource availability. The RACFs

interact with transport functions to perform one or more of the following traffic management functionalities in the transport layer: packet filtering, traffic classification, marking and policing, bandwidth reservation and allocation, NAPT, anti-spoofing of IP addresses and NAPT/FW traversal. More specifically, the RACS covers following mechanisms [ETSI-ES-187-003]:

- *Session admission control*: Estimating the QoS level that a new user session will need and whether there is enough bandwidth available to service this session
 - *Resource reservation*: permitting applications to request bearer resources in the access network
 - *Service-based local policy control*: Authorizing QoS resources and defining policies
 - *Network address translation (NAT) traversal*: establishing and maintaining IP connections traversing NAT.
- *Network Attachment Control Functions (NACFs)* – These functions provide mechanisms for subscriber registration at the access level and for initialization of the end-user functions for accessing NGN services. They provide network-level identification/authentication, access network IP address space management, and access session authentication. These functions also announce the contact point of the NGN service and application functions to the end user. In particular, the NACF includes following mechanisms [ETSI-ES-187-004]:
 - Authentication of network access based on user profiles
 - Authentication of end users
 - Dynamically provisioning the IP addresses and other terminal configuration parameters
 - Authentication at the IP layer, before or during the address allocation procedure
 - Location management at the IP layer
- *Transport User Profile Functions (TUPFs)*. These functions are responsible for compilation of user and other control data into a single “user profile” function in the transport stratum. TUPFs are specified and implemented as a set of cooperating databases with functionality residing in any part of the NGN.

5.2.2.2 Service Stratum Functions

The Service Stratum Functions provide session-based and non-session-based services, including subscriber notification for presence information and methods for instance messaging. The functions provided by the NGN Service Stratum are

Service Control Functions (SCF), Application Support Functions and Service Support Functions (ASSSF), and Service User Profile Functions.

5.2.2.2.1 Service Control Functions (SCF)

The SCF is responsible for resource control, registration, authentication and authorization at the service level for both mediated and non-mediated services. As shown in figure 5-1, the SCF compromises the functionalities of the PSTN/ISDN emulation subsystem, the IMS core and other multimedia subsystems that will be summarized in the following.

- *PSTN/ISDN Services in an NGN.* An aim of the NGN is to serve as a PSTN and ISDN replacement. That is, an NGN takes off a PSTN/ISDN from the point of view of legacy terminals (or interfaces) via an IP network through a residential access gateway. This is referred to as PSTN/ISDN emulation. All PSTN/ISDN services remain available and identical so that the end users are unaware that they are not connected to a time-division multiplexing (TDM)-based PSTN/ISDN. The ITU-T H.248 protocol is used by the emulation to control the gateway. The NGN also provides PSTN/ISDN simulation, allowing PSTN/ISDN-like services to be supported at advantaged IP terminals or IP interfaces. The 3GPP/TISPAN SIP version is used to provide these simulation services.
- *Core IMS.* The IMS is the main platform for convergence and is currently at the heart of NGNs. The IMS is IP-based and allows applications and services to be supported seamlessly across all networks. IMS mechanisms are subscriber registration, authentication and authorization at service level. More about IMS will be addressed in chapter 5.1.
- *Other multimedia subsystems.* The NGN service stratum also comprises other multimedia subsystems such as streaming subsystem, content broadcasting subsystem.

5.2.2.2.2 Application Support Functions and Service Support Functions

In comparison with SCF, the ASSSF refers to these same functions but at the application level and not at the service level. ASSSF includes functions such as the gateway, registration, authentication function at the application level. These functions are available to functional groups of applications and end users. The ASSSF works in conjunction with SCF to provide end-users and applications with the NGN services they request.

5.2.2.2.3 Service User Profile Functions

These functions represent the compilation of user data and other control data into a single user profile function. They may be specified and implemented as a set of cooperating databases residing in any part of the NGN.

5.2.2.3 Management Functions

Supporting management capabilities is a fundamental basis of NGNs. These functions enable the management of NGNs in order to provide service with expected QoS, security and reliability. As conventional networks, NGN management functions cover the following areas [Hoa-2005]:

- *Fault management.* Fault management deals with the mechanisms for detection, isolation and correction of abnormal operations at the NGN devices and terminals. The fault management includes functions to
 - Maintain and examine error logs
 - Trace and identify faults
 - Accept and act upon error notifications
 - Carry out diagnostic tests and correct faults
- *Configuration management.* Configuration management is a set of facilities that allow network managers to exercise control over the configuration of the network components and OSI layer entities. Configuration management includes the functions to
 - Record the current configuration
 - Record changes in the configuration
 - Initialize and close down managed objects
 - Identify the network components
 - Change the configuration of managed objects
- *Accounting management.* Accounting management deals with the collection and processing of accounting information for charging and billing purposes. It should enable accounting limits to be set and costs to be combined when multiple resources are used in the context of a service. The NGN accounting management functions also include charging and accounting functions, which interact with each other in the NGN to collect accounting information, in order to provide the NGN service provider with appropriate resource utilization data, enabling the service provider to bill the users of the system.
- *Performance management.* Performance management is the set of facilities that enable the network managers to monitor and evaluate the performance of the systems and layer entities. Performance management involves three main steps: (1) performance data are gathered on variables

of interest to the network administrators, (2) the data are analyzed to determine normal levels, and (3) appropriate performance thresholds are determined for each important variable so that exceeding these thresholds indicates a network problem worth attention. Management entities continually monitor performance variables. When a performance threshold is exceeded, an alert is generated and sent to the network management system.

- *Security management.* Security management addresses the control of the access to the network resources according to local guide lines so that the network cannot be damaged and persons without appropriate authorization cannot access sensitive information. A security management subsystem, for example, can monitor users login on to a network resource and can refuse access to those who enter inappropriate access codes. Security management provides support for management of:
 - Authorization facilities
 - Access control
 - Encryption and key management
 - Authentication
 - Security log.

5.2.2.4 End User Functions

End User Functions provide mechanisms at the end users that have data, media and management interfaces (figure 5-1). All customer equipment types (either fixed or mobile) are supported in the NGN.

5.3 IP Multimedia Subsystem

This section addresses the fundamental of the IP multimedia subsystem (IMS). It starts with an introduction to IMS and its standards. After that the IMS architecture is described in 5.3.1. Subsection 5.3.2 explains the IMS fundamental mechanisms. The IMS services are discussed in 5.3.3. Key protocols used within IMS are illustrated in 5.3.4. Finally, IMS implementations are expressed.

5.3.1 Introduction

IP Multimedia Subsystem (IMS) is an architectural framework specified in a set of 3rd Generation Partnership Project (3GPP) documents that defines components, services and interfaces for Next Generation Networks. IMS uses the 3GPP standardized SIP implementation for the Internet signalling, and runs

over the Internet Protocol (IP). IMS supports the connectivity with existing packet-switched networks (e.g. the Internet) and circuit-switched networks (e.g. the PSTN). IMS allows operator to use any type of access network technologies (e.g. fixed line, CDMA, WCDMA, GSM/EDGE/UMTS, 3G, WIFI or WiMax), because IMS is an access independent platform. Furthermore, IMS allows telecommunication operators to provide both mobile and fixed multimedia services.

The big difference between IMS and the other new technologies is that IMS is not a new technology (e.g. MPLS), not a new protocol (e.g. IPv6), not a new product. In fact, IMS integrates many existing network concepts, protocols and standards, such as SIP signalling (section 3.9), Voice over IP (section 3.12), IPv6 and IPv4 (section 4.4), Authentication Authorization Accounting (e.g. Diameter and Radius protocol), presence, call direction services, multimedia services, and traffic management and QoS (sections 3.2, 3.3, 3.4, 3.5, 3.6, 3.8, 3.10).

What this new IMS framework does is draw together call control and service provisions into a horizontally integrated system that allows new services and combination of services (e.g. presence list, rich call group chat, push-to-talk, multimedia advertising, instance massaging, multiparty gaming) to be developed and deployed by mobile and fixed network operators in shorter time cycles and with greater interoperability. IMS enables the carriers to find out the new revenue-generating applications and services and which are the right choices for network-infrastructure evolution. The main revenue is still generated from legacy networks. They are basically single-purpose networks providing a “silo” solution, referred to as vertically integrated networks. The user who wants to access different services must go back and forth between these silos to get the complete set of services (figure 5-3 a). Carriers have to establish a totally converged future network for fixed, wireless, and cable on common network architecture to offer a complete set of services with reduced running cost. The IMS is widely accepted as a solution to control and develop new applications and services on a single layer. The key economic driver of IMS is to avoid the parallel development of the same common services for each network, for example presence service for mobile network, presence service for PSTN/ISDN and presence service for the IP network. What IMS does is to draw together session control, multimedia delivery and service provisions into a horizontally integrated system (figure 5-3 b). This allows carriers to introduce new, interesting services in combination with the web environment (chat, presence, etc.) and existing services (telephony, SMS, MMS, TV). The main goal is to enrich the user’s communication experience without the need to know which communication platforms are being used. In other words, with IMS the

traditional vertical store pipe telecommunication networks will be moved into horizontally layered network (figure 5-3).

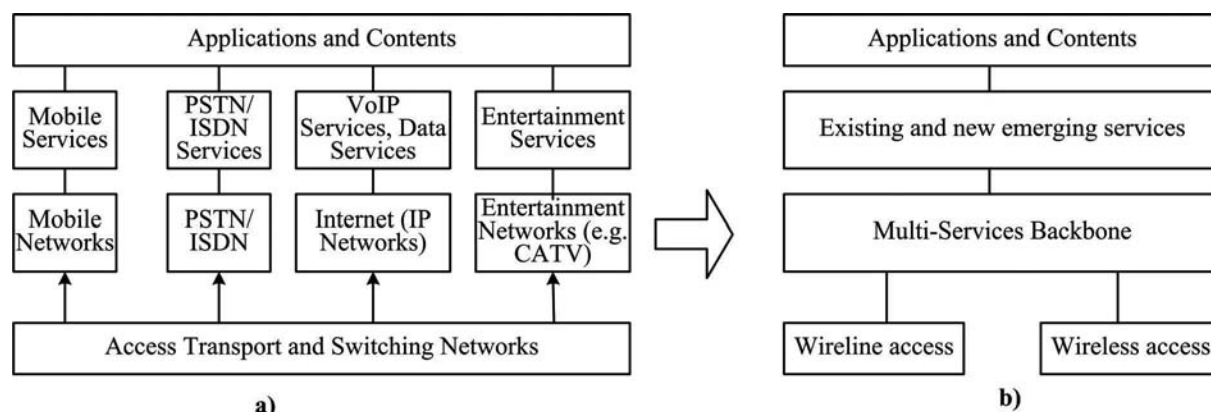


Figure 5-3: Traditional vertical integration of Services (a) vs. future converged services based horizontally integrated services (b)

The key reason to use the IMS is that it is able to offer multimedia services over fixed and mobile networks. Key issues addressed in IMS are convergence, fast and efficient service creation and delivery, as well as service interconnection and open standards.

- *Convergence.* IMS defines the concept of convergence including service convergence and network convergence. A significant benefit of IMS is the service convergence that enables services such as presence, push-to-talk and telephony to be equally equipped to work in both the fixed and mobile worlds and to bridge the gap between them. The another benefit is the network convergence allowing one single integrated network for all access types such as fixed voice access, fixed broadband access using DSL, Wi-Fi, mobile packet networks and more.
- *Fast and efficient service creation and delivery.* In a non-IMS network, services are specified and supported by a single logical node or set of nodes that performing specialized tasks for each specific service. Each service is an island, with its own service-specific nodes. With the IMS, many functions can be reused for fast service creation and delivery that can be accessed through standardized means. Thus, sign-on and authentication process in IMS becomes simpler for subscribers and operators
- *Service interconnection and open standards.* IMS enables not only the creation of a wide range of communication services but also the delivery of these services across the whole operator community. These communication services span the whole operator network, from the user-network interface (UNI) to the network-network interface (NNI). The

User applications such as telephony or video on demand will be interconnected through APIs built on these communication services. Instead of establishing separate interconnection agreements per service (e.g. service agreement for PSTN, service agreement for PLMN, service agreement for IP) by non-IMS, the IMS enables the operator to agree a set of basic agreements used for a service. Additionally, new IP services developed within IMS inter-work successfully with wide range of existing PSTN and PLMN services. Thus, one main advantage of IMS is that it has been developed to inter-work with existing networks such as PSTN, PLMN and mobile networks. IMS is recognized as an open standard to offer multimedia services, including multimedia telephony. It is an international standard, first specified by 3GPP/3GPP2 and now being embraced by other standards such as ETSI/TISPAN, OMA and WiMAX forum. This open standard enables IMS to work across different networks, devices and access technologies.

The first step in the development of IMS came about when the Universal Mobile Telecommunications System (UMTS), as it moved toward an all-IP network, saw the need to coordinate its efforts and standardize protocols and network elements. Following this, the 3GPP then first provided a formal definition of a wireless IP network in its release 4 that specified basic IP connectivity between a UMTS operator and external IP networks.

The IMS is primary introduced in the release 5 as a part of 3GPP (the 3rd Generation Partnership Project) specifications. This release also allowed a UMTS operator to provide all services, end-to-end over IP. Release 5 described IMS, SIP and the desirability of end-to-end QoS as a part of “all IP” feature. This release also provided descriptions of VoIP services.

3GPP Release 6 IMS was completed in September 2005. It defined IMS phase 2, where IMS is generalized and made independent of the access network. Release 6 IMS key functions are IMS conferencing, IMS group management, presence service, IMS messaging, inter-networking with WLAN, IMS charging and QoS improvements. 3GPP IMS release 7 added two more access technologies (data over cable service interface and xDSL) and more features such as supplementary services for multimedia telephony, SMS over any IP access, combining circuit switched calls and IMS sessions, IMS emergency calls, Interconnection Border Control Function (IBCF), identification of communication services in IMS, voice call continuity between circuit switching and packet switching domain and policy and charging control. 3GPP IMS release 8 added the support for fixed broadband access via IMS, deals with policing issues, specifies voice call handover between cable and WLAN/IMS systems and standardized end-to-end QoS.

5.3.2 IMS Functional Architecture

The IMS architecture has been designed to enable operators to provide a wide range of real-time, packet-based services and to track their use in a way that allows both traditional time-based charging as well as packet and service-based charging. It has become increasingly popular both with wireline and wireless service providers as it is designed to increase carrier revenues, delivering integrated multimedia services, and creating an open, standards-based network. The 3GPP does not standardize the IMS nodes, but only the IMS functions. This means, the IMS architecture standardized by 3GPP is a collection of functions linked by standardized interfaces. Two or more IMS functions can be implemented in a single physical node. Similarly, a single function can be split into two or more nodes.

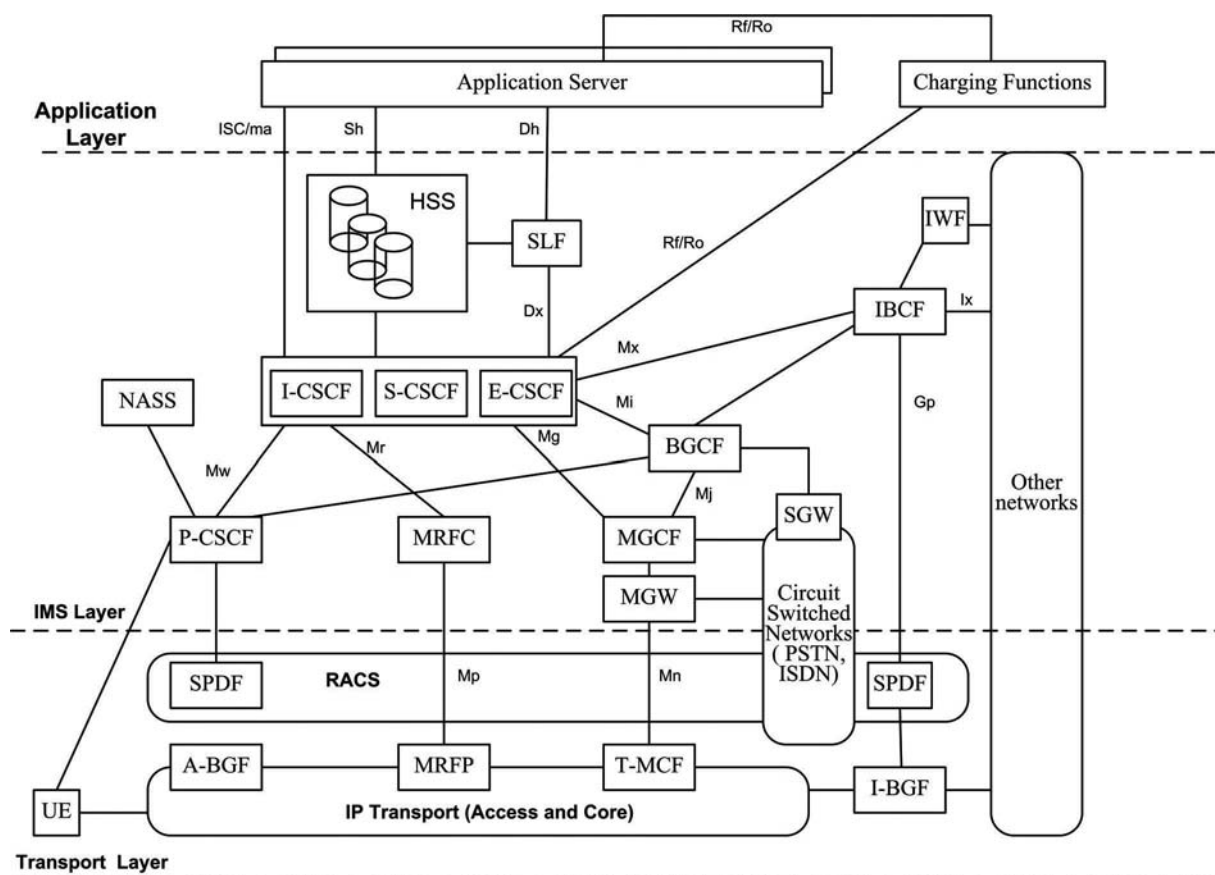


Figure 5-4: IMS Architecture Overview

Figure 5-4 depicts an overview of the IMS architecture standardized by 3GPP. The picture does not show all interfaces defined in IMS, but only the most relevant signalling interfaces referred by a two or three-letter code. Each interface is specified as a “reference point”, which defines both protocols over the interface and the functions between which it operates. The 3GPP/TISPAN

IMS architecture is split into three main layers: Application Layer, IMS Layer and Transport Layer.

- *Application Layer.* The application layer includes the IMS functions for provisioning and controlling the IMS services. The application layer defines standard interfaces to common functionality including
 - configuration storage, identity management, subscriber status (such as presence and location), which is held by the Home Subscriber Server (HSS)
 - billing services, provided by a Charging Gateway Function (CGF)
 - Control of voice and video calls and messaging, provided by the control plane.
- *IMS layer.* The IMS layer sits between the application and transport layer. It is responsible for routing the calls, controlling the signalling and the traffic access, and generating the billing information. The core of this IMS layer is the Call Session Control Function (CSCF), which comprises the Proxy-CSCF (P-CSCF), Interrogating-CSCF (I-CSCF), the Serving-CSCF (S-CSCF) and the E-CSCF. These functions will be addressed in the next sub section. This IMS layer provides an extremely flexible and scalable solution. For example, any of the CSCF functions can generate billing information for each operation. The IMS layer also controls the transport layer traffic through the Resource and Admission Control Subsystem (RACS). It consists of the Policy Decision Function (PDF), which implements local policy on resource usage, for example to prevent overload of particular access links, and Access-RAC Function (A-RACF), which controls QoS within the access network. Furthermore, the IMS layer contains the so called Home Subscriber Server (HSS) that controls the subscriber-related information and performs user authentication and authorization as well as provides the subscriber's location and the IP information.
- *Transport Layer.* The transport layer provides a core QoS-enabled IP network with access from User Equipment (UE) over mobile, WiFi and broadband networks. This infrastructure is designed to provide a wide range of IP multimedia server-based and P2P services. Access into the core network is through Border Gateways (GGSN/PDG/BAS). These enforce policy provided by the IMS core, controlling traffic flows between the access and core networks. The IMS functions within the user plane are
 - *Interconnect Border Control Function (I-BCF)* controls transport level security and tells the RACS what resources are required for a call.

