



Computer Networks

Chapter I. Computer Networks and the Internet

The Network Edge

Constitution of the Internet

The Internet is a computer network that interconnects billions of computing devices throughout the world. In Internet jargon, the devices that exploits Internet connection are called **end systems** or **hosts**.

End systems are connected together by a network of communication links and packet switches. Different links can transmit data at different rates, with the transmission rate of a link measured in bits/seconds. When one system wishes to send some data to another system, it splits the data and put resulting segments into the packages which are called **packets**.

A **packet switch** takes a packet arriving on one of its incoming communication links and forwards that packet on one of its outgoing communication links.

End systems access the Internet through **Internet Service Providers (ISPs)**. Each ISP is in itself a network of packet switches and communication links. An upper-tier ISP consists of high-speed routers interconnected with high-speed fiber-optic links.

End systems, packet switches, and other pieces of the Internet run **protocols** that control sending and receiving of information within the Internet. The **Transmission Control Protocol (TCP)** and **Internet Protocol (IP)** are two of the most important protocols in the Internet.

From another side of view, Internet can be defined as an infrastructure that provides services to applications. End systems attached to the Internet provide a **socket interface** that specifies how a program running on one end system asks the Internet

infrastructure to deliver data to a specific destination program running on another end system. **Socket Interface** is a set of rules that the sending program must follow.

A **protocol** defines the format and the order of messages exchanged between two or more communicating entities, as well as the actions taken on the transmission and/or receipt of a message or other event.





End systems are also referred to as *hosts* because they host application programs. Hosts are sometimes further divided into two categories: clients and servers.

Access networks — the network that physically connects an end system to the first router on a path from the end system to any other distant end system.

Today, the most prevalent types of broadband residential access are **digital subscriber line (DSL)** and **cable**.

Physical Media

Physical media fall into two categories: guided media and unguided media

 Media	 Brief	 Rates	 Prices
<u>Twisted-Pair Copper Wire</u>	The least expensive and most commonly used guided transmission medium. It consists of two insulated copper wires, each about 1 mm.	10Mbps — 10 Gbps (6a cable)	Low, the most spreaded
<u>Coaxial Cable</u>	It consists of two copper conductors, but they are concentric rather than parallel.	Hundreds of Mbps	Still quite a common
<u>Fiber Optics</u>	Thin, flexible medium that conducts pulses of light.	from 51.8 Mbps up to 39.8 Gbps.	Moderately higher
<u>Terrestrial Radio Channels</u>	Radio channels carry signals in the electromagnetic spectrum. Highly dependent on the propagation environment.	Unspecified	Unspecified

Aa Media	Brief	Rates	Prices
<u>Satellite Radio Channels</u>	Links two or more Earth-based microwave transmitter/receivers known as ground stations. The satellite receives transmissions on one frequency band, regenerates the signal using a repeated and transmits the signal on another frequency. Subdivided into geostationary satellites and low-earth orbiting (LEO) satellites	Unspecified	Unspecified

The Network Core

In a network application, end systems exchange messages with each other. Messages can contain anything the application designer wants. Between source and destination, each packet travels through communication links and packet switches.

Store-and-Forward Transmission

Store-and-Forward Transmission means that the packet switch must receive the entire packet before it can begin to transmit the first bit of the packet onto the unbound link.

End-to-end delay formula is:

$$delay = N \frac{L}{R}$$

N is a number of links, R is a transmission rate in bit/s and L is a size of a packet in bits.

Queuing Delays and Packet Loss

Each packet switch has multiple links attached to it. For each attached link, the packet switch has an **output buffer**, which stores packets that the router is about to send into that link. Thus, in addition to the store-and-forward delays, packets suffer output buffer **queuing delays**. Since the amount of the buffer space is finite, an arriving packet may find that the buffer is completely full causing the **packet loss**.

Forwarding Tables and Routing Protocols

In the Internet, every end system has an address called an IP address. When the end system wants to send a message to another one, it includes the destination's IP address in the packet's header. When a packet arriving at a router in the network, the router examines the relevant portion of a packet's destination address and forwards the

packet to an adjacent router. Specifically, each router has a **forwarding table** that matches the destination addresses with router's outbound links.

The Internet has a number of a special routing protocols that are used to automatically set the forwarding tables.

Circuit Switching

In a circuit-switching based networks a connection between two end systems is reserved and packets are transmitted at a constant rate without any congestion. A circuit in a link is implemented with either frequency-division multiplexing (FDM) or time-division multiplexing (TDM).

Connection Characteristics

When travelling from source end node to a destination, a packet might experience delays. The most important of these delays are the **nodal processing delay**, **queuing delay**, **transmission delay**, and **propagation delay**.

Processing Delay

The time required to examine the packet's header and determine where to direct the packet is part of the processing delay. It also can include other factors, such as the time needed for checking bit-level errors in the packet.

Queuing Delay

At the queue, the packet experience a queuing delay as it waits to be transmitted onto the link.

Transmission Delay

Transmission delay is the time taken to push the entire packet into the link. Assuming FIFO algorithm, a packet can be transmitted only after all the packets that have arrived before it have been transmitted. The transmission delay is:

$$\frac{L}{R}$$

where L is a bits per packet (packet length) and R is the transmission rate.

Propagation Delay

The time required to propagate from the beginning of the link to its end is the propagation delay. The propagation delay is the distance between two routers divided by the propagation speed.

Propagation vs Transmission Delay

The propagation delay is mainly caused by physical medium, while the transmission delay depends only on the packets' size and transmission rate.

Queuing Delay and Packet Loss

The ratio $\frac{La}{R}$ is called **traffic intensity** (a — the average rate at which packets arrive at the queue). If this ratio is above one, the queue will grow infinitely, which will lead to network throttle.

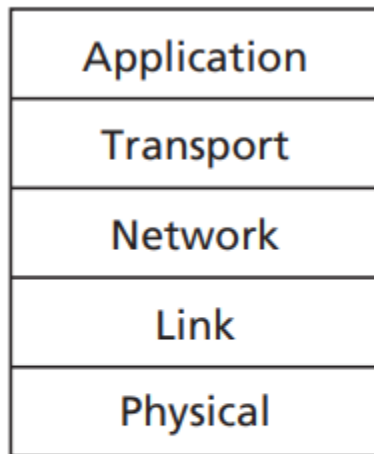
Throughput

The **instantaneous throughput** at any instant is the rate (bits/sec) at which one host is receiving the data. If the data consists of F bits and takes T seconds to reach the destination, then the **average throughput** is determined as $\frac{F}{T}$ bits/sec.