- o *I-BGF and A-BGF Border Gateway Functions* provide media relay for hiding endpoint addresses with managed pinholes to prevent bandwidth theft. Furthermore these functions implement NAT and NAT/Firewall traversal for media flows.

In the following sub-sections, the key functions defined in the IMS architecture will be illustrated in more detail.

5.3.2.1 The Call Session Control Function (CSCF)

As described in the previous section, the CSCF comprises of the Proxy-CSCF (P-CSCF), Interrogating-CSCF (I-CSCF), Serving-CSCF (S-CSCF) and Emergency-CSCF (E-CSCF). The relation between CSCF components, the Application server (AS) and the HSS is illustrated in figure 5-5.

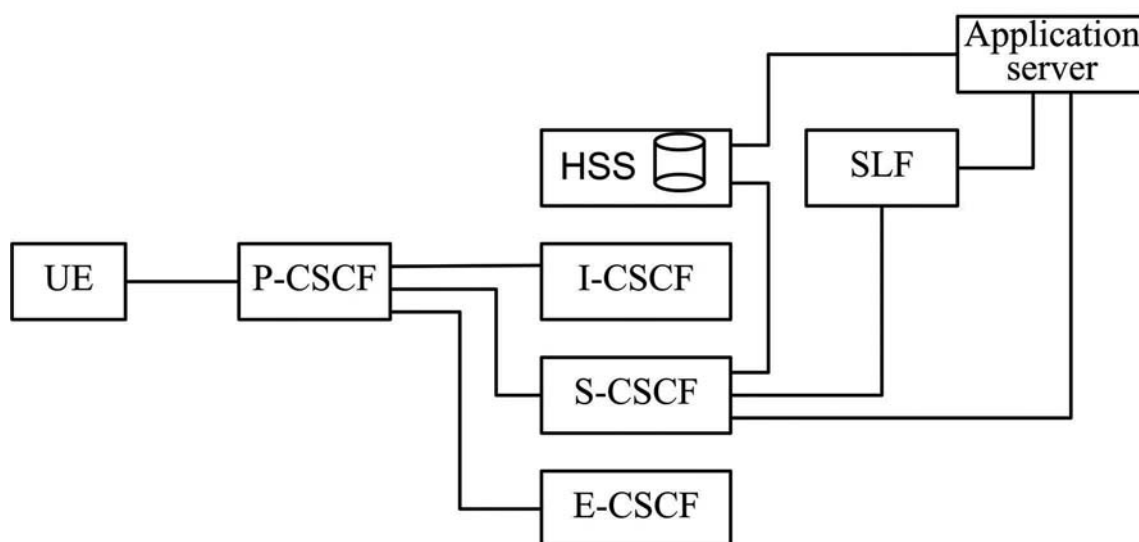


Figure 5-5: CSCF components, the AS and HSS

The CSCF components will be discussed in the following.

5.3.2.1.1 The Proxy-CSCF (P-CSCF)

In order to start receiving e-mail, voice mail or phone calls etc., all UEs (User Equipment) first need to have access to the IMS network. Access to the IMS network is achieved through the P-CSCF. The P-CSCF serves as the first entry point of a UE to the IMS core network. The P-CSCF is responsible for routing incoming SIP messages to the IMS registrar server, for facilitating policy control towards the PCRF (Policy and Charging Rules Function) and setting up IPsec Security associations with the UEs to ensure secure access to the IMS core. Generally, P-CSCF can provide following main functions: P-CSCF discovery, Subscriber authentication, IPsec Security Association, SIP compression, Policy

Decision Function, interaction with Policy and Charging Rules Function (PCRF), Generating Charging Information, and emergency call detection. P-CSCF functions are described in 3GPP TS 24.229 [Ts24.229].

- *P-CSCF discovery.* A UE must find the P-CSCF within its present domain prior to access to the IMS core network. So the P-CSCF discovery is performed between this UE and the P-CSCF. Thus, P-CSCF must be assigned to an IMS UE before registration and does not change for the duration of the SIP registration. P-CSCF discovery can be done through the IP address assignment in the DNS or through a DHCP query.
- *Subscriber Authentication.* P-CSCF provides subscriber authentication that may be established via IPsec Security Association with the IMS UE. That means that P-CSCF maintains the Security associations (SAs) and applies the integrity and confidential protection for the SIP signalling. The IPsec security association is negotiated at the P-CSCF during the SIP registration as the UE. After finishing initial registration, the P-CSCF is able to apply integrity and confidential protection of the SIP signalling.
- *Security for the SIP Messages.* P-CSCF provides security mechanisms to control all SIP signalling traffic sent between UEs through IMS network. This means that P-CSCF will inspect the SIP messages to ensure that communications into the network are from trusted UEs, and not from unauthorized UEs.
- *SIP Compression.* The SIP is text-based VoIP signalling protocol, which contains a large number of headers and header parameters including extensions and security related information so that the SIP message sizes are larger than with binary-encoded protocols. This may lead to delay of the SIP session establishment. In order to reduce the round-trip time for the SIP session establishment, P-CSCF can compress the SIP messages between users and P-CSCF if the UE (user equipment) has indicated that it wants to receive the SIP messages compressed. P-CSCF can also decompress the SIP messages.
- *Policy Decision Function.* P-CSCF may include a Policy Decision Function (PDF), which authorizes media plane resources e.g. QoS over the media plane, if one operator wants to apply policy control and bandwidth management. The PDF allows operators to establish rules to be applied for access to the network. It also controls the policy Enforcement Function in the bearer network. This allows operators to control the flows of packets at the bearer level according to destination and original addresses and permissions.

- *Policy Charging Rule Function (PCRF)*. P-CSCF may include PCRF, which derives authorized QoS information of the media streams and charging rules that will be passed to the access gateway.
- *Generating Charging Information*. With PCRF, P-CSCF is also able to generate the charging information.
- *Emergency Call Detection*. P-CSCF also provides emergency calls
P-CSCF may locate either in the home network or in the visited network.

5.3.2.1.2 The Interrogating-CSCF (I-CSCF)

While P-CSCF is the entry point into IMS network, the I-CSCF is the home network first point of contact from peered IMS networks. It serves as an inbound SIP proxy server in the IMS network. The I-CSCF is responsible to determine whether or not access is granted to other networks. For this reason, I-CSCF can be used to hide the IMS core network details from other operators, determining routing within the trusted domain. Thus, the S-CSCF and HSS can be protected from unauthorized access by other networks. The I-CSCF functions are described in 3GPP TS 24.229 [TS24.229]. Generally, I-CSCF can provide following main functions:

- *Retrieving User Location Information*. I-CSCF is responsible for identifying the location of the user being addressed. In particular, it identifies the S-CSCF assigned to the UE, and the HSS where the subscriber data is stored. This is done during the IMS registration, in which the I-CSCF is responsible for querying the HSS and the SLF using Diameter Cx and Dx interfaces in order to select an appropriate S-CSCF which can serve the UE.
- *Routing the SIP request to the S-CSCF*. After retrieving the S-CSCF, the I-CSCF forwards the SIP-messages to this S-CSCF.
- *Topology Hiding*. The I-CSCF may encrypt a part of SIP messages that contain sensitive information about the domain, such as the DNS names and their capacity. Thus, I-CSCF can be used to hide the IMS core network details from other operators, determining routing within the trusted domain.
- *Providing Load balancing and load sharing*. The I-CSCF's property of S-CSCF selection can be utilized for load sharing amongst multiple S-CSCF nodes in the IMS core

I-CSCF is usually located in the home network.

5.3.2.1.3 The Serving-CSCF (S-CSCF)

The S-CSCF is the heart of the IMS layer. It controls all aspects of a subscriber's service, maintaining status of every session. The S-CSCF controls messaging content and delivery content. It provides the status of a subscriber's registration to other application servers and keeps control over these services as long as UE is registered.

Moreover, the S-CSCF facilitates the routing path for mobile originated or mobile terminated session requests. The S-CSCF is the most processing intensive node of the IMS core network due to its initial filter criteria processing logic which enables IMS service control. It also interacts with the Media Resource Function for playing tones and announcements. The S-CSCF functions are detailed addressed in TS 24.229 [TS24.229]. Generally, S-CSCF can provide following main functions:

- *User authentication.* The S-CSCF acts as a SIP registrar. This means that it maintains a binding between the UE location (the IP address of the UE the user is logged on) and the public user identify. S-CSCF is responsible for authenticating all subscribers who attempt to register their location with the network. The subscriber authentication is done by using the so called authentication vector, which is downloaded from HSS via the diameter interface.
- *Informing the HSS about S-CSCF allocation time.* The S-CSCF informs the HSS that is the S-CSCF allocated to the UE for the duration described in the SIP registration message.
- *Routing SIP messages to the application servers.* The S-CSCF also has the responsibility for enabling services by providing the access to various application servers within the network. This means that the S-CSCF needs to know what services a subscriber is allowed to use and the addresses of servers providing theses services. This is done by using the service profile. The S-CSCF accesses to the HSS and downloads the user profile. The user profile includes the service profile that may cause a SIP message to be routed through one or more application servers
- *Scalability and redundancy.* An IMS network includes a number of S-CSCFs for providing the scalability and redundancy. Each S-CSCF serves a number of UEs, depending on the capacity of nodes.

5.3.2.1.4 The Emergency-CSCF (E-CSCF)

The E-CSCF is responsible for routing the emergency calls to the appropriate public safety answering point (PSAP) or to emergency centre based on the

location of the UE as indicated by the UE in the session setup signalling. E-CSCF communicates with other CSCF functions via SIP signalling.

When P-CSCF receives an originating session setup (SIP INVITE), it compares the telephone number in the INVITE request with a configured list of emergency destinations. If there is a match, the call is handled as emergency call, which will be prioritized by further processing and forwarding in the network. The P-CSCF forwards the emergency INVITE to the E-CSCF configured in the P-CSCF. When the INVITE arrives at the E-CSCF, the E-CSCF checks the location in the message. If the location is not provided, the E-CSCF queries the HSS to find the location. The E-CSCF queries the routing decision function for getting an appropriate emergency centre number (or addresses). Finally, the E-CSCF routes the emergency call to this number.

5.3.2.2 The Home Subscriber Server (HSS)

The HSS is the central IMS user database keeping the relevant subscriber information of IMS users. The HSS within the IMS core is shown in figure 5-5. The HSS stores all of the user-related subscription data required to handle multimedia sessions. This subscription data include, among other items, various identities (public user identities and private user identity), security information (including both authentication and authorization information), the assigned services a subscriber is allowed to access, the networks a subscriber is allowed to roam to, and the location information of the subscriber UE. HSS can provide following main functions:

- *Providing user profile to the S-CSCF.* When a subscriber registers with the network, the S-CSCF accesses the HSS to retrieve the user profile that identifies the subscriber.
- *Informing the S-CSCF about the subscription change.* If there is a change in the subscription of a subscriber UE (for example a subscriber changes its location), the HSS sends all the subscription data of this subscriber UE to the S-CSCF. If a change occurs, SIP registration is done automatically. The purpose of this registration is to provide a location for a subscriber UE. The location can be the GPS coordinates of a wireless user. In fixed networks, the location is e.g. the IP address assigned to the subscriber UE, and the IP address the P-CSCF uses to access to the IMS network.
- *Allowing Service Barring.* If a subscriber is to be barred from the service access, the operator is able to bar the public user identify or the private user identify associated with a subscription at the HSS.
- *Providing Encryption and authentication keys for each subscription.* A critical function of HSS is to provide the encryption and authorization keys for each subscription. When a subscriber UE registers on the

network, the assigned S-CSCF challenges the UE for the correct credentials stored in HSS. The S-CSCF queries HSS with the first REGISTER message to find out what the correct credentials should be during the registration. The subscriber UE then sends the second REGISTER message containing the correct credentials.

- *Managing multiple public user identifies.* HSS is able to manage multiple public identifies under one common subscription. A subscription may have only one private user identify but it may contain multiple public user identifies. Each public user identify may have one set of services.

5.3.2.3 The Subscription Location Function (SLF)

The SLF provides the scalability of HSS nodes, offering routing services to discover which HSS node has the subscription information of a given user identity. An IMS network may contain more than one HSS, if the number of subscribers is too high to be handled by only one HSS. An IMS network with a single HSS does not need a Subscription Locator Function (SLF). On the other hand, a network with more than one HSS does require an SLF. The SLF is a database that maps user's addresses to the corresponding HSSs.

An S-CSCF queries the SLF with a user's address as the input and obtains the HSS that contains all of the information related to the requested user as the output.

5.3.2.4 Application Servers

The job of an application server is to host and to execute services to end users as well as to interface with the S-CSCF, HSS and SLF using the SIP and diameter protocol. Generally there exist more than one application servers. Typically, there will be several ASs, each specialized in providing a particular service. Depending on services a AS can operate in SIP proxy mode, SIP User Agent mode, or SIP B2BUA (Back to Back User Agent) mode. The AS interfaces the S-CSCF using SIP. It uses Diameter to interface HSS and SLF.

All the IMS services (such as presence, push to talk over cellular, call forwarding, call hold, call waiting, call line identification) will be developed in SIP application server. In the layered design of the IMS architecture shown in figure 5-4, the ASs function on top of the IMS core. An AS relies in the user's home network or in a third-party location. The main functions of a SIP application server are:

- Processing and impacting incoming SIP sessions received from the IMS core
- Originating SIP requests.

- Sending accounting information to the charging functions

The 3GPP defines three different types of application servers, depending on their functionality: SIP Application Server, Open Service Architecture (OSA) Service Capacity Server (SCS), and CAMEL IP Multimedia Service Switching Function (IM-SSF). Thus, services offered by application servers are not limited to SIP-based services, because an operator is able to offer access to services based on the CAMEL (Customized Applications for Mobile Network Enhanced Logic) services developed for GSM in the IMS.

5.3.2.5 The Interconnection Border Control Function (IBCF)

IBCF (Interconnection Border Control Function) acts as a border controller for internetworking with IMS network and processes issues related to interlock and security for inter-working with various networks. IBCF provides media relay between terminals in the network of a service provider. IBCF implements security, resource allocation and management, session filtering, topology and infrastructure hiding, billing and media relay.

5.3.2.6 The Media Resource Function (MRF)

The MRF (Media Resource Function) deals with ability to play announcements, mix media streams, transcode between different codecs, obtain statistics, and do any sort of media analysis. The MRF is divided into MRFC (Media Resource Function Controller) and Media Resource Function Processor (MRFP) (figure 5-4). The MRFC provides all media-related functions (e.g. playing and mixing media). It acts as a SIP User Agent and has a SIP interface toward the S-CSCF. The MRFC controls the resources in the MRFP via H-248 interfaces. The MRF always locates in the home network.

5.3.2.7 The Breakout Gateway Control Function (BGCF)

The BGCF (Breakout Gateway Control Function) (figure 5-4) is a SIP server that is responsible for routing based on telephone numbers. The BGCF only operates in sessions initiated by an IMS UE and addressed to a user in a circuit-switched network, such as PSTN or PLMN. The main functions of the BGCF are:

- Selecting an appropriate network where internetworking with the circuit-switched is required. or
- Selecting an appropriate PSTN/CS gateway for internetworking in the same network where the BGCF is located.

5.3.2.8 The Circuit-Switched Network Gateway

The circuit-switched network (CSN) gateway provides an interface toward a circuit-switched network, allowing IMS UEs to initiate and receive calls to and from the PSTN or any other circuit-switched network. The CSN gateway mainly includes following functions (figure 5-4):

- *Media Gateway Control Function (MGCF)*. MGCF is the central function of the PSTN gateway. It provides the protocol conversion and mapping SIP to either ISUP over IP or BICC over IP. MGCF controls the resources in an MGW. The protocol used between MGCF and MGW is the H.248.
- *Signalling Gateway (SGW)*. The SGW interfaces the signalling plane of the circuit switched networks. It transforms ISUP (ISDN User Part) or BICC (Bearer Independent Call Control) over MTP into ISUP or BICC over SCTP/IP.
- *Media Gateway (MGW)*. The MGW interfaces the media plane of the circuit-switched networks. On one site, MGW is responsible to send and receive IMS media over real time protocol (RTP). On the other side, the MGW uses one or more PCM time slots to connect to the circuit switched networks. Furthermore, MGW performs transcoding when IMS UE does not support the codec used by circuit switched side.

5.3.3 Fundamental IMS Mechanisms

In the last section, key IMS functions specified for the IMS architecture are illustrated. As mentioned above, one or more of these functions can be implemented in one IMS physical component. This section will describe fundamental IMS protocol mechanisms, which are needed for controlling the sending of messages from an IMS function and for the receiving of IMS messages at an IMS function within the IMS functional architecture discussed in section 5.3.2 above.

5.3.3.1 IMS Addressing

The addressing in the packet-switched network described in the chapter 4 and 3 above is used to deliver the packets between sender and receiver. This addressing schema depends on the layer the data is sent and received. For example, while the MAC address is used for identify the frame at the physical layer; the IP address is used for identifying the IP packets at the network layer and the port number is used to identify the segments at the transport layer. At the application layer, there exist several addressing schemes depending on the application. For example, email applications use the email address for sending

and receiving the email, and the Web applications use the Universal Resource Locators (URLs) to identify the web sites.

In the circuit-switched networks, such as PSTN or PLMN, telephone numbers are used to route the calls.

As mentioned above, the IMS provides the connectivity with existing packet-switched networks and circuit-switched networks. It allows telecommunication operators to provide both mobile and fixed multimedia services that a subscriber needs to use. In order to enable this communication through packet-switched and circuit-switched networks the addressing in IMS is needed. The IMS addressing must be able to identify a user, user's subscription, UE and public user identity combination, service, and IMS network entities. To identify them, following addressing scheme are used:

- Public User Identity. This addressing schema is used to identify the IMS subscriber
- Private User Identity. This addressing schema is used to identify the user's subscription.
- Public Service Identity. This addressing schema is used to identify the services
- Globally Routable User Agent. This addressing schema is used to identify the combination of UE and public user identity.

These addressing scheme are described in the following subsections.

5.3.3.1.1 Public User Identity

Public user identities are identities used for communication with other users. IMS users are able to initiate sessions and receive sessions from other users attached on different networks such as PSTN, PLMN, GSM and the Internet. To reach the circuit-switched networks, the public user identity must conform to the telecom numbering (e.g., +495214179493). Similarly, to communicate with the Internet clients, the public user identity must conform to the Internet naming (e.g. Mai.Hoang@gmx.de).

The requirements for IMS public user identities are specified in [3GPP TS 23.228, TS 23.003].

5.3.3.1.2 Private User Identify

The private user identity is a unique global identity defined by the home network operator. It is not used to identify the user. It identifies the user's subscription and therefore it is used for authenticating the subscribers and UEs. The requirements for private user identities are specified in [3GPP TS 23.228, TS 23.003].

Figure 5-6 illustrates the relationship between the private user identity and public user identities. In this example, Mai is working for Coffee Asian and is using a single terminal for her work life and her personal life. She has a private user identity and four public user identities. Two of them (sip:mai.hoang@CoffeeAsian.de, and tel:+495214179493) are for her work life. And another two public user identities are for her personal life. For these public user identities two different service profiles are assigned. One service profile contains data and information about her work life identities, and another profile contains data and information about her personal life identities. These work life identities and personal life identities are stored and maintained in the HSS and downloaded to the S-CSCF when needed.