If the route consists of the nodes with transmission rates $\{R_1, R_2, \dots, R_n\}$, then the throughput of such connection is calculated as $\min\{R_1, R_2, \dots, R_n\}$.

Protocol Layers and Their Service Models

To provide structure to the design of network protocols, network designers organize protocols in layers. The services that a certain layer offers are referred as **service model of a layer**.



**Five-layer
Internet
protocol stack**

Application Layer

The application layer is where network applications and their application-layer protocols reside. A packet of information at the application layer is referred as a **message**.

Transport Layer

The Internet's transport layer transports application-layer messages between application endpoints. A packet of information at the transport layer is referred as a segment.

Network Layer

The Internet's network layer is responsible for moving network-layer packets known as **datagrams** from one host to another.

Link Layer

To move packet from one node (host or router) to the next node in the route, the network layer relies on the services of the link layer. The packet of information at this layer is referred as a **frame**.

Physical Layer

Physical layer is responsible for moving individual bits within the frame from one node to the next.

History of the Networks

Stay tuned, the notes will appear soon.

Chapter II. Application Layer

At the core of network application development is writing programs that run on different end systems and communicate with each other over the network.

There are two **predominant** architectures for network applications: client-server and peer-to-peer (P2P). One of the most cool feature of the P2P architecture is a scalability.

Process Communication

In terms of operating systems, they are not the programs, but processes who communicate with each other. Processes on two different end systems communicate with each other by exchanging messages across the computer network.

In the context of communication session between a pair of processes, the process that initiates the communication is labeled as the *client*. The process that waits to be contacted to begin the session is the server.

A process sends message into, and receives messages from, the network through a software interface called a **socket**. A socket is the interface between the application layer and the transport layer within a host.

Addressing Processes

In the Internet, the host is identified by its **IP address**. A destination **port number** helps identifying the particular network application running on the machine addressed by certain IP.

Transport Services Available to Applications

The transport layer protocols might provide the following services:

- **Reliable Data Transfer.** Protocol provides a guaranteed data delivery services (once packet is passed to the transportation layer, it is certainly delivered to destination).
- **Throughput.** Protocol provides transportation of packets at a certain constant transmission rate (throughput in bits/sec).
- **Timing.** ****Protocol provides guarantees related to timing (an upper-bound for latency and etc.)
- **Security.** Protocol provides security services (encoding, authentication and etc.)

Transport Services provided in the Internet

There are two transport-layer protocols provided in the Internet:

- **Transmission Control Protocol (TCP)** provides connection-oriented service (handshaking) and reliable data transfer along with congestion control. With **Transport Layer Security** also provides security services.
- **User Datagram Protocol (UDP)** provides no special services.

Application Layer Protocols

An **application-layer protocol** defines how an application's processes, running on different end systems, pass messages to each other. In particular, an application-layer protocol defines:

- The types of messages exchanged
- The syntax of various message types
- The semantics of the fields
- Rules for determining how and when a process sends messages and responds to them

HTTP

The HTTP protocol is **the heart** of the Web. The HTTP is implemented in two programs: a client program and a server program. HTTP uses TCP as its underlying transport protocol. The HTTP clients first initiates a TCP connection with the server. Once the connection is established, the browser and the server processes access TCP through

their socket interfaces. Since HTTP maintains no information about the clients, HTTP is said to be a **stateless protocol**.

Persistent and non-persistent connections

When the client-server interaction is taking place over TCP, the application developer needs to make an important decision — should each request/response pair be sent over a *separate* TCP connection, or should all of the requests and their corresponding responses be sent over the *same* TCP connection. The first approach the non-persistent connection is said to be used, while in the second the approach is referred as persistent connection.

The default mode of HTTP uses persistent connections with pipe-lining.

There are two types of HTTP messages: request and response messages. Each of the message is written in human-readable ASCII text. Each message consists of at least two parts: **request (status) line** and **header line**. The request line has three fields: the method field, the URL field and the HTTP version field in case of request or the HTTP version field, the status code and the status message in case of response message. The header line contains information about the connection type, timestamps and other metadata. Some requests and responses might contain some entity which is stored after header in an **entity body**.