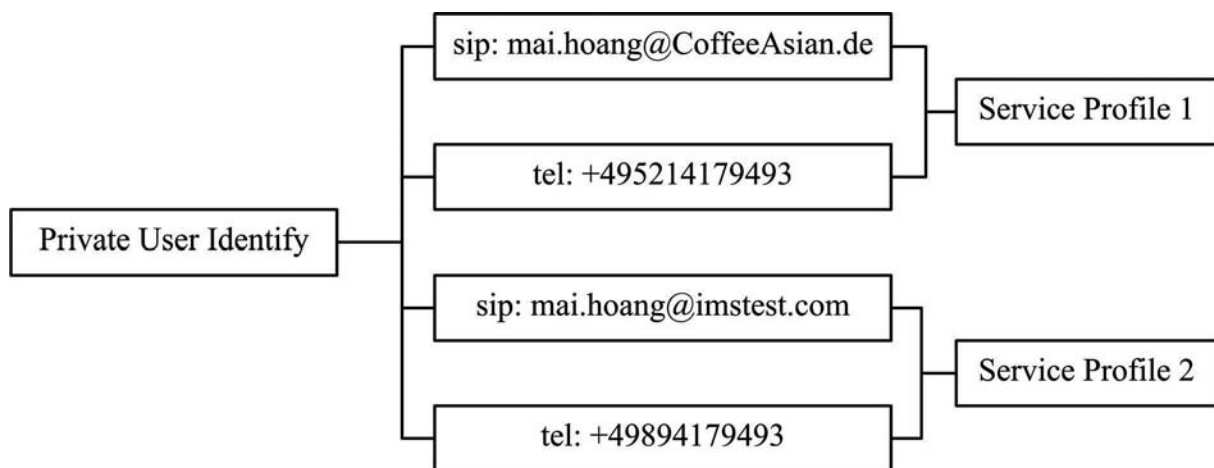


Figure 5-6: Relationship of the private user identity and public user identities

5.3.3.1.3 Public Service Identity

Public service identities are used to identify the IMS services, such as presence, messaging, conferencing and push to talk over cellular that are hosted by application servers. Public service identities are in a SIP URL or of in a tel URL format. For example, for messaging services there could be a public service identify (e.g. sip:messaging_mai@CoffeeAsian.de) to which the users send messages and then the messages are distributed to other members on the messaging list by the Instance Messaging Application Server.

5.3.3.1.4 Globally Routable User Agent

Globally Routable User Agent URI (GRUU) is a specific identifier that must be used to reach a particular IMS UE. For example, user Mai has a shared public user identity and her presence status indicates that she is willing to play games with UE1 and she is willing to accept a video session with UE2 then the GRUU

of UE1 can be used to establish a game session with Mai and the GRUU of UE2 can be used to setup a video session.

The relationship between UE, GRUU and public user identities [PM-2008] are shown in figure 5-7.

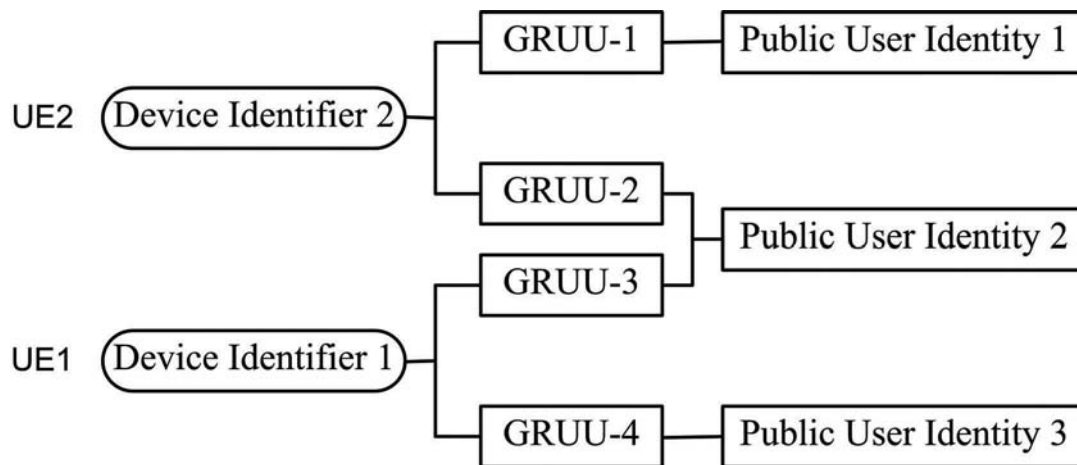


Figure 5-7: Relation between UE, GRUU and Public User Identities

5.3.3.2 P-CSCF Discovery

As already mentioned, a UE needs first access to the IMS network to start receiving e-mail, voice mail or phone calls. Since this access is achieved through the P-CSCF, the UE must find the P-CSCF within its present domain. This is done with the P-CSCF discovery mechanism.

The P-CSCF discovery is the procedure by which an IMS UE (also called IMS terminal) obtains the IP address of a P-CSCF, which acts as an outbound/inbound SIP proxy server toward the IMS UE. The P-CSCF discovery can be done in three different ways – using static IP address assignment, using the GPRS (General Packet Radio Service) procedure, and using the DHCP DNS that returns the domain name for the appropriate P-CSCF that serves the area the UE is located. These mechanisms will be described in the following.

1. *Using static IP address assignment.* An easy method is to configure either the IP address of the P-CSCF or the P-CSCF name in the UE. This IP address is then fixed and can only be changed by an administrator.
2. *Using the GPRS procedure.* In the GPRS procedure the UE sends the PDP (packet data protocol) context activation request with the P-CSCF address request flag is set. The UE receives the IP address of the P-CSCF in the PDP context activation response [3GPP TS 24.008].
3. *Using the Dynamic Host Configuration Protocol (DHCP) DNS procedure.* In the DHCP DNS procedure, the UE first establishes an

IP-CAN (IP connectivity Access Network) connection and sends a DHCP query to the IP-CAN (e.g. GPRS), which passes the request to a DHCP server. The UE then obtains a list of available P-CSCFs IP addresses, the used transport protocols and the corresponding port numbers in the DHCP response message. When domain names are returned, the UE needs to perform a DNS query to resolve the given P-CSCF domain name to get the IP address of the P-CSCF. The DHCP DNS procedure is described in the figure 5-8.

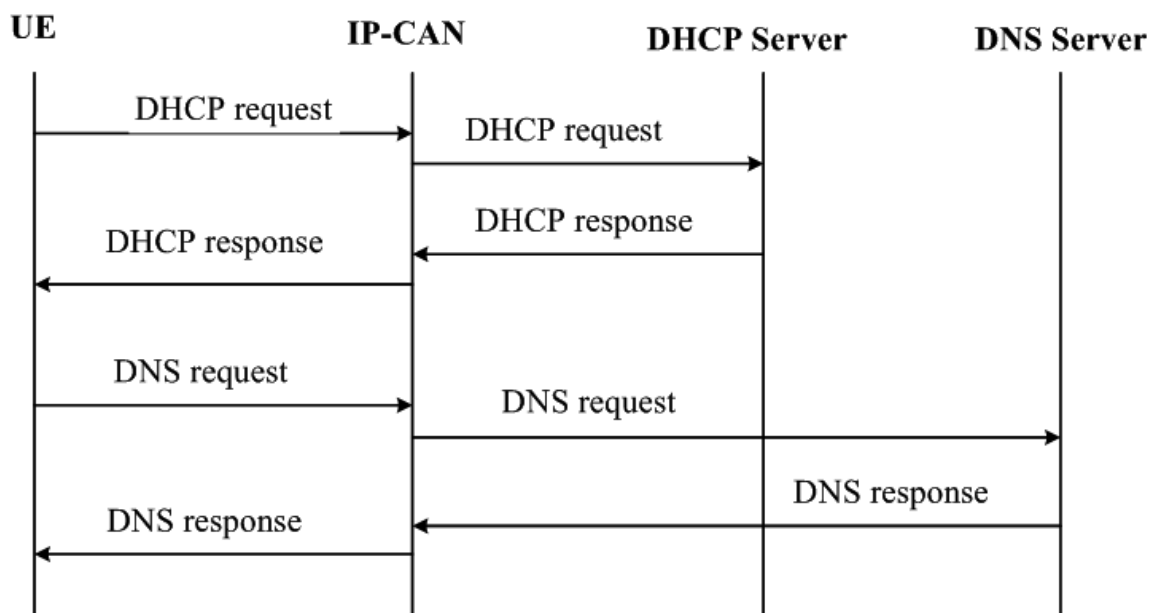


Figure 5-8: Discovering P-CSCF with the DHCP DNS procedure

5.3.3.3 IMS Session Control

There are a lot of dialog forms we can use for our communication in a IMS network, such as using a cell phone to make voice calls, peaking up the phone at home and placing calls, or using a DSL connection to surf the internet. Any time a user wishes to establish any form of one of these dialogs, a session must be created. A session can be thought as a portion of dialog between two parties. For example, in a video conference, the voice stream of the transmission would be one session, while the video stream would be a completely different session. Thus session control is needed.

The IMS session control includes subscriber registration, session initiation, session termination and session modification. These mechanisms will be discussed in this section. The IMS uses SIP for controlling all sessions within the IMS domain. Therefore the rules of SIP registration, session initiation and of session termination described in session 3.9.3.3 are applied in IMS. In addition,

there are some extensions to SIP that have been defined by 3GPP specifically for use within the IMS domain to make the communication more robust and secure.

5.3.3.3.1 Initial Registration

The IMS registration is the procedure where an IMS subscriber requests authorization to use the IMS services in the IMS network. The IMS network authenticates and authorizes the subscriber to allow him to have the access to the IMS network. IMS registration includes initial registration, re-registration and de-registration. While a initial registration is used to register a new SIP session in IMS, the re-registration is applied to extend a ongoing SIP session and the de-registration is used to remove a ongoing session. In this section, only initial registration will be addressed.

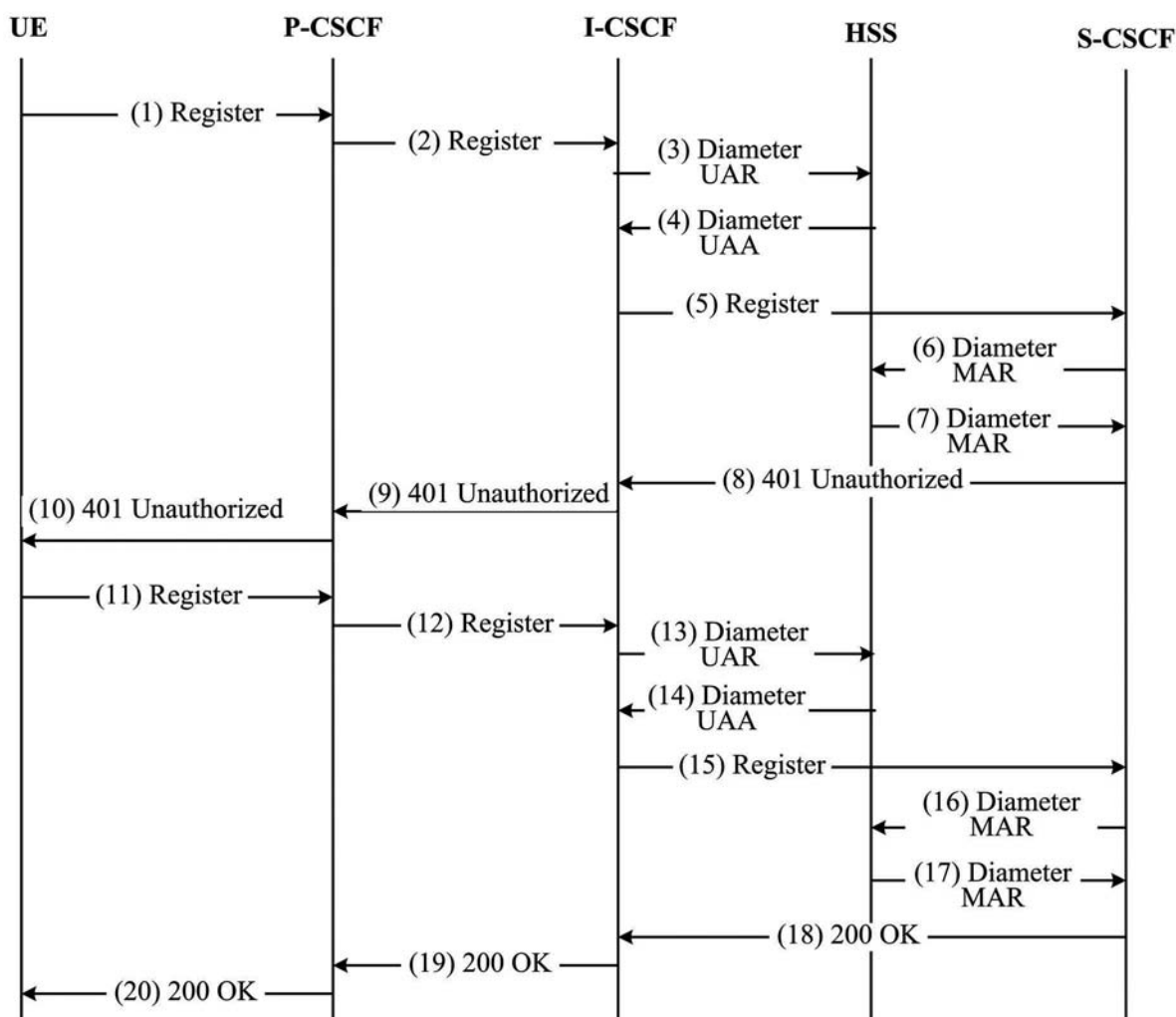


Figure 5-9: Main principle of the IMS registration

Figure 5-9 describes the main principle of an IMS registration. The IMS functions involved in the IMS registration process are P-CSCF, I-CSCF, S-CSCF and HSS. The IMS registration is initiated by a SIP REGISTER request, and completed by receiving a 200 OK message at the IMS UE. The registration process includes 20 SIP messages. Each of them is indicated by a number shown in figure 5-9.

```
REGISTER sip:ims-test.com SIP/2.0
Via:SIP/2.0/UDP
1.1.1.1:5060;branch=z9hG4bK004c301fd16bdf1181b6005056c00008;rport
From: <sip:495214179493@ims-test.com>;tag=1939515614
To: <sip:495214179493@ims-test.com>
Call-ID: 0005BB36-D06B-DF11-81B2-005056C00008@195.71.5.151
CSeq: 1 REGISTER
Contact: <sip:495214179493@1.1.1.1:5060>;Expires=0
Authorization: Digest username="hoang1234@ims-
test.com",realm="imstest.de",
nonce="2a8279b485d663ffa7c0cee5206159d3",uri="sip:ims-test.com", re-
sponse="38a9f7789365bf9ff9569e20bfd6eebb",algorithm=MD5,
cnonce="234abcc436e2667097e7fe6e1a53e8dd", qop=auth, nc=00000001
User-Agent: SIPPER for PhonerLite
Expires: 0
Content-Length: 0
```

Figure 5-10: (1) Register

5.3.3.3.1.1 SIP REGISTER sent from UE to P-CSCF

After obtaining the IP address of the P-CSCF via the P-CSCF discovery procedure, the IMS terminal (IMS UE) initiates a SIP REGISTER request to the P-CSCF ((1) in figure 5-9), which relays the REGISTER to the I-CSCF located in the home network.

The SIP REGISTER request sent from UE contains four parameters: the registration URI, the public user identity, the private user identity and the contact address. The content of a SIP REGISTER sent from an IMS UE to the P-CSCF is shown in figure 5-10.

- *The registration URI.* This is a SIP URI, which identifies the home network domain used to address the SIP REGISTER request. In figure 5-10, the registration URI is “sip:ims-test.com”
- *The public User Identity.* This is a SIP URI used to represent the subscriber ID under registration. The public user identity is included in the from header field. In the example in figure 5-10 is “sip:495214179493@ims-test.com”.

- *The Private User Identity.* This identity is used for authentication purposes. It is included in the user name parameter of the authentication header field, which is included in the SIP REGISTER request.
- *The Contact Address.* This is a SIP URI that includes the IP address of the IMS UE (terminal) or the host name where the subscriber is reachable. This contact address is found in the sip contact header field in the SIP REGISTER request.

5.3.3.3.1.2 SIP REGISTER sent from P-CSCF to I-CSCF

The P-CSCF needs to locate an entry point into the home network by executing the DNS procedures, which provide the P-CSCF with SIP URI of an I-CSCF. The P-CSCF then inserts a P-Visited-Network-ID that contains an identifier of the network where the P-CSCF is located. This SIP header field is used at the home network for validating the existence of roaming agreement between the home and the visited network. The P-CSCF also inserts a path header field with its own SIP URL to request the home network to forward all SIP requests through this P-CSCF. The P-CSCF then forwards this SIP REGISTER request to the assigned I-CSCF in the home network (see the second SIP REGISTER in the figure 5-9).

5.3.3.3.1.3 DIAMETER user request and answer sent between I-CSCF and HSS

After receiving the SIP REGISTER request from P-CSCF, the I-CSCF extracts the public user identities, private user identity and the visited network identifier from this SIP request and sends them within a Diameter User Authentication Request (UAR) to the HSS ((3) in figure 5-9). The HSS authorizes the user to roam the visited network and validates that the private user identity is allocated to the public user identity under registration. The HSS answers with a Diameter User-Authentication-Answer (UAA), (4) in the figure 5-9. The HSS also adds the SIP URL of a previously allocated S-CSCF in the Diameter UAA message, if there was an S-CSCF already allocated to the user. By the first registration, the HSS returns a set of S-CSCFs so that the I-CSCF can use as input for selecting an assigned S-CSCF. After receiving the UAA the I-CSCF selects an appropriate S-CSCF for forwarding the REGISTER request.

5.3.3.3.1.4 REGISTER sent from I-CSCF to the S-CSCF

After selecting an appropriate S-CSCF, the I-CSCF continues with the process by proxying the SIP REGISTER request to the selected S-CSCF, (5) in figure 5-9.

5.3.3.3.1.5 Diameter Multimedia-Authentication-Request (MAR) and Diameter Multimedia-Authentication-Answer (MAA)

After receiving the REGISTER request from I-CSCF, the S-CSCF needs to save the S-CSCF URL in the HSS for further query to the HSS for the same subscriber. Moreover, the S-CSCF needs to download the authentication data from the HSS to perform authentication for this particular subscriber. To achieve it, the S-CSCF sends a Diameter Multimedia-Authentication-Request (MAR) to the HSS, (6) in the figure 5-9. The HSS save the S-CSCF URL in the subscriber data and responses with a Diameter Multimedia-Authentication-Answer (MAA), which consists of one or more authentication vectors that are used at the S-CSCF for authenticating the subscriber, (7) in the figure 5-9.

5.3.3.3.1.6 401 Unauthorized Response

After receiving the MAA, the S-CSCF sends the “401 Unauthorized” toward IMS UE via I-CSCF and P-CSCF, (8), (9) and (10) in the figure 5-9.

5.3.3.3.1.7 Second SIP REGISTER

When a IMS UE receives an “401 unauthorized” response from the P-CSCF, it recognizes it as a challenge and thus initiates a new SIP REGISTER to the P-CSCF, (11) in figure 5-9. The P-CSCF does the same action as for the first REGISTER request: determining the entry point, finding an I-CSCF in the home network and then forwarding the REGISTER request to the selected I-CSCF.

5.3.3.3.1.8 New DIAMETER UAR and UAA sent between I-CSCF and HSS

I-CSCF sends a new Diameter UAR message, (13) in figure 5-9, for the same reason as described for the first Diameter UAR message. The difference to the first Diameter UAA message is the second Diameter UAA message includes routing information: SIP URI and the S-CSCF allocated to the user.

5.3.3.3.1.9 Second SIP REGISTER sent from I-CSCF to S-CSCF

Because the HSS stored the URI when it received a Diameter MAR message (6), therefore the second REGISTER request ends up in the same S-CSCF, which is allocated to the user at the time of the registration. The S-CSCF validates the credentials in the REGISTER messages

5.3.3.3.2 Basic Session Establishment

The IMS basic session establishment is a procedure to setup a SIP session in the IMS network. Depending on participants, there exist 3 different basic session

setups: (1) IMS UE to IMS UE, (2) IMS UE to PSTN UE and (3) IMS UE to PLMS UE. For the sake of simplicity, we only focus on the session setup from an IMS UE and to an IMS UE.

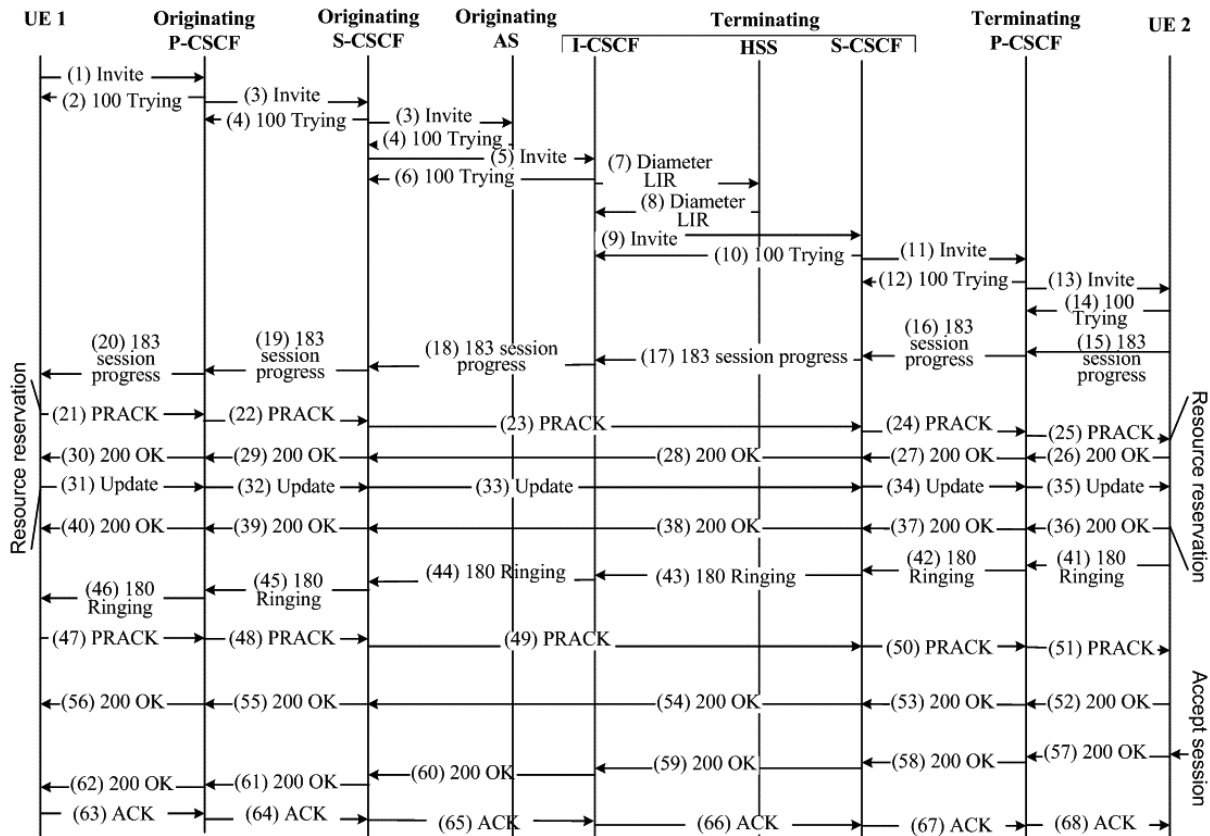


Figure 5-11: SIP basic session setup

Figure 5-11 shows a description flow chart of the signalling sequences involved in a basic SIP session setup between the UE 1 and UE 2. As we see in this figure, there are many functional components involved in setting up the session. We assume that the UE 1 and UE 2 belong to different home networks, the originating and terminating home network. Also for simplicity, we call the originating P-CSCF and originating S-CSCF as the P-CSCF and S-CSCF that are serving the caller. Similarly, we call the terminating P-CSCF and terminating S-CSCF as the P-CSCF and S-CSCF that are serving the callee.

The P-CSCF must be present in all the signalling exchanged with the UE because it is the first entry point of a UE to the IMS core. The S-CSCF is traversed in all requests to allow the triggering of services requested from the UE. The S-CSCF plays an important role in service provision by involving one or more application servers. As described in the previous session. The diameter interaction between I-CSCF and HSS is also found in the figure 5-11.

Remark in the figure 5-11 is that the “183 session progress” flowing from UE 2 back to the UE1, starting after “100 Trying” (message (14)) are not shown in this figure. Also the “PRACK” messages sent from caller UE (UE1) toward to the callee’s UE (UE2 as responses to the “183 session progress” are not assigned in the figure 5-10. For simplify, the charging messages sent from S-CSCF to the mediation node are not described in this figure.

```

INVITE sip:495214179493@ims-test.com SIP/2.0
Via: SIP/2.0/UDP
1.1.1.1:5060;branch=z9hG4bK80a7409a1ca5df119dc5005056c00008;rport
From: "Mai Hoang" <sip:495246333333@ims-test.com>;tag=441055954
To: <sip:495214179493@ims-test.com>
Call-ID: 80A7409A-1CA5-DF11-9DC4-005056C00008@1.1.1.1
CSeq: 6 INVITE
Contact: <sip:495246333333@1.1.1.1:5060>
Content-Type: application/sdp
Allow: INVITE, OPTIONS, ACK, BYE, CANCEL, INFO, NOTIFY, MESSAGE,
UPDATE
Max-Forwards: 70
Supported: 100rel, replaces
User-Agent: SIPPER for PhonerLite
P-Preferred-Identity: <sip:495246333333@ims-test.com>
Content-Length: 395
v=0
o=- 2032832383 0 IN IP4 195.71.5.196
s=SIPPER for PhonerLite
c=IN IP4 1.1.1.1
t=0 0
m=audio 5062 RTP/AVP 8 0 2 3 97 110 111 9 101
a=rtpmap:8 PCMA/8000
a=rtpmap:0 PCMU/8000
a=rtpmap:2 G726-32/8000
a=rtpmap:3 GSM/8000
a=rtpmap:97 iLBC/8000
a=rtpmap:110 speex/8000
a=rtpmap:111 speex/16000
a=rtpmap:9 G722/8000
a=rtpmap:101 telephone-event/8000
a=fmtp:101 0-16
a=sendrecv

```

Figure 5-12: Example for an INVITE request (1) from UE1 to P-CSCF

5.3.3.3.2.1 Handling the INVITE requests

An example of the Invite message sent from UE1 to the P-CSCF is shown in the figure 5-12. The request URI contains the public User Identity of the intended destination. In this example, this public user identify is phone number “495214179493” that belongs to the ims-test.com. The “via” header field

contains the IP address and port number where the UE will receive the response to the INVITE request. After receiving the INVITE, the P-CSCF will response the recipient addressed with the IP address and with the port number involved in the via header field. The “via” header field also indicates the transport protocol used to transport the SIP messages to the next node. The P-Preferred-Identity field indicates which one of the public user identities should be used for this SIP session if the user has several public user identities. In this example, the identity 495246333333 is used. The content-type and content-length header field indicate that the accompanying body is an SDP body of a certain length. The lines following the content-length header field line belong to the SDP body. The “c=” line indicates the IP address the UE1 wants to establish one media stream, indicated the presence of one “m=” line, the audio stream. Also the UE 1 indicates the support for a lot of codec, such as PCMA/8000, PCMU/8000 etc. We also observe the presence of a few attributes that indicate the current and desired local QoS.

- *Handling the INVITE request at the originating P-CSCF.* When the P-CSCF received the INVITE request (1), the P-CSCF verifies that the UE 1 is acting correctly according to the IMS routing requirements. The P-CSCF also inspects the SDP offer, because some media parameters are not allowed in the network. Then the P-CSCF checks whether the P-Preferred-Identity header field is involved in the INVITE request, and verify the values in this header field. During the registration, the P-CSCF learns all public user identities registered to the UE. It deletes the P-Preferred-Identity header field and inserts a P-Asserted-Identity header following the RFC 3325. The P-Asserted-Identity header field is set to a registered public user identity. The P-CSCF removes and modifies the header relating to the security agreement. P-CSCF inserts the charging header, recording the routes. Finally the P-CSCF sends the modified SIP INVITE requests to the S-CSCF. An example of the INVITE sent from P-CSCF to the S-CSCF is shown in the figure 5-13 below.

```

INVITE sip:495214179493@ims-test.com SIP/2.0
Via: SIP/2.0/UDP
2.2.2.2:5070;branch=z9hG4bKq38lrc101g2h8eulv0u0.1
Via:SIP/2.0/UDP1.1.1.1:5060;received=1.1.1.1;branch=z9hG4bK80a7
409a1ca5df119dc5005056c00008;rport=5060
From: "Mai Hoang" <sip:495246333333@ims-test.com>;tag=441055954
To: <sip:495214179493@ims-test.com>
Call-ID: 80A7409A-1CA5-DF11-9DC4-005056C00008@1.1.1
CSeq: 6 INVITE
Contact: <sip:495246333333-
ubdq76q83j7i4@10.244.0.132:5070;transport=udp>
Content-Type: application/sdp

```

```

Allow: INVITE, OPTIONS, ACK, BYE, CANCEL, INFO, NOTIFY,
MESSAGE, UPDATE
Max-Forwards: 69
Supported: 100rel, replaces
User-Agent: SIPPER for PhonerLite
Content-Length: 396
P-Asserted-Identity: <sip:495246333333@ims-test.com>
Route: <sip:scscf01.imstest.mai.com:5060;lr>
P-Visited-Network-ID: imstest2.mai.de
P-Charging-Vector: icid-
value=mgv40046ghb43qg6e1csioc6i9lsk4lee3t4nqdekbp86nge4bb0jos04
-4; icid-generated-at=2.2.2.2
v=0
o=- 3243707894 0 IN IP4 3.3.3.3
s=SIPPER for PhonerLite
c=IN IP4 3.3.3.3
t=0 0
m=audio 11040 RTP/AVP 8 0 2 3 97 110 111 9 101
a=rtpmap:8 PCMA/8000
a=rtpmap:0 PCMU/8000
a=rtpmap:2 G726-32/8000
a=rtpmap:3 GSM/8000
a=rtpmap:97 iLBC/8000

a=rtpmap:110 speex/8000
a=rtpmap:111 speex/16000
a=rtpmap:9 G722/8000
a=rtpmap:101 telephone-event/8000
a=fmtp:101 0-16
a=sendrecv

```

Figure 5-13: Example for an INVITE request (3) from P-CSCF to S-CSCF

- *Handling the INVITE request at the originating S-CSCF.* When the originating S-CSCF assigned to the caller receives the INVITE request from the P-CSCF, it examines the P-Asserted-Identity header to identify the user that originated the INVITE request. The S-CSCF then downloads the authentication vector from HSS and uses this information to determine whether the INVITE request has to traverse one or more application servers. The originating S-CSCF is the first node that has to route the SIP request based on the destination (the callee) in the request-URI. If the originating S-CSCF finds a SIP URI in the request-URI, the regular SIP routing procedures described in RFC 3263 is applied. Basically, for a given domain name, an S-CSCF has to discover an I-CSCF in this domain name. The procedure is done via DNS queries. Once the S-CSCF has found the sip server I-CSCF, it adds a new value (the public user identity) to the existing P-Asserted-Identity header field. The originating S-CSCF sends the INVITE request to the application servers and to the I-CSCF.

- *Handling the INVITE request at the terminating I-CSCF.* The I-CSCF in the destination home network receives the INVITE request (request (5) in the figure 5-11) from the originating S-CSCF. The I-CSCF recognizes the callee identified in the request-URI of the INVITE request. The I-CSCF then has to forward the request to the S-CSCF assigned to this callee. The I-CSCF discovers the assigned S-CSCF via queries the HSS with the Diameter Location Information request (LIR) message.
- *Handling the INVITE request at the terminating S-CSCF.* The S-CSCF in the terminating network that takes care of the callee receives the INVITE request ((9) in the figure 5-11). The S-CSCF first identifies the callee in the Request-URI header of the INVITE request. It then evaluates the initial filter criteria of the called user. The S-CSCF is looking for services that should be applied to session setup toward the UE. To forward the INVITE request to the callee's UE, the S-CSCF must know a set of proxies the INVITE request will traverse to reach the callee's UE. This set of proxies will always include the P-CSCF and may include one or more I-CSCFs. As mentioned in the previous sessions, the S-CSCF learn the set of proxies during the registration process of the callee. Therefore, the S-CSCF creates a new Request-URI involving the content of contact header field value registered by callee during the registration. Finally it sends the INVITE request to the terminating UE.

5.3.3.3.2.2 Handling the “183 Session Progress” requests

The 183 session progress traverses the same proxies that the corresponding INVITE request traversed.

- *Action at the callee's UE.* The INVITE request in the figure 5-11 is received at the callee's UE (UE 2 in the figure). This INVITE request ((13) in figure 5-11) involves an SDP offer generated in the caller's UE. The SDP offer indicates the IP address and port number where the caller wants to receive media streams, the desired and allowed codecs for each of the media stream. Some concept requires that the callee's UE responses with the “183 Session Progress” that contains the SDP answer. With “183 session progress” the callee's UE starts resource reservation. If several codecs are possible, it would need to reserve resource for high demand codec. The callee's UE forwards the message to the P-CSCF.
- *Action at the terminating P-CSCF.* When the P-CSCF receives the “183 session progress” ((15) in figure 5-11), it verifies the correctness of the message, such as that the via and record-route header must contain the value the callee's UE must use in the response to the INVITE request. If the values are not as expected, the P-CSCF discards the response or

rewrites the new values in the header. The P-CSCF inserts a P-Assert-Identity header field, whose value is the same as that included in the P-Called-Party-ID header field of the INVITE request arrived at this P-CSCF before ((11) in figure 5-11). Finally, P-CSCF forwards the 183 session progress to the S-CSCF, (16) in figure 5-11.

- *Action at the terminating S-CSCF.* The S-CSCF receives the 183 session progress response (17). It removes the P-Access-Network-Info header field and forwards the message to the terminating I-CSCF.
- *Action at the I-CSCF.* The I-CSCF does not take any action on the receiving the 183 response, it forwards the response to the originating S-CSCF.
- *Action at the originating S-CSCF and P-CSCF.* The S-CSCF receives the 183 response and may remove the P-Asserted-Identity header field if there is a private requirement for it. The S-CSCF then forwards the 183 response to P-CSCF. The P-CSCF then forwards it to the caller's UE .
- *Action at the caller's UE.* When the caller's UE receives the 183 response, it focuses on the SDP answer that contains the IP address and the port number of the remote UE. SDP also includes an indication of whether the callee accepted establishment of a session with these media streams. The SDP answer also contains an indication from the callee that it want to be notified when caller's UE has completed the resource reservation process. The UE then creates a new SDP offer and adds it to the new PRACK. The caller's UE starts with the resource reservation process, beginning with the PRACK (21) and ending with the UPDATE. If several codecs were negotiated, the UE will reserve the maximum bandwidth required by the most demanding codec.

5.3.3.3.2.3 Handling the PRACK requests

The PRACK requests traverse the proxies that are requested to remain in the path. These proxies typically will be a subset of the proxies that the INVITE requests traverse. The path is determined in the Route header involved in the PRACK request.

The response to the "183 session progress" is the "PRACK" initiated by the caller's UE and sent to the callee's UE. This PRACK visits the same proxies as the "183 session progress" had visited. The response to the "PRACK" is "200 OK" (26) sent from callee's UE to the caller. This "200 OK" response is just confirmation of the media stream and codecs of the session. The "200 OK" response to the PRACK request traverses the same set of the SIP proxies that the PRACK request traversed.

When the “200 OK” (30) response arrives at the caller’s UE, the caller’s UE is almost involved in its resource reservation process. Once the caller’s UE has got the required resource from the network, it sends an UPDATE request containing another SDP offer, in which the caller’s UE indicates that the resources are reserved at his local segment. This UPDATE request visits the same set of proxies as the PRACK request.

When the callee’s UE receives the UPDATE request, it will generate a “200 OK” response, (36) in figure 5-11. At this time, the callee’s UE may have already finished his resource reservation or not, which is indicated in its own local QoS status. This “200 OK” response follows the same path as the UPDATE request

5.3.3.3.2.4 Handling the “180 Ringing” SIP message

- *Action at the callee’s UE.* The response to the “PRACK” is “180 Ringing” sent from callee’s UE to the caller. The “180 ringing” traverses those proxies the INVITE request traversed. This SIP message is created when the callee’s UE rings. This response typically does not contain SDP, since all session parameters (codecs, media streams, etc) have been already negotiated in the previous exchanges via “183 session progress” and “PRACK”.
- *Action at caller’s UE.* When the caller’s UE receives the “180 ringing” response (20), it will generate a ring-back tone to indicate to the caller. The response to the “180 ringing” is a PRACK request generated at the caller’s UE and sent to the callee’s UE. The PRACK request traverses the same proxies as the previous PRACK and UPDATE requests.

5.3.3.3.3 Basic Session Termination

A SIP session can be terminated from either caller’s or callee’s UE. This is done by using the BYE message from a UE to the other UE in the SIP session.

```

BYE sip:05214179493@3.3.3.3:5060;transport=udp SIP/2.0
Via:SIP/2.0/UDP
2.2.2.2:5060;branch=z9hG4bK805d1d13573be0118c2f001de08aa467;rport
From: "Mai Hoang" <sip:4952417414019@ims-test.com>;tag=2290385219
To: <sip:05214179493@ims-test.com>;tag=1724274314-1298203380432
Call-ID: 00B95D0B-573B-E011-8C2E-001DE08AA467@2.2.2.2
CSeq: 5 BYE
Contact: <sip:4952417414019@2.2.2.2:5060>
Max-Forwards: 70
User-Agent: SIPPER for PhonerLite
Content-Length: 0

```

Figure 5-14: a simple BYE message sent from UE to the P-CSCF

Each participated UE will then respond with a “200 OK” message. A simple BYE message sent from a UE to the P-CSCF is shown in the figure 5-14 below.

There are some situations where the S-CSCF must terminate a session in progress. In these cases, S-CSCF sends a BYE message in two directions: the originator and the called party. The S-CSCF then expects to receive a 2xx response from both parties.

5.3.3.3.4 Basic Session Modification

Any established SIP session can be modified while the session is in progress. For example, if the originator UE wants to add video into the call during a conference call, the originator sends a new INVITE request (or a UPDATE) to each participated UE. This new request will identify the participated UEs and the media to be added and any other modifications to be made. The UEs must accept the new request by sending a successful response. Otherwise the session modification request is rejected.

5.3.3.4 S-CSCF Assignment

Section 5.3.3.2 describes how the UE discovers the P-CSCF as the IMS entry point. The next entry of a signaling session is the S-CSCF. There exist three situations when the S-CSCF assignment is required [MG-2008]: (1) During registration (when a UE registers with the network); (2) When S-CSCF is needed to execute services on behalf of unregistered UE; (3) when a previously assigned S-CSCF is out of service.

- *S-CSCF Assignment during Registration.* When an IMS subscriber is registering with an IMS network, the UE sends a REGISTER request to the assigned P-CSCF, which finds the I-CSCF for this subscriber. By exchanging the messages with the HSS, the I-CSCF obtains a set of S-CSCFs (also called S-CSCF capability information [3GPP TS29.228, TS 29.229]). This capability information is transferred between HSS and I-CSCF within the Server-capability Attributes Value Pair (AVP) that contains information about mandatory, optional capability AVPs and server-name AVP. Based on this information, the I-CSCF then selects a suitable S-CSCF for this subscriber.
- *S-CSCF Assignment to Execute Services for unregistered User.* If the HSS knows that no S-CSCF is currently assigned and that the user has services related to unregistered state, it then sends the S-CSCF capability information to the I-CSCF. I-CSCF then selects a suitable S-CSCF for this subscriber as described for the S-CSCF assignment during registration.