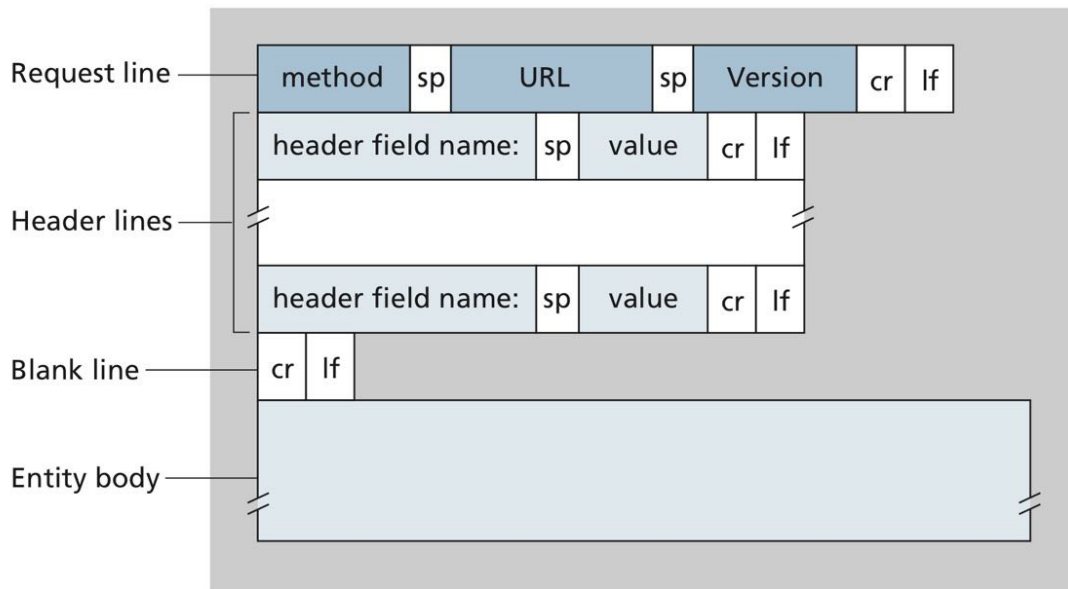


Figure 2.8 ♦ General format of an HTTP request message

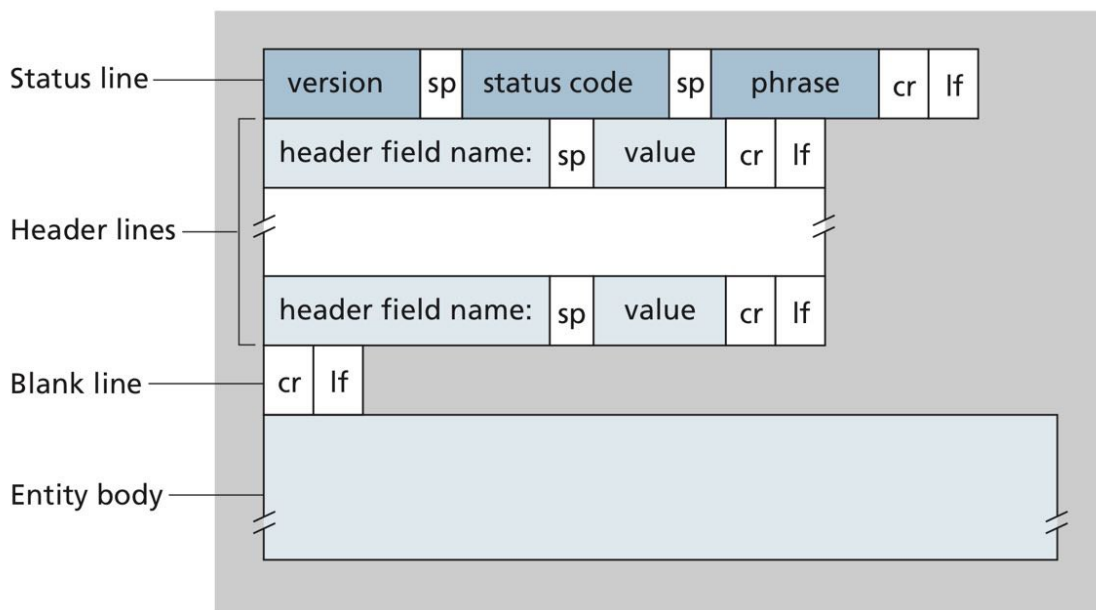


Figure 2.9 ♦ General format of an HTTP response message

User-Server Interaction: Cookies

Cookies, defined in [RFC 6265], allows sites to keep track of users. Cookie technology has four components: a cookies header line in HTTP response message, a cookie header line in the HTTP request message, a cookie file kept on the user's end system and managed by the user's browser, back-end database at the website.

File-Transfer Protocol (FTP)

FTP is an application-layer protocol that provides an ability to exchange files between server and client. FTP exploits two TCP connection: the control connection that serves for the command supply, and data protocol that transfers the requested data.

The general control connection is established at the default port 21.

The data control connection may be done in two ways:

- **Passive.** The client send the `PASV` command and waits for the server to send the address for the data transfer connection
- **Active.** The client send the `PORT` command providing the server with an address for the data transfer connection.

The data can be transferred in either binary or ASCII mode.

Electronic Mail

Electronic mail has been around since the beginning of the Internet. It was the most popular application when the Internet was in its infancy. The three major components that constitute the electronic mail are:

- User Agents
- Mail Servers
- Simple Mail Transfer Protocol (SMTP)

User agents allow users to read, reply to, forward, save and compose messages. When one is finished composing their message, their user agent sends the message to their mail server where the message is placed in the mail server's outgoing message queue.

Mail servers form the core of the e-mail infrastructure. Each recipient has a **mailbox** located in one of the mail servers. Bob's mailbox manages and maintains the messages that have been sent to them.

If one's message cannot be delivered, it is held in the sender's mail server which waits for the recipient's server to be turned on to try sending once again.

SMTP is the principal application-layer protocol for the Internet electronic mail. It uses the reliable data transfer server of TCP. As the most application-layer protocols, SMTP has two sides: a client side, which executes on the sender's mail server, and a server side, which executes on the recipient's mail server. Both client and server sides are run on every mail server.

SMTP, defined in RFC 5321, is at the heart of Internet electronic mail. To illustrate the basic operation of SMTP, let us consider the following scenario:

1. Alice invokes her user agent for e-mail, provides Bob's e-mail address, composes a message, and instructs the user agent to send the message
2. Alice's user agent sends the message to her mail server, where it is placed in a message queue.
3. The client side of SMTP, running on Alice's mail server, sees the message in the message queue. It opens a TCP connection to an SMTP server, running on Bob's mail server.
4. After some initial SMTP handshaking, the SMTP client sends Alice's message into the TCP connection.
5. At Bob's mail server, the server side of SMTP receives the message. Bob's mail server then places the message in Bob's mailbox.
6. Bob invokes his user agent to read the message at his convenience

It is important to observe that SMTP does not normally use intermediate mail servers for sending mail, even when the two mail servers are located at opposite ends of the world.

SMTP has TCP establish a connection at port 25 at the server SMTP. If the server is down, the client tries again later. Once the connection is established, the server and client perform some app-layer handshaking. Once that is done, the data transfer is started.

SMTP uses persistent connections, transferring multiple data frames within a single connection.

Mail Access Protocols

Once SMTP delivers message to one's mail server, the user executes it is user agent to retrieve the messages from the server. Today, there are two common ways to do it: using HTTP or IMAP (Internet Mail Access Protocol).

Domain Name System

Internet hosts can be identified in several ways. One identifier for a host is its hostname. Another identifier which is mostly used by applications is an IP address. The task of matching the hostname with an IP address is a task for the **DNS (Domain Name System)**.