- *S-CSCF Assignment when a previously assigned S-CSCF is out of service.* When the I-CSCF recognizes that it cannot reach the assigned S-CSCF, it sends the Diameter User Authentication Request (UAR) message to the HSS and sets the type of authentication information to the value registration and capabilities.. After obtaining the S-CSCF capability information, the I-CSCF performs the S-CSCF assignment as described for the S-CSCF assignment during registration.

5.3.3.5 AAA in the IMS

Authentication, authorization, and accounting (AAA) deals with mechanisms for intelligently controlling access to computer resources, enforcing policies, auditing usage, and providing the information necessary to bill for services. Based on the node functionalities of IMS, we separate the description of authentication and authorization from the description of accounting. While the first part will be addressed in section 5.3.3.5.1, the second part will be described in the section 5.3.3.5.2.

5.3.3.5.1 Authentication and Authorization

The authentication and authorization are performed in IMS through three interfaces: the Cx, Dx, and Sh interfaces [3GPP TS 29.228, TS 29.229, TS 29.328, TS 29.329] (figure 5-15).

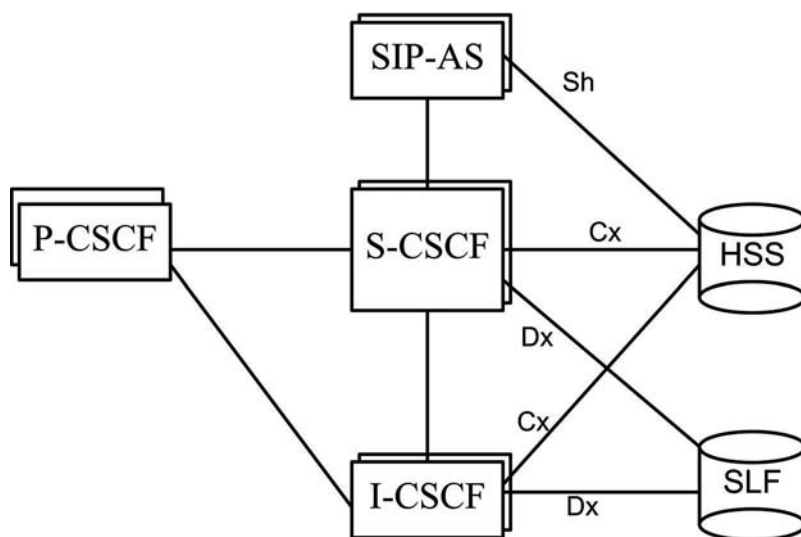


Figure 5-15: Interfaces for authentication and authorization

The Cx interface is used between a HSS and either an S-CSCF or an I-CSCF. The Dx interface is specified between a SLF and either an S-CSCF or an I-CSCF. The Sh interface is used between the SIP application server

The IMS charging architecture specified in [3GPP TS 32.240, TS 32.200, TS 32.225] includes two charging models: offline charging and online charging. Offline charging is applied to users who pay for their services periodically. Online charging is used for prepaid services and applied to users who need to have money in their account before consuming services. That means prepaid services require Online Charging Support (OCS), which must be checked before allowing users to use the services. OCS is responsible for interacting in real time with the user's account and for controlling or monitoring the changes related to the services.

Figure 5-16 shows a high level IMS charging architecture [3GPP-TS32.240, MG-2006]. This figure shows that all IMS SIP functions are communicating with the offline charging entity – the Charging Data Function (CDF) – by using Diameter-based Rf interface [3GPP TS 32.299]. After the CDF receives the Diameter requests from IMS entities and from access functions, it creates Charging Data Records (CDRs) and sends to the Charging Gateway Function (CGF) via the Ga interface [3GPP TS 32.295]. The CGF processes the CDRs and delivers the final CDRs to the billing system using the Bx interface [3GPP TS 32.240]. In comparison with the offline charging, online charging only deals with three IMS functions (SIP AS, NRFC and S-CSCF) that communicate with OCS via Diameter-based Ro interface. OCS receives the diameter requests from these three entities; it processes the requests and creates CDRs, which are sent to the billing system.

In addition to the interfaces described in the figure 5-16, the IMS entities exchanges the SIP messages and take actions based on the SIP message header information. There are two SIP header fields specified in RFC 3445 [MHM-2003] that used to carry charging-related information in the IMS: P-Charging-Vector and P-Charging-Function-Address.

- *P-Charging-Vector*. The P-Charging-Vector is used to transfer charging related correlation information. Three types of this information are included in the P-Charging-Vector: Charging Identity (ICID) value, the address of SIP proxy that creates the ICID value, and the Inter Operator Identifiers (IOI). ICID is a globally unique charging value used to identify a dialog or a transaction outside a dialog. IOI is used to identify the originating and terminating networks involved in a SIP dialog. There may be an IOI generated from each side of the dialog to identify the network associated with each side. Figure 5-17 shows an example of P-Charging header within an INVITE message sent from a PGW to an IBCF (NNI SBC) within a PSTN to SIP call flow. The ICID value, the address of SIP proxy that creates the ICID value and the IOI value of the originator are displayed in this P-Charging-Vector.

```

INVITE sip:+4952117414019@ibcf-test.ims.com:5060;user=phone
SIP/2.0
Via: SIP/2.0/UDP
ims.sip.mgc.voip.abc.de:5060;branch=z9hG4bKterm-49458-
+4952117414019-+495266701614-95101
From: +495266701614
<sip:+495266701614@ims.sip.mgc.voip.abc.de:5060;user=phone>tag=
1727288113
To: +4952417414019 <sip:+4952417414019@ibcf-
test.ims.com:5060;user=phone>
Call-ID: 5577b7a0-142cde5b-49d4a1ed-
6969@ims.sip.mgc.voip.abc.de
CSeq: 1 INVITE
Max-Forwards: 18
Supported: timer
Session-Expires: 1800
Min-SE: 1800
Contact: +495266701614
<sip:+495266701614@ims.sip.mgc.voip.abc.de:5060;transport=udp>
Allow: IN-
VITE,ACK,PRACK,SUBSCRIBE,BYE,CANCEL,NOTIFY,INFO,REFER,UPDATE
P-Asserted-Identity: +495266701614
<sip:+495266701614@ims.sip.mgc.voip.abc.de;user=phone>
P-Charging-Vector: icid-value=ims.test.mgc-4bf288c6-4529cd6c-
8d-5da97355;icid-generated-at=ims.sip.mgc.voip.abc.de;orig-
ioi=abcDE
Content-Type: application/sdp
Content-Length: 673

```

Figure 5-17: P-Charging-Vector Example within an INVITE message

- P-Charging-Function-Address.** The P-Charging-Function-Address indicates the common charging functional entities used by each SIP proxy involved in a transaction to receive the generated charging records or charging events. There are two types of charging functional entities proposed by 3GPP: Charging Collection Function (CCF) and Event Charging Function (ECF). The CCF is used for off-line charging. ECF is used for online charging. To provide the network redundancy, there may be more than a single instance of CCF and ECF in a network. In the case there are more than a single instance of either CCF or ECF addresses, one of these instances is configured as primary and other is secondary. The charging data is then sent to the primary instance; if the primary instance is out of service, the data is sent to the secondary instance. Figure 5-18 shows an example of the P-Charging-Function-Address header within a “180 ringing” for a PSTN to SIP call. The content of the P-charging-Function-Address shows that it is a off-line charging with two

CCF, the primary CCF and the secondary CCF. The addresses of these CCFs are included in this header.

```
SIP/2.0 180 Ringing
To: "+4952117414019"<sip:4952117414019@ims.test.com:5060>;
tag=910460916-1274185933784
From: "+495266701614"<sip:+495266701614@ims.test.com:5060;
user=phone>;tag=1727288113
Call-ID: 5577b7a0-142cde5b-49d4a1ed-
6969@ims.sip.mgc.voip.telefonica.de
CSeq: 1 INVITE
Content-Length: 0
Via: SIP/2.0/UDP
1.1.1.1:5060;branch=z9hG4bKuiinur3030m162l8m7i0.1
Record-Route:
<sip:3Zqkv7%0BaGqmaaaaacqsip%3A4952417414019%40ims.test.com@scs
cfmtb01.ims.abc.com:5062;lr;maddr=10.1.172.52>
Contact: <sip:2.2.2.2:5060>
Allow: ACK, BYE, CANCEL, INFO, INVITE, OPTIONS, PRACK, REFER,
NOTIFY, UPDATE
Supported: timer
P-Asserted-Identity:
"MHuang"<sip:+4952117414019@ims.test.com;user=phone>
Privacy: none
P-Charging-Vector: icid-value=ims.test.mgc-4bf288c6-4529cd6c-
8d-5da97355;icid-generated-at=ims.sip.mgc.voip.abc.de;orig-
ioi=abcDE;term-ioi=1
P-Charging-Function-Addresses:
ccf="aaa://primaryCCF.ims.test.de:3868;transport=tcp";ccf="aaa:
//secondaryCCF.ims.test.de:3867;transport=tcp"
```

Figure 5-18: P-Charging-Function-Address within an INVITE message

5.3.4 IMS Services

Fundamental IMS services include presence, group management, push to talk over cellular, messaging, conference and multimedia telephony. In this section, some selected services will be summarized. Detail of IMS services are described in [MG-2008].

5.3.4.1 Presence

Presence is the service that allows a user to be informed about the reach-ability, availability and willingness to communicate with another user. This service is able to monitor whether the other users are online or offline. And if they are online, whether they are idle or busy. Presence service involves making the status of a user available to others and the statuses of others to this user. The presence information may include person and terminal availability,

communication presences, terminal capabilities, current activity, location, currently available services.

The IMS presence service allows the home network to manage a user's presence information, which may be obtained from the user and from the information supplied by network devices. The IMS presence service was introduced in 3GPP release 6 as a standalone service capability. Figure 5-19 shows the IMS presence service architecture defined in TS 23.141 [TS23.141]. The names of reference points between components are not displayed in this figure.

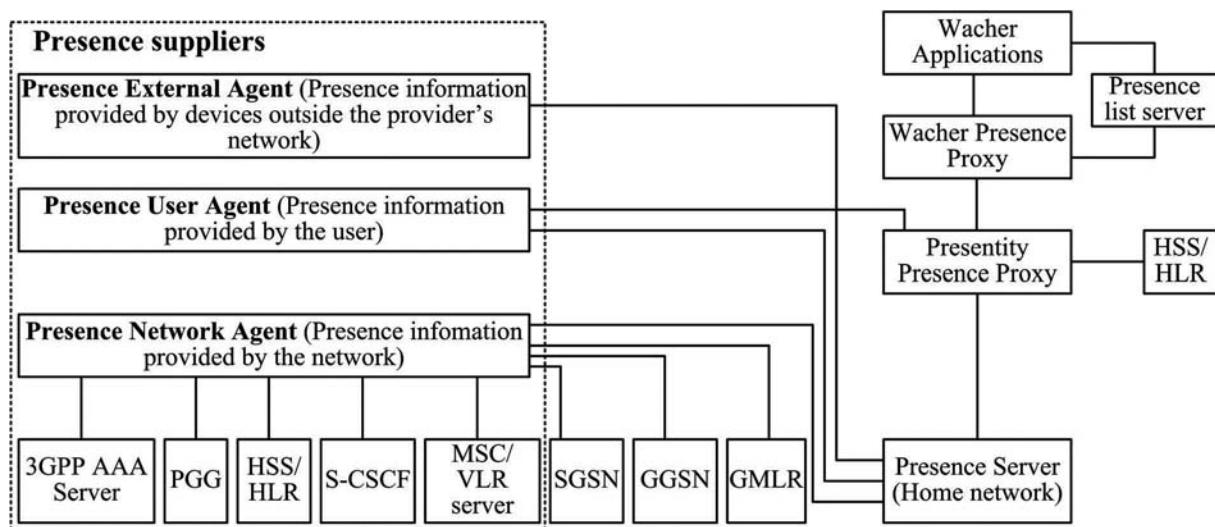


Figure 5-19: Reference architecture to support presence service [TS23.141]

The reference points between components in this reference architecture are:

- *Presence User Agent – Presence Server.* This reference point shall allow the presence user agent to manage subscription authorization policies.
- *Presence Network Agent – Presence Server.* This reference point shall allow presentity's presence information to be supplied to the presence server. It shall provide following mechanisms for the network agent: management of subscription authorisation policies, supplying/updating a certain subset of presentity, presence information to the presence server, activating/deactivating the report of presence information for a given presentity.
- *Presence External Agent – Presence Server.* This reference point shall allow presentity's presence information to be supplied to the Presence Server. It shall provide mechanisms for the presence External Agent to supply or update only a certain subset of the presentity's presence information to the Presence Server. The format of the presence information transported via this interface is specified in RFC 3863.

- *Watcher applications – Presentity Presence Proxy.* This reference point allows a Watcher application to request and obtain presence information.
- *HSS/HLR – Presence Network Agent.* This reference point allows the Presence Network Agent to query HSS/HLR about the state and status of a subscriber from the serving network (for 3GPP this is the CS domain or GPRS) and IMS perspective. It permits the Presence Network Agent to activate and deactivate the reporting of mobility management events from the serving network and/or the IMS-specific report from the S-CSCF.
- *S-CSCF – Presence network Agent.* The S-CSCF provides IMS-specific presence information (e.g. about IMS registration state). The mechanisms used for this reference point is defined in 3GPP TS 23.002.
- *Presentity Presence Proxy – HSS.* This interface assists locating the Presence Server of the presentity. It is implemented using the mechanisms defined for the Cx and Dx reference points as specified in TS 23.002
- *Presence network agent – GMLC.* This interface is used by the present network agent to obtain subscriber –related location information.
- *Presence Network Agent – SGSN.* This interface allows the SGSN to report the mobility management related events and mobility states (e.g. idle, connected) to the Presence Network Agent.
- *Presence Network Agent – MSC Server/VLR.* This interface enables MSC server/VLR to report mobility management related events, call related events, mobility states and call states to the presence network agent.
- *Presence network agent – GGSN.* This interface allows the GGSN to report presence relevant events to the Presence network Agent. The interface implementation is defined in TS 29.061.
- *Presence Network Agent – 3GPP AAA Server.* This interface allows the 3GPP AAA server to report the IP-connectivity related events to the presence Network Agent.
- *Presence User Agent – Presentity Presence Proxy.* This interface deals with mechanisms allowing the Present User Agent to supply or update a certain subset of the presentity's presence information to the presence server.
- *Watcher Applications – Presence List Server.* This interface enables a watcher application to manage presence list information in the presence list server.

Publishing and updating the presence information are initiated by the presence source UE that uploads this information using the SIP PUBLISH message sent from this UE to presence server. The SIP PUBLISH message is first passed to P-CSCF and S-CSCF before it arrives at the presence server. 200 OK as response to the SIP PUBLISH message is sent from presence server to

the presence source UE. This response is first passed S-CSCF and P-CSCF before it arrives at the presence source UE.

A watcher UE can obtain the presence information of other users by sending a SIP SUBSCRIBER request targeted to own presence list containing a list of users with presence information the watcher wants to discover. This request will be routed to the RLS (Resource List Server) that authorizes the watcher's subscription and extracts members of presence list and makes individual subscription to each presentity. The RLS accepts the subscription with 200 OK and sends empty NOTIFY message to the watcher. Once the RLS receives the presence information from the presence servers, it will deliver the NOTIFY request containing presentity's presence state to the watcher.

5.3.4.2 Messaging

IMS messaging is a service that allows a user to send some content to another user in near-real time. This service is one of today's most popular services. The content of an IMS message can be text message, a picture, a video clip or a music song. There are two different types of IMS messaging: page-mode and session-based mode.

The page-mode IMS messaging, or immediate messaging, was introduced in release 5 of the 3GPP specifications described in 3GPP TS 23.228 and TS 24.229. In the page-mode messaging, the SIP MESSAGE method [RFC3428] is used to send messages between IMS terminals in near-real time. The main goal of the page-mode messaging is to allow the S-CSCF or application servers to send short message to IMS terminals. Since MESSAGE method is implemented in the IMS terminal, users are able to send page-mode messages to other IMS users.

The session-based messaging was first introduced in release 6 of the 3GPP specifications described in 3GPP TS 24.247. It relates to the Internet Relay Chat (IRC) [RFC2810]. In the session-based messaging, the user takes part in session in which the main media component often consists of short textual messages. Each message session has a well-defined lifetime: a message session starts when session starts and stops when session is closed. After the session is set up, media then flows directly from peer to peer using the SIP and SDP between the participants. The Message Session Relay Protocol (MSRP) [RFC4975] is used for transmitting the messages within a session.

5.3.4.3 Push to Talk over Cellular

Push to talk over Cellular (PoC) is an IMS service that allows users to engage in immediate communication with one or more users connected real-time sessions.

The working principle is simple. Users select an individual user or a group of users they wish to talk to, and then press the “push to talk” key to start talking. The PoC service supports two-modes of PoC session establishment: the Pre-established session mode and the on-demand session mode. The PoC communication is a half duplex – while one participant speaks the other (s) only listen. Even PoC supports group communication; it is based on uni-casting and no multi-casting is performed. Each sending client sends data to a dedicated PoC application server. In the case of a group of users, the PoC application server then duplicates the traffic to all the recipients. The PoC service typically supports [OMA-2009, MG-2008]:

- *PoC Communication type.* PoC provides several types of communications, e.g. dial-out group communication, join-in group communication and chat group communication. The main differences between these communication types depend on the group policy und session setup.
- *Simultaneous PoC Sessions.* Difference to the traditional telephone service is that the PoC service allows subscribers to participate in more than one PoC session at the same time without placing any of session on hold. This capability is called simultaneous PoC session feature.
- *PoC Session Establishment Models.* There are two different session establishment models: on-demand and pre-established session. These models differ on their medial parameter negotiation. In a pre-established session model, PoC user establishes a session towards her participating PoC function and negotiates all media parameters prior to make request for PoC sessions to other PoC users. This model allows a PoC client to invite other PoC clients without negotiating the media parameters again. In an on-demand model, traditional SIP method is used (e.g. media parameters are negotiated when a user makes a request for a PoC session.
- *Incoming PoC Session Handling.* Two models have been defined for controlling the incoming PoC session: auto-answer model and manual answer model. When auto-model is configured, the PoC terminal will accept the incoming PoC sessions without waiting any actions from the PoC user. When manual answer model is turned on, a user must accept an incoming PoC session to the PoC server. After that the incoming media streams can be played immediately. Using auto-answer model would be useful feature. However, PoC users cannot be sure who may be the callers and therefore this model may not be comfortable for all possible PoC users. But using manual answer model all time is not suitable. In addition, a Poc user also wants to automatically refuse PoC sessions from some users or PoC groups. To solve these problems, access control mechanism was developed that is executed at the PoC server to perform the

participant role for the called PoC user. This access control enables a PoC user to allow or to block incoming PoC session from other PoC users or PoC groups. Moreover, the access control enables a PoC user to define users whose sessions are to be automatically accepted.

- *Instant Personal Alerts.* It deals with the mechanism to inform about the calling user's wish to communicate and to request the invited user to call-back. It is used when a caller user is not able to reach a recipient.
- *Group Advertisement.* This feature enables a PoC user to advertise a new created chat PoC group to the PoC users defined in this group. A group advertisement could be sent to one or more users or could be sent to all group membership using a SIP MESSAGE, which has PoC-specific content in form of a MIME (Multipurpose Internet Mail Extension) body.
- *Barring Features.* As described above, a PoC user can selectively block PoC incoming sessions using a pre-configured access control list. Additionally, a PoC user is able to initiate a PoC server to reject all new incoming PoC sessions. This feature is called incoming session barring.
- *Participant Information.* This feature allows a PoC user to request and obtain information about PoC session participants and their status in the PoC session.

5.3.4.4 Multimedia Telephony

IMS multimedia telephony deals with services that allow IMS users to establish communication between them and to use IMS supplementary services. The multimedia telephony services are not limited to always include speech; they also enable other media or combinations of media.

Establishing, handling and terminating a multimedia session between users of IMS multimedia telephony are performed via SIP methods. In addition to these SIP basic procedures, IMS multimedia telephony also provides communication service identification and telephony application server.

Following IMS supplementary services are provided in 3GPP release 7.

- *Calling Line Identification (CLI).* This service includes Calling Line ID Delivery (CLIP) and Calling Line ID Delivery Blocking (CLIR). While the CLIP enables a user to deliver his/her identity to the called user, the CLIR enables a user to block delivery of his/her identity to the called user.
- *Connected Line Identification (COL).* This service includes Connected Line ID Presentation (COLP) and connected Line ID Restriction (COLR). While the COLP allows the calling user to obtain the address information of the final connected user (the party causing the connect message transmission at the remote end, the COLR allows the final connected user

to block the delivery of its address information so that it will not be obtained by the calling user .

- *Incoming Calls.* This service deals with mechanisms to handle the incoming calls, e.g. Call Forwarding Always (CFU), maximum number of call forwarding, Call Forwarding Busy (CFB), Call Forwarding no Answer (CFNR), Call Forwarding Selective (CFS), Call Forwarding not Registered (CFNL), Anonymous Call Rejection (ACR), Voice2Mail and Fax2Mail.
- *Call Control.* This service includes Call Waiting, Call Hold, Music on Hold, Flash Call Hold, Three Way Call and Call Completion on Busy.
- *Call Barring.* This service includes outgoing call barring (OCB) and incoming call barring (ICB). OCB enables administrators to block IMS users from making certain types of outgoing calls, such as long distance or premium. ICB enables administrators to block specified incoming calls to individual users or group of users (such as group, department, and company).

5.4 NGN and IMS Solutions

NGN and IMS solutions are provided by several companies, such as Cisco, Ericsson, ACME and Italtel. The main components of IMS and NGN are session border control (SBC), SIP application server (AS), present server (PS), media softswitch, IMS core, subscriber databases, and media gateway. In this section, selected examples about NGN and IMS solutions will be described.

5.4.1 Session Border Control (SBC)

In NGN and IMS networks, SBC's are increasingly used for more advanced functionality, such as:

- *Core network protection and security.* This functionality includes access control, session policing, media policing and topology hiding and privacy. While access control is responsible for protecting the DoS attack from specific devices or from whole networks, session policing deals with mechanisms that drop a volume-based attack from trusted sessions in order to ensure that this attack will not overflow the SBC's normal call processing and subsequently not overflow systems beyond it (such as softswitch and IMS core). The media policing controls the RTP (and RTCP) rate and if this rate exceeds the pre-defined maximum rate, the incoming RTP traffic flows will be dropped. Finally, the topology and privacy function is used to hide the core topology and to prevent directed attacks [RFC-5853].

- *QoS marking.* QoS marking allows SBC to set the DSCP field for incoming media and signalling packets. Further network components use this DSCP field to handle these packets in overload situations (see chapter 3.10).
- *Call admission control and overload protection.* This function allows the control of signalling (such as SIP registrations) and media traffic based on different pre-defined policies [RFC-5853]. A new call is admitted if it meets its policy requirements. Fundamental call admission control mechanisms are discussed in 3.8).
- *Load balancing.* SBC also provides load balancing across the defined internal signalling endpoints (e.g. Softswitches, SIP application servers, SIP proxies, SIP gateways). Load balancing feature allow the setting of concurrent session capacity and rate attributes for each signalling endpoint.

According to the figure 5-4, a SBC can function as a P-CSCF at the user-network interface and an IBCF at the network-network interface. SBC platforms used in most telecommunication service providers are e.g. Cisco 7600, ACME Net-Net 4500, ACME Net-Net 4250.

5.4.2 Softswitch

A softswitch is a multiprotocol Media Gateway Control (MGC) typically has to support various signalling protocols (such as SIP [RFC3261], H.323, MGCP [RFC3435], SS7, and others), designed to provide internetworking in NGNs for IP-to-IP, IP-to-PSTN, and PSTN-to-PSTN connectivity by using of Session Initiation Protocol (SIP).

Softswitches are used in both NGN and IMS networks on the boundary point between packet networks and circuit switched networks. According to the figure 5-4, a softswitch can function as a Breakout Gateway Control Function (BGCF) server and a Media Gateway Controller Function (MGCF) server as well as a signalling gateway (SG). The key functions of a softswitch is to converting SIP signalling to ISUP/BICC signalling and to control the media gateway (MGW). The communication between two softswitch is performed via SIP or EISUP. Softswitch platforms used in a lot of telecommunication service providers are e.g. Cisco PGW 2200, Italtel softswitch (iSSW).

5.4.3 Media Gateway

A Media Gateway interfaces the media plane of the circuit switched (CS) network. On one side the MGW is able to send and receive NGN and IMS

media over the Real-Time Protocol (RTP). On the other side the MGW uses one or more PCM (Pulse Code Modulation) time slots to connect to the CS network. Additionally, the MGW performs transcoding when the NGN or IMS terminal does not support the codec used by the CS side. Each media gateway is controlled by a softswitch.

An example of media gateway platforms is the MGX 8880 from Cisco.

5.4.4 IMS Core

The IMS core has to take the role of the I-/S-/E-CSCF described in 5.3.2.1. IMS core platforms are provided for example by Italtel (www.italtel.com) and Ericsson (www.ericsson.com).

5.4.5 Subscriber Databases

Subscriber databases are central user databases keeping the relevant subscriber information of IMS users or of NGN users. Within the IMS network, the HSS functions as subscriber database. HSS platforms are offered e.g. by Ericsson (www.ericsson.com) and Italtel. Within the NGN network, the iUDB (Italtel Universal Database) platform is provided e.g. by Italtel (www.italtel.com).

5.4.6 Application Servers

Application servers discussed in 5.3.2.4 are provided by BroadSoft (www.broadsoft.com) and by Italtel.

5.5 Summary

This chapter started with an overview of NGN architecture covering service stratum and transport stratum. While the service stratum includes control functions and application functions, the transport stratum covers all functions that are responsible for forwarding and routing the IP packets. The NGN functions belong to both of these stratum. These functions are addressed in 5.2.2 as transport stratum functions, service stratum function, management function and user function. The IMS as the core of each NGN is illustrated in section 5.3. This section gives a survey of the IMS main functions (CSCF, HSS, SLF, application servers, IBCF, MRF, BGCF), their mechanisms (IMS addressing, P-CSCF discovery, IMS session control, S-CSCF assignment and AAA) and services (presence, messaging, Push to Talk over Cellular, Multimedia Telephony). NGN and IMS solutions on examples of their platforms are illustrated in section 5.4.

Reference

- [AA-2006] L. Andersson, E. Rosen. Framework for Layer 2 Virtual Private Networks (L2VPNs). RFC 4664, September 2006.
- [AAB-2000a] E. Altman, K. Avrachenkov, C. Barakat. A Stochastic Model of TCP/IP with Stationary Random Losses. Proceedings of ACM SIGCOMM, August 2000.
- [AAB-2000b] E. Altman, K. Avrachenkov, C. Barakat. TCP in presence of bursty losses. Performance Evaluation Nr. 42, pages 129-147, 2000.
- [ACE-2002] D. Awduche, A. Chiu, A. Elwalid, I. Widjaja and X. Xiao. Overview and Principles of Internet Traffic Engineering. RFC 3272, May 2002.
- [AF-2003] Ayyorgun S. and W. Feng. A probabilistic definition of burstiness characterization. Technical Report LA-UR 03-3668, Los Alamos National Laboratory, 2003.
- [AFM-1992] Armstrong, Freier and Cornell. „Multicast Transport Protocol“, RFC 1301, Feb. 1992.
- [AK-2005] K. Ahmad and R. Kapoor. The NGN Handbook. Cisco System, 2005.
- [AL-2005] F. Adrangi, H. Levkowitz. Problem Statement: Mobile IPv4 Traversal of Virtual Private Network (VPN) Gateways. RFC 4093, August 2005.
- [AM-2005] L. Andersson and T. Madsen. Provider Provisioned Virtual Private Network (VPN) Terminology. RFC 4026, March 2005.
- [ANS-2005] A. Adams, J. Nicholas and W. Siadak. Protocol Independent Multicast – Dense Mode (PIM-DM): Protocol Specification. RFC 3973, January 2005.
- [Arm-2000] G. Armitage. Quality of Service in IP Networks. Macmillan Technical Publishing. 2000.
- [APS-1999] M. Allman, V. Paxson, and W. Stevens, “TCP Congestion Control”, RFC 2581, IETF, April 1999
- [AS-2006] W. Augustyn, Y. Serbest. Service Requirements for Layer 2 Provider-Provisioned Virtual Private Networks. RFC 4665, September 2006.
- [APS-1999] M. Allman, V. Paxson and W. Stevens. TCP Congestion Control. RFC 2581, April 1999.

- [AWK-1999] G. Apostolopoulos, D. Williams, S. Kamat, R. Kurin, A. Orda and T. Przygienda. QoS Routing Mechanisms and OSPF Extensions. RFC 2676, August 1999.
- [BB-1995] A. Bakre and B. Badrinath. "I-TCP: Direct TCP for Mobile Hosts". Proceedings of 15th International Conference on Distributed Computing, Vancouver, Canada, 1995, pp. 136-143.
- [BBL-2000] Boorstyn R.; A. Burchard; J. Liebeherr; C. Oottamakorn. Statistical service assurances for traffic scheduling algorithms. IEEE Journal on Selected Areas in Communications. Special Issue on Internet QoS, 18(12): 2651-2664, 2000.
- [BBC-1998] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang and W. Weiss. An Architecture for Differentiated Services. RFC 2475, December 1998.
- [BCC-1998] B. Braden, D. Clark, J. Crowcroft. Recommendation of Queue Management and Congestion Avoidance in the Internet. RFC 2309, April 1998.
- [BJS-2000] L. Breslau; S. Jamin; A. Shenker. Comments on the Performance of Measurement-Based Admission Control Algorithms. IEEE Infocom 2000.
- [Bgp-2010] <http://bgp.potaroo.net/as6447/>
- [BH-2004] P. Broström, K. Holmberg. Multiobjective Design and Survivable IP Networks. Technical Report LiTH-MAT-R-2004-03, Division of Optimization, Linköping Institute of Technology, 2004.
- [BHK-2001] J. Backers, I. Hendrawan R. Kooij, and R. Van der Mei. Generalized Processor Sharing Performance Models for Internet Access Lines. Proceedings of the 9th IFIP Conference on Performance Modeling and Evaluation of ATM and IP Networks, Budapest, 2001.
- [BK-2000] A. Berger and Y. Kogan. Dimensioning Bandwidth for Elastic Traffic in High-Speed data Networks. IEEE/ACM Transactions on Networking, 2000.
- [BK-1999] A. Berger and Y. Kogan. Multi-Class Elastic Traffic: Bandwidth Engineering Via Asymptotic Approximations. Proceedings of 16th International Teletraffic Congress, 1999.
- [BGS-2001] L. Berger, D. Gan, G. Swallow, G. Pan, P. Tommasi. RSVP Refresh Overhead Reduction Extensions. RFC 2961, April 2001.

- [BKG-2001] J. Border, M. Kojo, J. Griner, G. Montenegro, Z. Shelby. Performance Enhancing Proxies Intended to Mitigate Link-Related Degradations. RFC 3135, June 2001.
- [BKS-2000] L. Breslau, E. Knightly, Scott Shenker. Endpoint Admission Control: Architectural Issues and Performance. Proceedings of ACM SIGCOMM 2000.
- [BLT-2000] F. Baker, B. Lindell, M. Talwar. RSVP Cryptographic Authentication. RFC 2747, January 2000.
- [BR-2002] P. Brockwell and D. Richard. Introduction to Time Series and Forecasting, 2nd ed, Springer-Verlag.
- [BRT-2004] L. Buriol, M. Resende and M. Thorup. Survivable IP Network Design with OSPF Routing. AT & T Labs Research Technical Report TD-64KUAW, 2004.
- [Bru-2004] M. Brunner. Requirements for Signaling Protocols. RFC 3726, April 2004.
- [Bol-1997] R. Bolla. "Bandwidth Allocation and Admission Control in ATM Networks with service separation". IEEE Communication Magazine, p. 130-137, 1997.
- [BRT-2004] L.S. Buriol, M.G Resende and M. Thorup. "Survivable IP Network Design with OSPF Routing", AT&T Labs Research Technical Report TD-64KUAW, 2004.
- [BPS-1996] H. Balakrishnan, V. Padmanabhan, S. Seshan. A Comparison of Mechanisms for Improving TCP Performance over Wireless Links. ACM Sigcomm 1996, Stanford, CA.
- [BT-2001] Boudec L. and P. Thiran. Network Calculus: A Theory of Deterministic Queuing Systems for the Internet. Springer-Verlag, 2001.
- [BZ-1993] R. Braudes and S. Zabele. „Requirements for Multicast Protocols“, RFC 1458, May 1993.
- [BZB-1997] R. Branden, L. Zhang, S. Berson, S. Herzog, S. Jamin. Resource Reservation Protocol (RSVP), RFC 2205. September 1997.
- [Cah-1998] R.S. Cahn. Wide Area Network Design – Concepts and Tool for Optimization. Morgan Kaufmann Publishers, Inc., San Francisco, CA, 1998.
- [CB-2001] B. Choi, R. Bettati. Endpoint Admission Control: Network Based Approach. Proceedings of 21th International Conference on Distributed Computing Systems, Phoenix, AZ, April 2001.

- [CB-2006] L. Coene and J. Balbas. Telephony Signalling over Stream Control Transmission protocol (SCTP) Applicability Statement. RFC 4166, February 2006.
- [CBL-2005] Ciucu F.; A. Burchard and J. Liebeherr. A Network service Curve Approach for Stochastic Analysis of Networks. Proceeding of Sigmetrics, 2005.
- [cis-2003-1] Internetworking Technology Handbook (4th Edition). Cisco System Inc. 2003.
- [Cisco-1] Classification Overview. Cisco Reference. http://www.cisco.com/en/US/docs/ios/12_0/qos/configuration/guide/qcclass.pdf
- [Cisco-2] Traffic Classification. In Cisco's "WAN and Application Solution Guide". http://www.cisco.com/en/US/docs/nsite/enterprise/wan/wan_optimization/chap05.pdf
- [Cisco-3] Traffic Policing. Cisco IOS Release 12.2. http://www.cisco.com/en/US/docs/ios/12_2t/12_2t2/feature/guide/ftpoli.pdf
- [Cisco-4] Configuring Generic Traffic Shaping. In "Cisco IOS Quality of Service Solution Configuration Guide". http://www.cisco.com/en/US/docs/ios/12_2/qos/configuration/guide/qcfgts.pdf
- [Cla-2004] B. Claise. Cisco Systems NetFlow Services Export Version 9. RFC 3954, October 2004.
- [RFC3630] D. Katz, K. Kompella and D. Yeung. Traffic Engineering (TE) Extensions to OSPF Version 2 . RFC 3630, September 2003.
- [Coo-1964] L. Cooper. Location-Allocation Problems. Operation Research, Vol. 11, 331-343, 1964.
- [Cro-1932] C.D. Crommelin. Delay probability formulae when the holding times are constant. Post Office Electrical Engineers Journal, Vol. 25(1932), pp. 41-50.
- [Cro-1934] C.D. Crommelin. Delay probability formulae. Post Office Electrical Engineers Journal, Vol. 26(1934), pp. 266-274.
- [CIA-2003] A. Caro, J. Iyengar, P. Amer, S. Ladha, G. Heinz and K. Shah. SCTP: A Proposed Standard for Robust Internet Data Transport. Computer Network, November 2003.
- [CM-2005] M. Carugi and D. McDysan. Service Requirements for Layer 3 Provider Provisioned Virtual Private Networks (PPVPNs). RFC 4031, April 2005.

- [CRA-1998] E. Crawley, R. Nair, Arrowpoint, B. Rajagopalan, H. Sandick. A Framework for QoS-based Routing in the Internet. RFC 2386, August 1998.
- [CQ-2001] H.J. Chao and X. Gui. *Quality of Service Control in High-Speed Networks*. John Wiley & Sons, 2001.
- [Dee-1989] S. Deering. Host Extensions for IP Multicasting. RFC 1112. August 1989.
- [DPB-2006] J. Davidson, J. Voice over IP Fundamentals. Macmillan Technical Publishing. August 2006.
- [Dvo-2001] An Assessment of IP-related Quality of Service work in ITU-T. Workshop on QoS and user-perceived transmission quality in evolving networks, Senegal, 18-19 October 2001.
- [EM-1993] A. Elwalid and D. Mitra. "Effective bandwidth of general Markovian traffic sources and admission control of high speed networks" , IEEE/ACM Trans. Netw., Vol. 1, No. 3, pp. 329-343, June 1993.
- [ERP-2002] M. Ericsson, M. Resende, P. Pardalos. A Genetic Algorithm for Weight Setting Problem in OSPF Routing. Journal of Combinatorial Optimization, 6:299-333, 2002.
- [EMH-2005] S. Eum, J. Murphy and J. Harris. A Failure Analysis of Tomography and EM Methods. TENCOM 2005.
- [E.360.1] ITU-T Recommendation E.360.1 (05/2002). Frameworks for QoS routing and related traffic engineering methods for IP-, ATM-, and TDM-based multiservice networks.
- [EQP-2006] A. Ezzouhari, A. Quintero and S. Pierre. A New SCTP mobility Schema Supporting Vertical Handover. IEEE International Conference on Wireless and Mobile Computing Networking and Communications, 19-21 June 2006, page 205-211.
- [EMW-1995] A. Elwalid, D. Miltra, H. Wendtworth. A new approach for allocating buffers and bandwidth to heterogeneous, regulated traffic in ATM node. IEEE journal on Selected Areas in Communications, Vol. 13, No. 6, August 1995, pp. 1115-1127.
- [ETSI-ES-187-003] Resource and Admission Control Sub-system (RACS) Functional Architecture.
- [ETSI-ES-187-004] NGN Functional Architecture; Network Attachment Sub System (NASS).
- [FGM-1999] R. Fielding, J. Gettys, J. Mogul, H. Frystyk. Hypertext Transfer Protocol -- HTTP/1.1. RFC 2616, June 1999.

- [FRT-2002] B. Fortz; J. Rexford; and M. Thorup. Traffic Engineering with Traditional IP Routing Protocols. IEEE Communications Magazine, 40(10):118-124, 2002.
- [Fen-1997] W. Fenner. Internet Group Management Protocol, Version 2. RFC 2236, November 1997.
- [FJ-1993] S. Floyd and V. Jacobson. Random Early Detection Gateways for Congestion Avoidance. IEEE/ACM Transactions on Networking, August 1993
- [FKP-2006] S. Floyd, K. Kohler and J. Padhye. Profile for Datagram Congestion Control Protocol (DCCP) Congestion Control ID 3: TCP-Friendly Rate Control (TFRC). RFC4342, March 2006.
- [FT-2000] B. Fortz and M. Thorup. Internet Traffic Engineering by Optimizing OSPF Weights. Proceedings of the IEEE Infocom, 2000.
- [FT-2002] B. Fortz and M. Thorup. Optimizing OSPF/IS-IS Weights in a Changing World. IEEE Journal on Selected Areas in Communications (JSAC), 20(4):756-766, May 2002.
- [FHH-2006] B. Fenner, M. Handley and H. Holbrook. Protocol Independent Multicast – Sparse Mode (PIM-SM). RFC 4601, August 2006.
- [Flo-1996] S. Floyd. Comments on measurement-based admission control for controlled-load services. Technical report, Lawrence Berkeley Laboratory, July 1996.
- [FK-2006] S. Floyd and E. Kohler. Profile for Datagram Congestion Control Protocol (DCCP) Congestion Control ID 2: TCP-like Congestion Control. RFC 4341, March 2006.
- [FS-2004] M. Fidler, V. Sander. A Parameter-based Admission Control for Differentiated Services Network. Computer Networks, Volume 44, Issue 4, March 2004.
- [FYT-1997] D. Funato, K. Yasuda and H. Tokuda. TCP-R: TCP Mobility Support for Continuous Operation. Proceeding IEEE ICNP'97, 1997, p. 229-236.
- [GC-2006] V. Grount and S. Cunningham. A Constrained Version of a Clustering Algorithm for Switch Placement and Interconnection in large Networks. Proceeding of 19th ISCA International Conference on Computer Applications in Industry and Engineering. Las Vegas, USA, 13-15 November 2006.