The DNS is defined in two ways:

1. A distributed database implemented in a hierarchy of DNS servers
2. An application layer protocol that allows hosts to query the distributed database

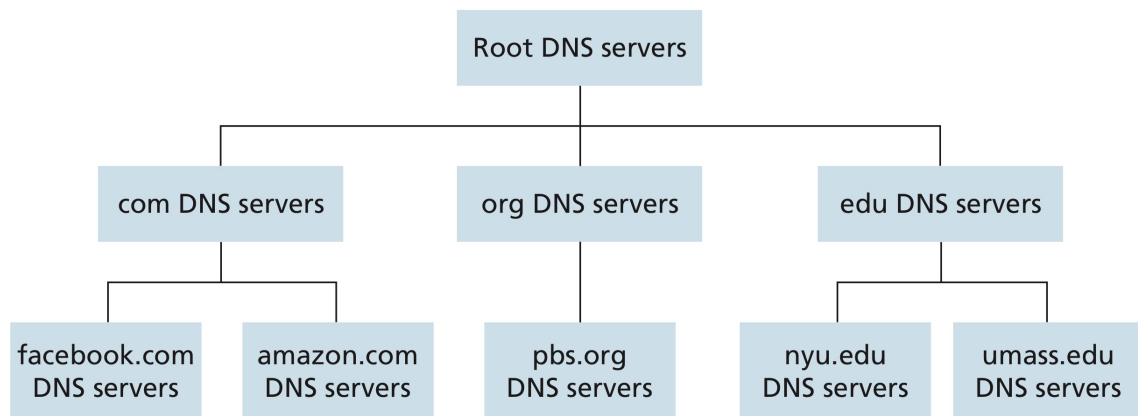
The DNS protocol runs over UDP and uses port 53.

DNS provides a few important services:

- Host Aliasing. A single host can be reached using several hostnames.
- Mail Server Aliasing
- Load Distribution. A single hostname can be shared over several endpoints.

The DNS is specified in RFC 1034 and RFC 1035.

The DNS is organized in a hierarchical fashion and distributed around the world. To a first approximation, there are three classes of DNS servers — **root DNS servers**, **top-level DNS servers** and **authoritative DNS servers**.



Another type of DNS servers are **local DNS server**. A local DNS server does not strictly belong to the hierarchy of servers but is nevertheless central to the DNS architecture. Each ISP — such as a residential ISP or an institutional ISP — has a local DNS server (also called a default name server).

The major advantage of DNS is **DNS caching** that provides an ability to significantly decrease the response time.

Messages and Records

The DNS servers that together implement the DNS distributed database store the **resource records (RRs)**. A resource record is a four-tuple that contains the following fields:

```
<NAME, VALUE, TYPE, TTL>
```

The meaning of **NAME** and **VALUE** depend on **TYPE**:

- If **TYPE=A**, then **NAME** is a hostname and **VALUE** is the IP address.
- If **TYPE=NS** then **NAME** is a domain and **VALUE** is the hostname of an authoritative DNS server that knows how to obtain the IP addresses for hosts in the domain.
- If **TYPE=CNAME** then **VALUE** is a canonical hostname for the alias hostname **NAME**.
- If **TYPE=MX** then **VALUE** is the canonical name of a mail server that has an alias hostname **NAME**.

If a DNS server is authoritative for a particular hostname, then the DNS server will contain a Type A record for the hostname. If not, then the server will contain a Type NS record for the domain that includes the hostname; it will also contain Type A record that provides the IP address of the DNS server in the **VALUE** field of the NS record.

DNS Messages

- The first 12 bytes is the *header section*, which has number of fields. The first field is a 16-bit number that identifies the query. This identifier is copied into the reply messages to a query
- The *questions* section contains information about the query that is being made.
- In reply from a DNS server, the *answer* section contains the resource records for the name that was originally queried.
- The *authority* section contains records of other authoritative servers.
- The *additional section* contains other helpful resources.

Identification	Flags	12 bytes
Number of questions	Number of answer RRs	
Number of authority RRs	Number of additional RRs	
Questions (variable number of questions)		Name, type fields for a query
Answers (variable number of resource records)		RRs in response to query
Authority (variable number of resource records)		Records for authoritative servers
Additional information (variable number of resource records)		Additional "helpful" info that may be used

Peer-to-Peer File Distribution

With peer-to-peer (P2P) architecture, there is minimal (or no) reliance on always-on infrastructure servers. In P2P file distribution, each peer can redistribute any portion of the file it has received to any other peers, thereby assisting the server in the distribution process.

Let us first determine the distribution time for the client-server architecture, which we denote by D_{cs} . In the client server architecture, none of the peers aids in the distributing the file. We make the following observations:

- The server must transmit one copy of the file to each of the N peers. Thus the server must transmit NF bits. Since the server's upload rate is u_s , the time to distribute the file must be at least

$$\frac{NF}{u_s}$$

- Let d_{\min} denote the download rate of the peer with the lowest download rate. This, the minimum distribution time is at least $\frac{F}{d_{\min}}$

Putting these two observation together, we obtain:

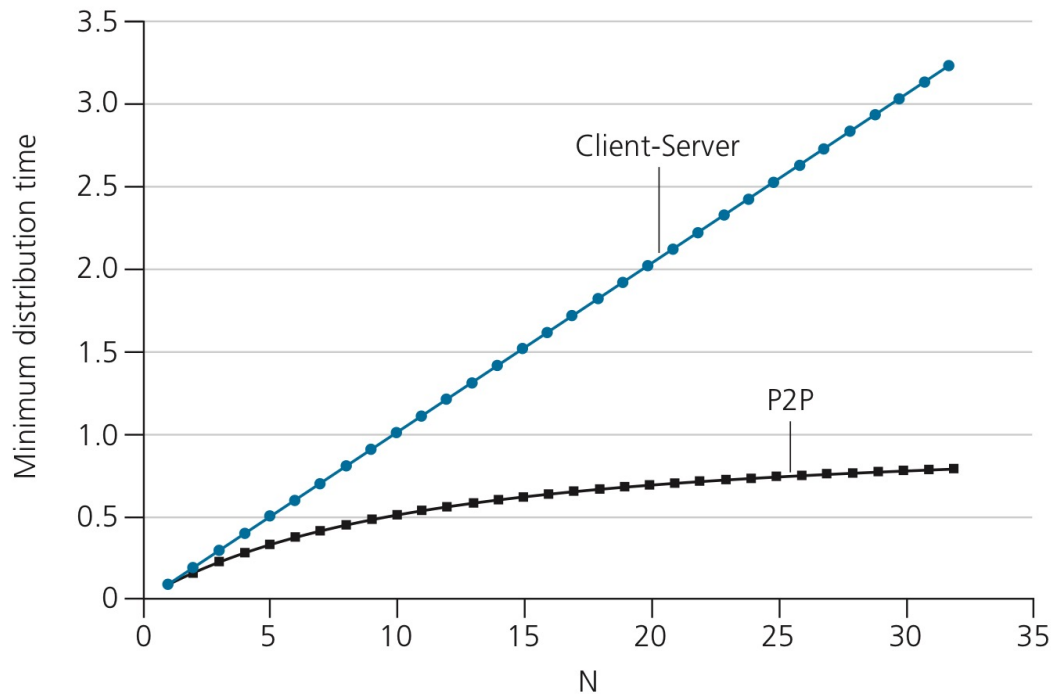
$$D_{cs} \geq \max\left\{\frac{NF}{u_s}, \frac{F}{d_{\min}}\right\}$$

When a peer receives some file data, it can use its own upload capacity to redistribute the data to other peers.

- At the beginning of the distribution, only the server has the file. To get this file into the community of peers, the server must send exact bit of the file at least once into its access link. Thus the minimum distribution time is at least $\frac{F}{u_s}$.
- As with the client server architecture, the peer with the lowest download rate cannot obtain all F bits of the file in less than $\frac{F}{d_{\min}}$ seconds. Thus, the minimum distribution time is at least $\frac{F}{d_{\min}}$
- Finally, observe the total upload capacity of the system as a whole is equal to the upload rate of the server plus the upload rate of each of the individual peers, that is $u_{\text{total}} = u_s + u_1 + \dots + u_N$. The must deliver (upload) NF bits to each of the N peers, thus delivering a total of NF bits. This cannot be done at a rate faster than u_{total} . Thus, the minimum distribution time is also at least $\frac{NF}{u_s + \sum_{i=1}^N u_i}$

Putting all observation together, we obtain the minimum distribution time for P2P, denoted by D_{P2P}

$$D_{\text{P2P}} \geq \max\left\{\frac{F}{u_s}, \frac{F}{d_{\text{min}}}, \frac{NF}{\sum_{i=1}^N u_i}\right\}$$



BitTorrent

In BitTorrent the collection of all peers participating in the distribution of a particular file is called a *torrent*. Peers in a torrent download equal-size *chunks* of the file from one another, with a typical chunk size of 256 KBytes. When a peer first joins a torrent, it has no chunks.

Each torrent has an infrastructure node called a *tracker*. When a peer joins a torrent, it registers itself with the tracker and periodically informs the tracker that it is still in the torrent. In this manner, the tracker keeps track of the peers that are participating in the torrent.

At any given instant of time, a user will have a subset of chunks and will know which chunks her neighbor peers have. In deciding which chunks to request, the user uses a technique called **rarest first**. In this manner, the chunks tend to be equally distributed among all peers.

After connection, a user choose the four most fast peers and recalculates their transmission rates updating the list of *unchoked* peers. Also, every 30 seconds, a user picks one additional neighbour and send it chunks.

Video Streaming and Content Distribution Networks

A video is a sequence of images, typically being displayed at a constant rate, for example, at 24 or 30 images per second. An uncompressed, digitally encoded image consists of an array of pixels. An *important characteristic* of video is that it can be compressed, thereby trading off video quality with bit rate.

Compressed Internet video typically ranges from 100 Kbps for low-quality video to over 4 Mbps for streaming high-definition movies.

HTTP Streaming and DASH

In HTTP streaming, the video is simply stored at an HTTP server as an ordinary file with a specific URL. When a user wants to see the video, the client establishes a TCP connection with the server and issues an HTTP `GET` request for that URL. The server then sends the video file, within an HTTP response message, as quickly as the underlying network protocols and traffic conditions will allow. On the client side, the bytes are collected in a client application buffer. Once the number of bytes exceeds a predetermined threshold, the client application begins playback — specifically, the streaming video application periodically grabs video frames from the application buffer.

The main drawback is that clients receive the same encoding of the video, despite the large variations in the amount of bandwidth available to a client.

In **DASH (Dynamic Adaptive Streaming over HTTP)** the video is encoded into several different versions, with each version having a different bit rate and, correspondingly, a different quality level. The client dynamically requests chunks of video segments of a few seconds in length.

With DASH, each video version is stored in the HTTP server, each with a different URL. The HTTP server also has a **manifest file**, which provides a URL for each version along with its bit rate. The client first requests the manifest file and learns about the various versions.

Content Distribution Networks

In order to meet the challenge of distributing massive amounts of video data to users distributed around the world, almost all major video-streaming companies make use of **Content Distribution Networks (CDNs)**. A CDN manages The CDN might be a private or a third-party one.

CDNs typically adopt one of two different server placement philosophies:

- **Enter Deep** into the access networks of Internet Service Providers by deploying server clusters in access ISPs all over the world.
- **Bring home** is a philosophy based on the building large clusters at a smaller number of sites.

When host is instructed to retrieve a specific media (identified by URL), the CDN must intercept the request so that it can (1) determine a suitable CDN server cluster for that client at that time, and (2) redirect the client's request to a server in that cluster.

At the core of any CDN deployment is a **cluster selection strategy**, that is mechanism for dynamically directing clients to a server cluster or a data center within CND.

One simple strategy is to assign the client to the cluster that is **geographically closest** to the end-user. Another strategy based on estimation of the current traffic (**real-time measurements**) and assigning user to the fastest CDN available.

Socket Programming

There are two types of network applications. One type is an implementation whose operation is specified in a protocol standard, such as an RFC or some other standards document. Another is a proprietary network application, that exploits a protocol described not in publicly viewable document.