- [GJT-2004] A. Gunnar, M. Johansson and T. Telkamp. Traffic Matrix Estimation on a Large IP Backbone – A Comparison on Real data. IMC04, October 25-27, Sicily, Italy, 2004.
- [GH-1991] R.J. Gibbens and P. J Hunt. Effective Bandwidth for Multi-Type UAS Channel. *Queuing Systems*, Vol. 29, No. 10, October 1991, pp. 17-28.
- [GKP-2006] A. Ganesh, P. Key, D. Polis. Congestion Notification and Probing Mechanisms for Endpoint Admission Control. *IEEE/ACM Transactions on Networking*, Vol. 14, No. 3, June 2006.
- [GS-1999] J. Golestani and K. Sabnani. Fundamental Observation on Multicast Congestion Control in the Internet. *Proceedings of IEEE INFOCOM 1999*.
- [GV-1995] P. Goya and H. Vin. Generalized Guaranteed Rate Scheduling Algorithms: A Framework. Technical report TR-95-30, University of Texas, Austin, September 1995.
- [GL-1997] F. Glover and M. Laguna. Tabu Search. In “Modern Heuristic Techniques for Combinatorial Problems. 1997.
- [GSE-2000] R. Gibben, S. Sargood, C. Eijl, F. Kelly. Fixed-Point Models for the End-to-End Performance Analysis of IP Networks. In 13th ITC Special Seminar: IP Traffic Management, Modeling and Management, 2000.
- [GG-1992] R. Guerin and L. Gun. A Unified Approach to Bandwidth Allocation and Access Control in Fast Packet-Switched Networks. *Proceedings of IEEE INFOCOM’92*.
- [HB-1996] J. Hawkinson and T. Bates. Guidelines for Creation, Selection, and Registration of Autonomous System (AS). RFC 1930, March 1996.
- [HE-2006] E. Hyttia and P. Emstad. A Model for TCP congestion control capturing the correlations in times between the congestion events. *Proceedings of Next Generation Internet Design and Engineering*, 2006.
- [HGP-2000] O. Hersent, D. Gurle and J. Petit. IP Telephony – Packet-based Multimedia Communications Systems, Addison Wesley, 2000.
- [HL-2005] F. Hillier and G. Lieberman. Introduction to Operations Research. McGraw-Hill Higher Education, 8th Edition, 2005.
- [Hag-2006] S. Hagen. Ipv6 Essentials. O’Reilly Media. 2006.

- [HA-1997] Z. Haas and P. Agrawal. Mobile-TCP: an asymmetric transport protocol design for mobile system. IEEE Communications, Volume 2, p. 1054-1058, 1997.
- [Has-1989] E. Hashem. Analysis of Random Drop for Geteway Congestion Control. Technical Report LCS/TR-465, Laboratory for Comouter Science, Massachusetts Institute of Technology, Cambridge, MA, 1989
- [HFP-2003] H. Handley, S. Floyd and J. Padhye. TCP Friendly Rate Control (TFRC): Protocol Specification. RFC 3448, January 2003.
- [HFW-2000] M Handley, S. Floyd, B. Whetten, R. Kermode, L. Vicisane and M. Luby. The Reliable Multicast Design Space for Bulk Data Transfer. RFC 2877, August 2000.
- [HKL-2005] R. Hancock, G. Karagiannis, J. Loughney, S. Bosch. Next Step in Signaling (NSIS): Framework. RFC 4080, June 2005.
- [HD-2003] R. Hinden and S. Deering. Internet Protocol Version 6 (IPv6) Addressing Architecture. RFC 3513, April 2003.
- [Hoa-2003] Thi-Thanh-Mai Hoang. Label Switched Path Dimension and Bandwidth Assignment in MPLS Networks. In proceeding of SPECTS'03, p. 93-98, 2003.
- [Hoa-1998] Thi-Thanh-Mai Hoang. Planning of Wide Area Networks on the Example of ATM. Ph.D. dissertation in German. University of Karlsruhe, 1998.
- [Hoa-1999] Thi-Thanh-Mai Hoang. Planung von Weitverkehrsnetze am Beispiel ATM". Ph.D Dissertation. University of Karlsruhe, 1998.
- [Hoa-2004] Thi-Thanh-Mai Hoang. Network Management: Basic Notions and Frameworks, in "The Industrial Information Technology Handbook", Edited by R. Zurawski, CRC Press, ISBN: 0-8493-1985 -4, 2004.
- [Hoa-2005] Thi-Thanh-Mai Hoang. Survey of Network Management Frameworks, in "The Industrial Communication Technology Handbook", Edited by R. Zurawski, Taylor & Francis Books, ISBN: 0-8493-3077 -7, 2005.
- [Hoa-2007a] Thi-Thanh-Mai Hoang. Planning and Optimization of Multi-service Computer Networks. Proceedings of "10th Communications and Networking Simulation Symposium (CNS'07)", Norfolk, VA, USA, March 25-29, 2007, p. 9-16, ISBN 1-56555-312-8

- [Hoa-2007b] Thi-Thanh-Mai Hoang. Bandwidth Dimension and Capacity Planning of Unicast and multicast IP Networks. Proceedings of the international Symposium on Performance Evaluation of Computer and Telecommunication Systems, July 16-18, 2007, San Diego, California, p. 137-144, ISBN:1-56555-317-9
- [Hoa-2007c] Thi-Thanh-Mai Hoang. Optimization Algorithms for Multi-service IP Network Planning. Proceedings of "High Performance Computing Symposium (HPC2007), Norfolk, VA, USA, March 25-29, 2007, p. 412-418, ISBN:1-56555-313-6.
- [Hoa-2007d] Thi-Thanh-Mai Hoang. Improving QoS in Unicast and multicast IP-based Networks through Capacity Planning. Proceedings of the 11th IASTED International Conference on Internet and Multimedia Systems and Applications, Honolulu, Hawaii, USA, August 20-22, 2007, p. 68-73, ISBN:978-0-88986-678-2
- [HZ-2001] Thi-Thanh-Mai Hoang and W. Zorn. Genetic Algorithms for Capacity Planning of IP-based Networks. Proceedings of IEEE Congress on Evolutionary Computation 2001 (CEC2001), South Korea, ISBN 0-7803-6657-3, p. 1309-1315
- [HZ-2000] Thi-Thanh-Mai Hoang and W. Zorn. Planning of IP Backbones with Particular Consideration of Real-Time Traffic, in A. Tentner, ed., "High Performance Computing 2000", Washington D.C, USA, ISBN:1-56555-197 -4, 2000, p. 262-267.
- [HS-2003] G. Haßlinger and S. Schnitter. Optimized Traffic load distribution in MPLS Networks. In G. Anandalingam and S. Raghavan, editors, Telecommunication Network Design and Management. Kluwer Academic Publishers, Boiston, 2003.
- [HY-2001] K. Homberg and D. Yuan. Optimal Network Design and Routing for IP Traffic. In 3th International Workshop on Design of Reliable Communication Networks, Budapest, Hungary, October 2001.
- [Hus-2002] G. Huston. Internet Performance Survival Guide – QoS Strategies for Multiservice Networks. John Wiley & Sons, 2002.
- [Hed-1988] C. Hedrick. Routing Information Protocol. RFC 1058, June 1988.

- [Hui-1988] J. Y. Hui. “Resource Allocation for Broadband Networks “, IEEE J. Select. Areas Comm., Vol. 6, No. 9, pp. 1598-1608, Dec, 1988.
- [ITU-20029] ITU-T Recommendation E.360.2: QoS routing and related traffic engineering methods – call routing and connection routing methods. May 2002.
- [IK-2001] I. Ivars and G. Karlsson. “PBAC: Probe-based Admission Control”, Proceedings of QofIS, Springer, 2001.
- [JC-2006] K. Jaroenrat and P. Charnkeitkong. On Routing Efficiency of a Network Design Algorithm. Proceedings of the ACM Mobility 2006, October 25-27, 2006, Bangkok, Thailand.
- [Jai-1999] Raj Jain. Congestion Control in Computer Networks: Issues and Trends. IEEE Network Magazine, May, 1990, pp. 24-30.
- [JEN-2004] Y. Jiang, P. Emstad, V. Nicola, A. Nevin. Measurement-Based Admission Control: A Revisit“, 17th Nordic Teletraffic Seminar, 2004.
- [JSD-1997] S. Jamin, S. Shenker, P. danzig. Comparison of Measurement-based Admission Control Algorithms for Controlled-Load Service. Proc. Of INFOCOM’s 97, April 1997.
- [KHF-2006] E. Kohler, M. Handley, S. Floyd. Datagram Congestion Control Protocol (DCCP)RFC 4340
- [KHB-2007] A. Kotti, R. Hamza and K. Bouleimen. Bandwidth Constrained Routing Algorithm for MPLS Traffic Engineering. Proceeding of International Conference on Networking and Services (ICNS’ 2007), pages 20-
- [KB-2003] S. Köhler and A. Binzenhöfer. MPLS Traffic Engineering in OSPF Networks – A combined Approach. Technical Report 304, University of Würzburg, Februar 2003.
- [Kei-1996] J. Keilson. The ergodic queue length distribution for queuing systems with finite capacity. Journal of Royal Statistical Society, Series B, Vol. 28, 1990-201, 1966.
- [Ker-1993] A. Kershenbaum. Telecommunication Network Design Algorithms. McGraw-Hill, Inc., New York, NY, 1993.
- [Kes-1997] S. Kesahv. An Engineering Approach to Computer Networking. Addison Wesley, 1997.
- [KK-2000] A. Kherani and A. Kumar. Performance Analysis of TCP with Nonpersistent Session. Workshop on Modeling of Flow and Congestion Control, September 2000.

- [Kle-2011] Leonard Kleinrock. Queueing Systems: Computer Applications Vol 3. John Wiley & Sons Inc, 2nd Revised Edition, 2011.
- [Kli-1955] J.F.C Kingman. Mathematical methods in the theory of queueing. London 1960.
- [KW-1995] J. Kowalski and B. Waefield. Modeling Traffic Demand between Nodes in a Telecommunications Network. In ATMAC95.
- [Kat-1997] D. Katz. IP Router Alert Option. RFC 2113, February 1997.
- [Kes-2001] S. Keshav. An Engineering Approach to Computer Networking – ATM Networks, the Internet, and the Telephone Network. Addison-Wesley, 2001.
- [KKN-2006] G. Keeni, K. Koide, K. Nagami. Mobile Ipv6 Management Information Base. RFC 4295, April 2006.
- [Kle-1975a] L. Kleinrock. Queueing Systems, Volume 1: Theory. Wiley Interscience, New York, 1975.
- [Kle-1975b] L. Kleinrock. Queueing Systems, Volume 2: Computer Applications. Wiley Interscience, New York, 1975.
- [Koo-2007] R. Koodli. IP Address Location Privacy and Mobile IPv6: Problem Statement. RFC 4882, May 2007.
- [Koh-2005] E. Kohler. Datagram Congestion Control Protocol Mobility and Multihoming. Internet Draft, Januar 2005.
- [KO-2002] L. Krank and H. Orlamünder. Future Telecommunication Traffic – A Methodology for Estimation. In Proceedings of 10th International Telecommunication Network Strategy and Planning Symposium (NETWORKS 2002), pages 139-144, Munich, Germany, June 2002.
- [KPL-2006] M. Kulkarni, A. Patel and K. Leung. Mobile IPv4 Dynamic Home Agent (HA) Assignment. RFC 4433, March 2006.
- [KR-2007] K. Kompella, Y. Rekhter. Virtual Private LAN Service (VPLS) Using BGP for Auto-Discovery and Signalling. RFC 4761, January 2007.
- [KR-01] James F. Kurose, Keith W. Ross. Computer Networking: A top-down Approach Featuring the Internet, Addison Wesley, Reading, MA, 2001.
- [KSK-2002] S. Köhler; D. Staehle; U. Kohlhaas. “Optimization of IP Routing by Link Cost Specification”. In Internet Traffic Engineering and Traffic Management, 15th ITC Specialist Seminar, Wuerzburg, Germany, July 2002.

- [Kur-2004] James F. Kurose and K.W. Ross. Computer Networking – A Top-Down Approach Featuring the Internet. Addison-Wesley Longman, 2004.
- [LA-1998] Levine and Aceves. „A Comparison of reliable multicast protocols“, Multimedia System, Springer Verlag, 1998.
- [LK-2007] M. Lasserre and V. Kompelle. Virtual Private LAN Service (VPLS) Using Label Distribution Protocol (LDP) Signalling. RFC 4762, January 2007..
- [LFH-2006] D. Le, X. Fu and D. Hogrefe. A Review of Mobility Support Paradigms for the Internet. IEEE Communications Surveys & Tutorials, Vol. 8, page 38-51, 2006.
- [LM-1997] T. Lakshaman and U. Madhow. The performance of TCP/IP for networks with high bandwidth-delay products and random loss. IEEE/ACM Transactions on Networking, pages 336-350, June 1997.
- [LMS-1997] T. Lakshaman, U. Madhow and B. Suter. Window-based error recovery and flow control with a slow acknowledgement channel: a study of TCP/IP performance. Proceedings of INFOCOM, 1997.
- [LMS-2000] T. Lakshaman, U. Madhow and B. Suter. TCP/IP Performance with Random Loss and Bidirectional Congestion. IEEE/ACM Transactions on Networking, 8(5): 541-555, October 2000.
- [LPA-1998] Xue Li , Sanjoy Paul , Mostafa Ammar. Layered Video Multicast with Retransmissions (LVMR): Evaluation of Hierarchical Rate Control. INFOCOM'98, Infocom'1998, 29th March 1998 2th Apr 1998, San Francisco, CA, USA. 1998.
- [LQDB-2004] D. Lu, Y. Qiao, P. Dinda and F. Bustamante. Characterizing and Predicting TCP Throughput on the Wide Area Networks. Technical Report NWU-CS-04-34, Northwestern University, 2004.
- [Mal-1994] G. Malkin. RIP Version 2. RFC 1723, November 1994.
- [MF-2005] J. Manner and X. Fu. Analysis of Existing Quality-of-Service Signaling Protocols. RFC 4094, May 2005.
- [MJV-1996] S. McCanne, V. Jacobson, M. Vetterli. Receiver-driven layered multicast. SIGCOMM'96.
- [MG-2008] Miikka Poikselkä and Gerg Mayer. IP Multimedia Concepts and Services. Wiley 2008.
- [MHM-2003] M. Martin, E. Henrickson, D. Mills. Private Header (P-Header) Extensions ro the Session Initiation Protocol (SIP)

- for the 3rd-Generation Partnership Project (3GPP). RFC 3455. IETF, January 2003.
- [MH-2000] A. Mahmoodian and G. Haring. Mobile RSVP with Dynamic Resource Sharing. Wireless Communications and Networking Conference, 2000. IEEE Volume 2, pages 896-901, 2000.
- [MKM-2007] J. Manner, G. Karagiannis, A. McDonald. NSLP for Quality-of-Service Signaling. Internet Draft, Juni, 2007.
- [MMF-1996] M. Mathis, J. Mahdavi, S. Floyd, and A. Romanow, "TCP Selective Acknowledgement Options", RFC 2018, IETF, October 1996
- [ML-2003] Matrawy, Lambadaris. „A Survey of Congestion Control Schemes for Multicast Video Applications“, IEEE Communicationa Surveys & Tutorials, 2003.
- [ML-2003] A. Matrawy and I. Lambadardis. A Survey of Congestion Control Schemes for Multicast Video Applications. IEEE Communications, Vol. 5, No. 2, 2003.
- [MMJ-2007] J. Milbrandt, Michael Menth, and Jan Junker. Journal of Communications, Vol. 2, No. 1, January 2007.
- [McC-1998] J.D. McCabe. Practical Computer Network Analysis and Design. Morgan Kaufmann Publishers, Inc., San Francisco, CA, 1998.
- [MCD-2002] E. Miguez, J. Cidras, J. Dorado. An Improved Branch-Exchange Algorithm for Large-Scale Distribution Network Planning. IEEE Transaction on Power Systems, Vol. 17, Part 4, Pages 931-936, 2002.
- [MK-2002] E. Mulyana and U. Killat. An Alternative Generic Algorithm to Optimize OSPF Weights. In Internet Traffic Engineering and Traffic Management, 15-th ITC Specialist Seminar, Würzburg Germany, July 2002.
- [MGP-1989] S. Monteiro, J. Gerla and M. Pazoz. Topology Design and Bandwidth Allocation in ATM Networks, IEEE JSAC, 7:1253-1262, 1989.
- [Mit-1998] M. Mitchell. An Introduction to genetic Algorithm. 1998.
- [MS-2007] A. Mishra and A. Sahoo. S-OSPF: A Traffic Engineering Solution for OSPF based Best Effort Networks. Proceeding of Globecom 2007.
- [Min-1993] D. Minoli. *Broadband Network Analysis and Design*. Artech House, Boston-London, 1993.
- [Mur-1993] M. Murat. *On a Network Dimensioning Approach for the Internet*. IEICE Trans. Commun., Vol. E85-B, No. 1, 1993.

- [Mor-2007] T. Morin. Requirements for Multicast in Layer 3 Provider-Provisioned Virtual Private Networks. RFC 4834, April 2007.
- [MRE-2007] L. Martini, E. Rosen and N. El-Aawar. Transport of Layer 2 Frames over MPLS. RFC 4906, June 2007.
- [MSK-2006] J. Manner, T. Suihko, M. Kojo, M. Liljeberg, K. Raatikainen. Localized RSVP, Internet draft, Februar 2006.
- [Moy-1991] J. Moy. OSPF Version 2. RFC 1247, July 1991.
- [Moy-1994a] J. Moy . OSPF Version 2. RFC 1583, March 1994.
- [Moy-1994b] J. Moy. Multicast Extension to OSPF. RFC 1584, March 1994.
- [Moy-1997] J. Moy. OSPF Version 2. RFC 2178, July 1997.
- [Moy-1998] J. Moy. OSPF Version 2. RFC 2328, April 1998.
- [NCS-1999] A. Neogi, T. Chiueh and P. Stirpe. Performance Analysis of an RSVP-Capable router. IEEE Network, 13(5): 56-69, September 1999.
- [NBB-1998] K. Nichols, S. Blanke, F. Baker, D. Black. Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers. RFC 2474, December 1998.
- [OMA-2009] Push to Talk over Cellular 2.1 Requirements. Candidate Version 2.1. 22 Dec 2009. www.openmobilealliance.org.
- [OY-2002] L. Ong and J. Yoakum. An Introduction to the Stream Control Transmission Protocol. RFC 3286, May 2003.
- [PD-2003] L. Peterson and B.S. Davie. Computer Networks – A System Approach. Morgan Kaufman, 3rd Edition, 2003.
- [Per-2002] C. perkins. IP Mobility Support for IPv4. RFC 3220, January 2002.
- [Per-2006] C. perkins. Foreign Agent Error Extension for Mobile IPv4. RFC 4636, October 2006.
- [PFT-1998] J. Padhye, V. Firoiu, D. Towsley and J. Kurose. Modelling TCP Throughput: A Simple Model and its Empirical Validation. Proc. ACM SIGCOMM 1998.
- [PFTK-1998] J. Padhye, V. Firoiu, D. Towsley and J. Kurose. Modeling TCP Throughput: A Simple Model and its Empirical Validation. In ACM SIGCOMM'1998
- [PFTK-2000] J. Padhye, V. Firoiu, D. Towsley and J. Kurose. Modeling TCP Reno Performance: A Simple Model. IEEE/ACM Transaction on Networking, Vol. 8, No. 2, April 2000.