Socket Programming with UDP

Before the sending process can push a packet of data using UDP, it must first attach a destination address to the packet. The destination host's IP address is a part of destination address. When a socket is created, an identifier, called a **port number**, is assigned to it. So, the packet's destination address also includes the socket's port number.

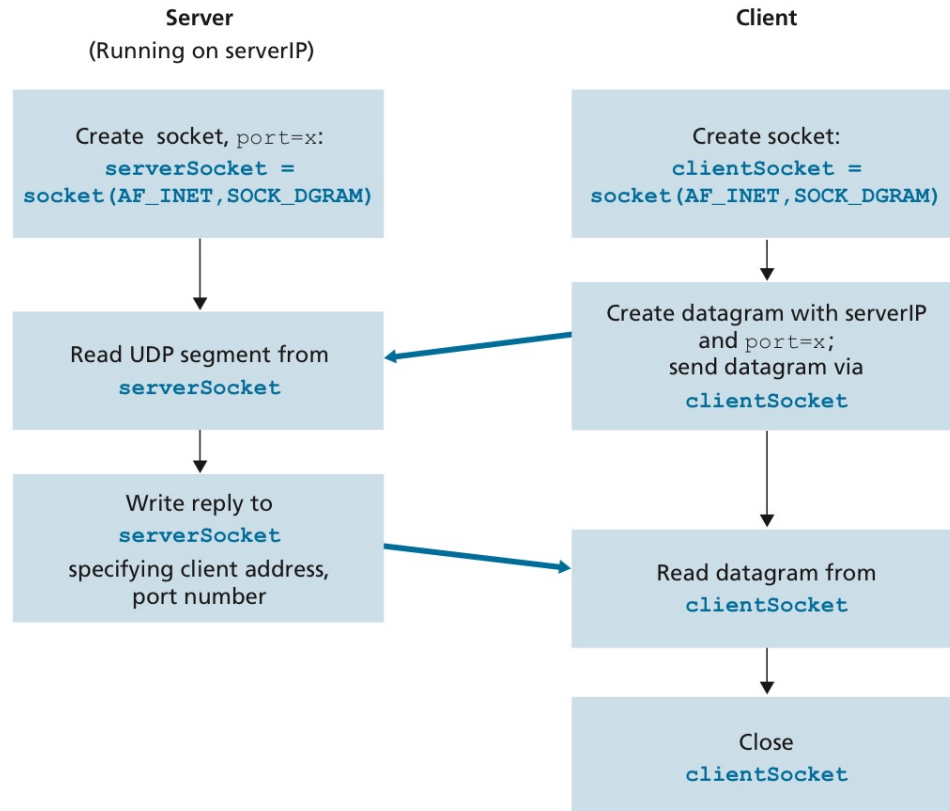


Figure 2.27 ♦ The client-server application using UDP

Socket Programming with TCP

Unlike UDP, TCP is a connection-oriented protocol. This means that before the client and server can start to send data to each other, they first need to handshake and establish a TCP connection. With the TCP connection established, when one side wants to send data to the other side, it just drops the data into the TCP connection via its socket.

The server has a special socket (welcoming socket) that is used to initiate the connection between server and client. Once the request is received via welcoming socket, a new connection via another socket is established for further communication.

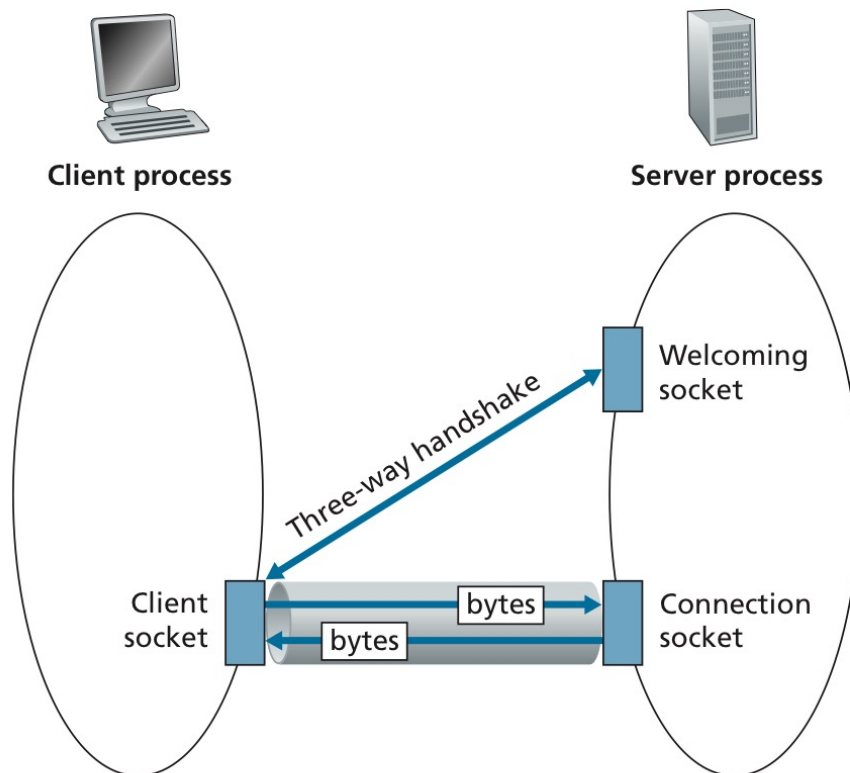


Figure 2.28 ♦ The TCPServer process has two sockets

Chapter III. Transport Layer

Residing between the application and network layers, the transport layer is a central piece of the layered network architecture. There exists two fundamental problems in computer networks, that are resolved by the transport-layer:

1. How two entities can communicate reliable over a medium that may lose and corrupt data?
2. How to control the transmission rate of transport-layer entities in order to avoid, or recover from, congestion within the network?

Transport-Layer Services

A transport-layer protocol provides for **logical communication** between application processes running on different hosts. On the sending side, the transport layer converts the application-layer messages it receives from a sending application process into transport-layer packets, known as transport-layer **segments** in Internet terminology.