- [PF-2001] J. Padhye, and S. Floyd, “ On Inferring TCP Behavior”, Computer Communications Review ACM-SIGCOMM, Vol. 31, August 2001
- [PK-2000] D. Pham and D. Karaboga. Intelligent Optimisation Techniques: Genetic Algorithms, Tabu Search, Simulated Annealing and Neural Networks. Springer Verlag Berlin, 2000.
- [PK-2006] S.Floyd and E. Kohler. Profile for Datagram Congestion Control Protocol (DCCP) - Congestion Control ID 2: TCP-like Congestion Control. RFC 4341. March 2006.
- [PG-1993] A.K. Parekh and R. G. Galager. A generalized processor sharing approach to flow control in integrated service networks: the single node case. IEEE/ACM Trans. Netw., Vol. 1, no. 3, pp. 344-357, Jun. 1993.
- [PG-1994] A.K. Parekh and R. G. Galager. A generalized processor sharing approach to flow control in integrated service networks: the multiple node case. IEEE/ACM Trans. Netw., Vol. 2, no. 2, pp. 137-150, Apr. 1994.
- [PG-2006] A. Patel and G. Giarrette. Problem Statement for Bootstrapping Mobile IPv6 (MIP6). RFC 4640, September 2006.
- [PM-2008] Miikka Poikselkä and Georg Mayer. IMS IP Multimedia Concepts and Services. Wiley, 2008.
- [PTK-1993] S. Pingali, D. Towsley and J. Kurose. „A Comparison of Sender Initiated and Receiver-Initiated Reliable Multicast Protocols“, Proceedings of the Sigmetrics Conference on Messurement and Modelling of Computer Systems. 1993.
- [PW-2000] K. Park and W. Willinger. Self-Similar Network Traffic and Performance Evaluation. John Wiley & Sons, 2000.
- [QBM-1999] N. Queija, H. Berg and M. Mandjes. Performance Evaluation of Strategies for Intergration of Elastic and Stream Traffic. Technical Report PNA-R9903, Centra for Mathematics and Computer Science, 1999.
- [REV-2001] M. Roughan, A. Erramilli and D. Veitch. Network Performance for TCP Networks: Part I: Persistent Sources. In Proceedings of the Internetonial Teletraffic Congress ITC-17, pages 24-28, September 2001.
- [RGK-2002] M. Roughan, A. Greeberg, C. Kalmanek, M. Rumsewicz, J. Yates and Y. Zhang. Experience in Measuring Backbone Traffic Variability: Models, Metrics, Measurements and

- Monitoring. ACM SIGCOMM Internet Measurement Workshop, 2002.
- [RFB-2001] K. Ramakrishnan, S. Floyd, D. Black. The Addition of Explicit Congestion Notification (ECN) to IP. RFC 3168, September 2001.
- [RFC1066] K. McCloghrie, M. Rose. Management Information Base for Network Management of TCP/IP-based internets. RFC 1066. August 1988.
- [RFC1155] M. Rose and K. McCloghrie. Structure and Identification of Management Information for TCP/IP-based Internets. RFC 1155, May 1990.
- [RFC1157] J. Case, M. Fedor, M. Schoffstall, J. Davin. A Simple Network Management Protocol (SNMP). RFC 1157. May 1990.
- [RFC1212] M. Rose, K. McCloghrie. Concise MIB Definitions. RFC 1212. March 1991.
- [RFC1301] S. Armstrong et. Al. „Multicast Transport Protocol“, RFC 1301, 1992
- [RFC1633] R. Braden, D. Clark, S. Shenker. Integrated Services in the Internet Architecture: an Overview. RFC 1633. June 1994.
- [RFC2205] R. Braden, L. Zhang, S. Berson, S. Herzog, S. Jamin. Resource ReSerVation Protocol (RSVP) Version 1 Functional Specification. RFC 2205. September 1997.
- [RFC2357] A. Mankin. „IETF Criteria for Evaluating Reliable Multicast Transport and Application“, RFC 2357, June 1998
- [RFC2474] RFC 2474—Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers
- [RFC2475] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, W. Weiss. An Architecture for Differentiated Services. RFC 2475, December 1998.
- [RFC2597] J. Heinanen, F. Baker, W. Weiss, J. Wroclawski. Assured Forwarding PHB Group. RFC 2597, June 1999.
- [RFC2661] W. Townsley, A. Valencia, A. Rubens, G. Pall, G. Zorn, B. Palter. Layer Two Tunneling Protocol "L2TP". RFC 2661, August 1999.
- [RFC2702] D. Awduche, J. Malcolm, J. Agogbua, M. O'Dell, J. McManus. Requirements for Traffic Engineering Over MPLS. RFC 2702, September 1999.
- [RFC2784] D. Farinacci, T. Li, S. Hanks, D. Meyer, P. Traina. Generic Routing Encapsulation (GRE). RFC 2784, March 2000.

- [RFC2810] C. Kalt. Internet Relay Chat: Architecture. RFC 2810, April 2000.
- [RFC2887] S. Handley et. Al. „The Reliable Multicast Design Space for Bulk Data Transfer“. RFC 2887, 2000
- [RFC2890] G. Dommety. Key and Sequence Number Extensions to GRE. RFC 2890, September 2000.
- [RFC3031] E. Rosen, A. Viswanathan, R. Callon. Multiprotocol Label Switching Architecture. RFC 3031, January 2001.
- [RFC3032] E. Rosen, D. Tappan, G. Fedorkow, Y. Rekhter, D. Farinacci, T. Li, A. Conta. MPLS Label Stack Encoding. RFC 3032, January 2001
- [RFC3036] L. Andersson, P. Doolan, N. Feldman, A. Fredette, B. Thomas. LDP Specification. RFC 3036, January 2001.
- [RFC3140] D. Black, S. Brim, B. Carpenter, F. Le Faucheur. Per Hop Behavior Identification Codes (Obsoletes RFC 2836). RFC 3140, June 2001.
- [RFC3209] D. Awduche, L. Berger, D. Gan, T. Li, V. Srinivasan, G. Swallow. RSVP-TE: Extensions to RSVP for LSP Tunnels. RFC 3209. December 2001.
- [RFC3246] B. Davie, A. Charny, J. Bennett, K. Benson, Le Bodec, W. Courtney. An Expedited Forwarding PHB, RFC 3246 (Obsoletes RFC 2598), March 2002.
- [RFC3260] D. Grossman. New Terminology and Clarifications for Diffserv, RFC 3260, April 2002.
- [RFC3428] B. Campbell, J. Rosenberg, H. Schulzrinne, C. Huitema, D. Gurle. Session Initiation Protocol (SIP) Extension for Instant Messaging. RFC 3428, December 2002.
- [RFC3448] M. Handley, S. Floyd, J. Padhye, J. Widmer. TCP Friendly Rate Control (TFRC): Protocol Specification. RFC 3448, January 2003.
- [RFC3931] J. Lau, M. Townsley, I. Goyret. Layer Two Tunneling Protocol - Version 3 (L2TPv3). RFC 3931, March 2005.
- [RFC3985] S. Bryant, P. Pate. Pseudo Wire Emulation Edge-to-Edge (PWE3) Architecture. RFC 3985, March 2005.
- [RFC4080] R. Hancock, G. Karagiannis, J. Loughney, S. Van den Bosch. Next Steps in Signaling (NSIS): Framework. RFC 4080, June 2005.
- [RFC4301] S. Kent, K. Seo. Security Architecture for the Internet Protocol. RFC 4301, December 2005.

- [RFC4309] R. Housley. Using Advanced Encryption Standard (AES) CCM Mode with IPsec Encapsulating Security Payload (ESP). RFC 4309, December 2005.
- [RFC4364] E. Rosen, Y. Rekhter. BGP/MPLS IP Virtual Private Networks (VPNs). RFC 4364, February 2006.
- [RFC4448] L. Martini, E. Rosen, N. El-Aawar, G. Heron. Encapsulation Methods for Transport of Ethernet over MPLS Networks. RFC 4448, April 2006.
- [RFC4594] J. Babiarez, K. Chan, F. Baker. Configuration Guidelines for DiffServ Service Classes. RFC 4594, August 2006.
- [RFC5321] J. Klensin. Simple Mail Transfer Protocol. RFC 5321, October 2008.
- RFC-5853] J. Hautakorpi, G. Camarillo, R. Penfield, A. Hawrylyshen, M. Bhatie. Requirements from Session Initiation Protocol (SIP) Session Border Control (SBC) Deployments. RFC 5853. April 2010.
- [RJ-1988] K.K. Ramakrishnan and Raj Jain. A binary feedback scheme for congestion avoidance in computer networks with a connectionless network layer. In SIGCOMM Symposium on Communications Architectures and Protocols, pp. 303-313, Standfort, California, August 1988.
- [RJ-1990] K.K. Ramakrishnan and Raj Jain. A binary feedback scheme for congestion avoidance in computer networks. ACM Transactions on Computer Systems, 8(2): 158-181, May 1990.
- [RLH-2006] Y. Rekhter, T. Li and S. Hares. A Border Gateway Protocol 4 (BGP-4). RFC 4271, January 2006.
- [RMV-1996] Robert J.; U. Mocci; J. Virtamo. Broadband Network Teletraffic. Final Report of Action COST 242, Nr. 1155, Lecture Note in Computer Science, Spring Verlag, 1996.
- [Ros-2007] C. Rosen. Multicast in MPLS/BGP IP VPNs. Internet Draft, July 2007, Expiration Date Hanuary 2008.
- [RR-2006] E. Rosen and Y. Rekhter. BGP/MPLS IP Virtual Private Networks (VPNs). RFC 4364, February 2006.
- [Rob-1992] W. Robert. Performance Evaluation and Design of Multiservice Networks. COST 224 Final Report, 1992.
- [Rob-1996] J. Robert, U. Mocci and J. Virtamo. Broadband Network Teletraffic. Final Report of Action COST 242, Nr. 1155, Lecture Note in Computer Science, Spring Verlag, 1996.
- [RS-1994] Richard Stevens. TCP/IP Illustrated, Volume 1. Addison-Wesley Professional Computing Series. 1994

- [RSC-2002] J. Rosenberg, H. Schulzrinne, G. Camarillo et. Al. SIP: Session Initiation Protocol, RFC 3261, June 2002.
- [RT-2007] M. Riegel and M. Tuexen. Mobile SCTP. Internet Draft, November 2007.
- [San-2002] Sanjay Jha and Mahbub Hassan. *Engineering Internet QoS*. Artech House, Inc., 2002.
- [Sch-2003] H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson. RTP: A Transport Protocol for Real-Time Applications. RFC 3550, July 2003.
- [SH-2007] H. Schulzrine and R. Hancock. GIST: General Internet Signaling. Internet Draft, April 2007.
- [SK0-2002] K. Stordahl, K. Kalhagen, B.T. Olsen. Traffic forecast models for transport network. In Proceeding of 10th International Telecommunication Network Strategy and Planning Symposium, pages 145-150, Munich, Germany, June 2002.
- [Spo-2002] M. Sportack. IP Routing Fundamental. Cisco Press, 2002.
- [STA-2006] M. Stiernerling, M Tschofenig, C. Aoun and E. Davies. NAT/Firewall NSIS Signaling Layer Protocol (BSLP). Internet Draft, October 2006.
- [SWE-2003] N. Spring, D. Wetherall, D. Ely. Robust Explicit Congestion Notification (ECN) Signaling with Nonces. RFC 3540, June 2003.
- [SXM-2000] R. Stewart, Q. Xie, K. Morneault. Streaming Control Transmission Protocol. RFC 2960, October 2000.
- [Sna-2005] J. C. Snader. VPNs Illustrated. Tunnels, VPNs, and IPSec. Addison-Wesley Longman, 2005.
- [San-2006] A. Santiago. QoS for IP/MPLS Networks. Macmillan Technical Publishing, 2006.
- [Sch-1977] M. Schwartz. Computer Communication Network Design and Analysis. Englewood Cliffs, Prentice-Hall, 1977. ISBN 0-13-165134-x.
- [Sha-1990] R. Shama. Network Topology Optimization. Van Nostrand Reinhold, New York, 1990, ISBN 0-442-23819-3.
- [SH-2008] H. Schulzrinne and R. Hancock. GISZ: General Internet Signalling Transport. Internet Draft, February 3, 2008.
- [SZ-2005] Sanaa Sharafeddine, Zaher Dawy. Capacity Assignment for Video Traffic in Multiservice IP Networks with Statistical QoS Guarantees. Proceedings of 10th IEEE Symposium on

- Computers and Communications (ISCC 2005), 27-30 June 2005, Murcia, Cartagena, Spain, pages 243-248, 2005
- [SKZ-2004] Sanaa Sharafeddine, N. Kongtong, Zaher Dawy. Capacity Allocation for Voice over IP Networks Using Maximum Waiting Time Models. Proceedings of 11th International Conference on Telecommunications (ICT 2004), Fortaleza, Brazil, August 1-6, 2004, pages 660-670, 2004.
- [SH-2002] S. Schnitter and G. Haßlinger. Heuristic Solutions to the LSP-Design for MPLS. Proceedings of 10th International Telecommunication Network Strategy and Planning Symposium (NETWORKS 2002), pages 269-273, Munich, Germany, 2002.
- [SH-2003] Schnitter and G. Haßlinger. Optimized Traffic Load Distribution in MPLS Networks. In G. Anandalingam and S. Raghavan, edi., Telecommunication Network Design and Management. Kluwer Academic Publishers, Boston, 2003.
- [SH-2002] Sanjha and Hassan M. Engineering *Internet QoS*, Artech House, Inc., 2002.
- [SJ-2006] H. Sinnreich and A. Johnston. Internet Communications Using SIP: Delivering VoIP and Multimedia Services with Session Initiation Protocol. Wiley & Son, 2006.
- [SH-2002] S. Schnitter and G. Haßlinger. 2002. "Heuristic Solution to the LSP-Design for MPLS". In Proceedings of 10th International Telecommunication Network Strategy and Planning Symposium (NETWORK2002), p. 269-273, Munich, Germany, 2002.
- [Schu-1997] H. Schulzrine. Re-engineering the telephone systems. In Proc. Of IEEE Singapore International Conference on Networks (SICON), Singapore, Apr. 1997.
- [Sch-1997] M. Schwarz . Computer Communication Network Design and Analysis. Englewood Cliffs. N.J. Prentice-Hall, 1997.
- [SFK-2004] H. Sugano, S. Fujimoto, G. Klyne. Presence Information Data Format. RFC 3863. August 2004.
- [Tan-2002] A. Tanenbaum. *Computer Networks*. Prentice Hall, 4rd Edition, 2002.
- [Tan-1978] D. Tang. Optimization of Teleprocessing Networks with Concentrators and Multi-connected Terminals. IEEE Transactions on Computers, Vol. 27, No. 7, pp. 594-604, 1978
- [TBA-2001] A. Talukdar, B. Badrinath, and A Acharya. "MRSVP: A Resource Reservation Protocol for an Integrated Services

- Network with Mobile Hosts”, Wireless Networks, Vol. 7, No. 1, pp. 5-19, 2001.
- [TG-2005] H. Tschofenig and R. Gravemen. RSVP Security Properties. RFC 4230, December 2005.
- [TG-1997] D. Tse and M. Grossglauber. Measurement-based Call Admission Control: Analysis and Simulation”, Proc. Of INFOCOM’97, April 1997.
- [TS23.141] Presence Service: Architecture and Functional Description, 3GPP TS 23.141 V9.0.0. Jan.2010
- [TS24.229] IP multimedia call control protocol based on session initiation protocol (SIP) and session description protocol (SDP). 3GPP TS 24.229
- [TR-180.000] Telecommunications and Internet Converged Services and Protocols for Advantaged Networking (TISPAN); NGN Terminology
- [TS181.001] Telecommunication and Internet converged Services and Protocols for Advanced Networking (TISPAN); NGN Release 1. ETSI TR 181 001, Technical report, 03/2006
- [TS181.002] Telecommunication and Internet converged Services and Protocols for Advanced Networking (TISPAN); Multimedia Telephony with PSTN/ISDN Simulation Services, ETSI TR 181 002, Technical Specification, 03/2006.
- [TS181.005] Telecommunication and Internet converged Services and Protocols for Advanced Networking (TISPAN); Services and Capabilities Requirements, ETSI TR 181 005, Technical Specification, 03/2006.
- [TS181.018] Telecommunication and Internet converged Services and Protocols for Advanced Networking (TISPAN); Requirements for QoS in a NGN, ETSI TR 181 018, Technical Specification, 08/2007.
- [TS188.001] Telecommunication and Internet converged Services and Protocols for Advanced Networking (TISPAN); NGN management, ETSI TR 181 018, Technical Specification, 09/2005.
- [TXA-2005] Y. Tian, Kai Xu and N. Ansari. TCP in Wireless Environments: Problems and Solutions. IEEE Communication Magazine, Vol. 43, Issue 3, March 2005, p. 27-32.
- [TZ-2007] H. Tran and T. Ziegler. A design framework towards the profitable operation of service overlay networks. Computer Networks Vol. 51, 2007, pages 94-113.

- [WI-2005] H. Wang and M. Ito. Dynamic of Load Sensitive Adaptive Routing. Proceedings of IEEE International Conference on Communications (ICC), 2005.
- [Y.2001] ITU-T. Next Generation Networks – Frameworks and functional architecture models. ITU-T Recommendation Y.2001, 10/2004.
- [Y.2011] ITU-T. General principles and general reference model for Next Generation Networks. ITU-T Recommendation Y.2011, 10/2004
- [YB-1994] R. Yavatka and N. Bhagawat. Improving End-to-End Performance of TCP over Mobile Internetworks. IEEE workshop on Mobile Computing Systems and Applications, Santa Cruz, CA, 1994
- [WH-2006] J. Widmer and M. Handley. TCP-Friendly Multicast Congestion Control (TFMCC): Protocol Specification. RFC 4654, August 2006.
- [WPD-1988] Distance Vector Multicast Routing Protocol. RFC 1075, November 1988.
- [YYP-2001] Identity Representation for RSVP. RFC 3182, October 2001.
- [XHB-2000] X. Xiao, A. Hannan, B. Bailey. Traffic engineering with MPLS in the Internet. IEEE Network magazine, pages 28-33, March/April 2000.
- [ZA-2005] B. Zheng and M. Atiquzzaman. System Design and Network Requirements for Interactive Multimedia. IEEE Transactions on Circuits and Systems for Video Technology, Vol. 15, No. 1, 2005.
- [ZRD-2003] Y. Zhang, M. Roughan, N. Duffield and A. Greenberg. Fast Accurate Computation of Large-Scale IP Traffic Matrices from Link Loads. SIGMETRICS'03, June 10-14, 2003, San Diego, USA.



Leslie Haddon (ed.)

The Contemporary Internet

National and Cross-National European Studies

Frankfurt am Main, Berlin, Bern, Bruxelles, New York, Oxford, Wien, 2011.
210 pp., num. tab. and graph.

Participation in Broadband Society.

Edited by Leopoldina Fortunati, Julian Gebhardt and Jane Vincent. Vol. 3
ISBN 978-3-631-60098-6 · hb. € 39,80*

The contemporary internet focuses on user experiences of more recent developments on the internet, specifically with the spread of broadband, the audio-visual applications it has enabled, Web2.0 uptake more generally and the growth of eGovernment. *The Contemporary Internet* is comparative in two senses. The first is at the cross-national level, examining factors affecting different countries' experiences of the internet, with a particular, but not a sole, interest in what may be termed 'cultural influences on perceptions, adoption and use'. Second, the book is comparative within countries, examining the, sometimes very, uneven experiences of the internet's possibilities. One question that pervades several chapters is how the digital divide is evolving in the light of the more contemporary developments outlined above.

Content: Internet · Broadband · Digital Divide · Web2.0 · Cross-National Comparisons · Cultural Influences

*The e-price includes German tax rate. Prices are subject to change without notice

Frankfurt am Main · Berlin · Bern · Bruxelles · New York · Oxford · Wien
Distribution: Verlag Peter Lang AG
Moosstr. 1, CH-2542 Pieterlen
Telefax 0041 (0) 32/376 17 27
E-Mail info@peterlang.com

40 Years of Academic Publishing
Homepage <http://www.peterlang.com>