Overview of the Transport Layer in the Internet

The Internet makes two distinct transport-layer protocols available to the application layer. One of these protocols is **UDP**(User Datagram Protocol), which provides an unreliable, connectionless service to the invoking application. The second of these protocols is **TCP** (Transmission Control Protocol).

The Internet's network-layer protocol has a name — IP, for Internet Protocol. IP provides logical communication between hosts. The IP service model is **best-effort delivery service**. This means that IP makes its best efforts to deliver data, but makes no guarantee about it. For these reasons, IP is said to be an **unreliable service**.

Extending host-to-host delivery to process-to-process delivery is called **transport-layer multiplexing and demultiplexing**. UDP and TCP also provide integrity checking by including error-detection fields in their segments' headers. TCP offers several additional services to applications. First and foremost, it provides **reliable data transfer**. Using flow control, sequence numbers, acknowledgments, and times, TCP ensures that data is delivered from sending process to receiving process, correctly and in order. TCP also provides congestion control.

Multiplexing and Demultiplexing

The job of delivering the data in a transport-layer segment to the correct socket is called **demultiplexing**. The job of gathering data chunks at the source host from different sockets, encapsulating each data chunk with header information to create segments, and passing the segments to the network layer is called **multiplexing**.

The transport-layer multiplexing requires that sockets have unique identifiers and each segment have special fields that indicate the socket to which the segment is to be delivered.

These special fields are the **source port number field** and **destination port number field**. Each port number is a 16-bit number, ranging from 0 to 65535. The port numbers

ranging from 0 to 1023 are called **well-known port numbers** and dedicated to the famous protocols.

Connectionless Multiplexing and Demultiplexing

A UDP socket is fully identified by a two-tuple consisting of a destination IP address and destination port number. As a consequence, if two UDP segments have different source IP addresses and/or source port numbers, but the same destination IP Address and destination port number, then the two segments will be directed to the same destination process via the same destination socket

Connection-Oriented Multiplexing and Demultiplexing

A TCP socket is identified by a four-tuple:

$$(\text{source IP, source port, destination IP, destination port})$$

Thus, when a TCP segment arrives from the network to a host, the host uses all four values to direct (demultiplex) the segment to the appropriate socket.

Consider the following example:

- The TCP server application has a “welcoming socket”, that waits for connection-establishment requests from TCP clients on port number 12000.
- The TCP client creates a socket and sends a connection establishment request segment.
- A connection-establishment request is nothing more than a TCP segment with destination port number 12000 and a special connection-establishment bit set in TCP header. The segment also includes a source port number that was chosen by the client.
- When the host operating system of the computer running the server process receives the incoming connection-request segment with destination port 12000, it locates the server process is waiting to accept a connection on port number 12000.
- Also, the transport layer at the server notes the following four values in the connection-request segment: (1) the source port number in the segment, (2) the IP address of the source host, (3) the destination port number in the segment, and (4) its own IP address. The newly created connection socket is identified by these four

values; all subsequently arriving segments whose four mentioned values coincide with these four values will be demultiplexed to this socket.

UDP Protocol

UDP, defined in [RFC 768], does just about as little as a transport protocol can do. In fact, if the application developer chooses UDP instead of TCP, then the application is almost directly talking with IP.

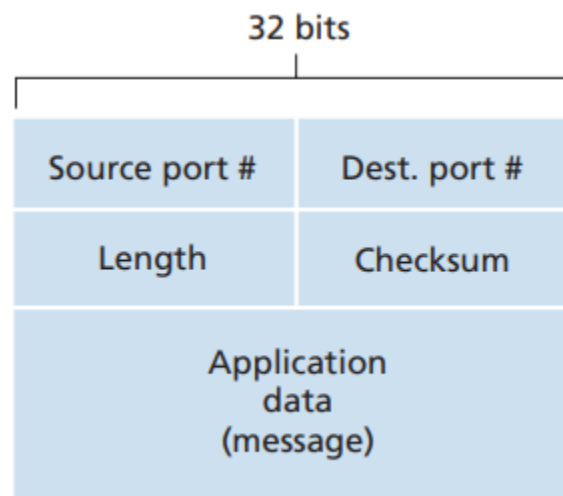
UDP takes messages from the application process, attaches source and destination port number fields for the multiplexing/demultiplexing service, adds two other small fields (checksum and length), and passes the resulting segment to the network layer.

The main advantages of UDP transport protocol are:

- No connection establishment is required
- No connection state is saved
- Small packet header overhead

However, it is possible to have a reliable data transfer using UDP (QUIC).

UDP Segment



UDP Checksum

The checksum for particular package is computed as a one's complement of a sum of all 16-bit words in the segment, with any overflow encountered during the sum being

wrapped around.

Principles of Reliable Data Transfer

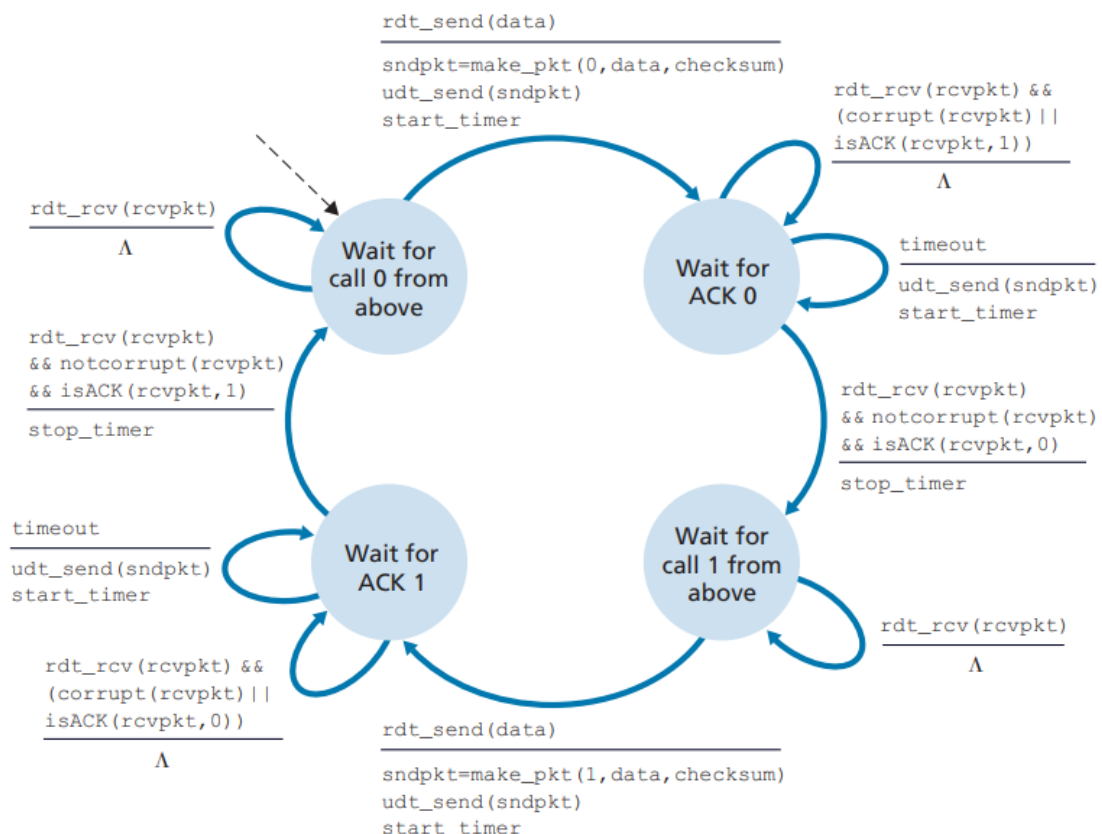
During the packet exchanging process, there different problems such are packet loss and packet error might arise. As a responsibility of reliable transport layer protocols, the ability to recover from that problems should be introduced to provide a transportation guarantee.

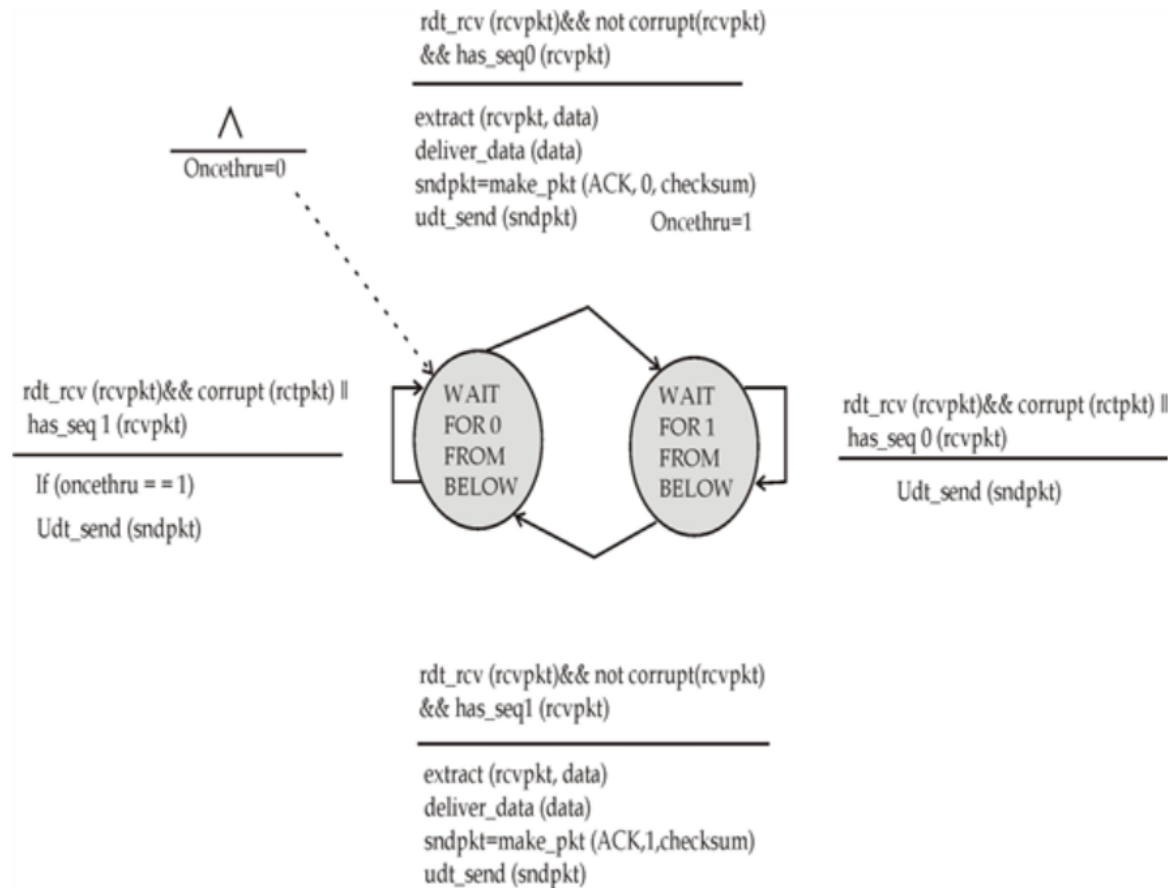
The crucial notions that are introduced in reliable data transfer are the **positive acknowledgment (ACK)** and **negative acknowledgment (NAK)**. In particular, acknowledgments are packets sent be receiver to inform whether it received data without any loss or error.

If any packet is lost or corrupter, it should be resent by the sender endpoint.

To avoid duplicates at the receiver endpoint, the packets should contain a **sequence number** that facilitates distinguishing duplicates from new packets.

The most advanced reliable data transfer protocol is summarized in two FSMs below.





Pipelined Reliable Data Transfer Protocol

Using the pipelining has the following consequences for reliable data transfer protocols:

- The range of sequence numbers must be increased since each in-transit packet must have a unique sequence number and there may be multiple, in-transit, unacknowledged packets
- The sender and receiver sides of the protocols may have to buffer more than one packet. Minimally, the sender will have to buffer packets that have been transmitted but not yet acknowledged. Buffering of correctly received packets may also be needed at the receiver.
- The range of sequence numbers needed and the buffering requirements will depend on the manner in which a data transfer protocol responds to lost, corrupted and overly delayed packets. Two basic approaches toward pipelined error recover are **sliding window protocol** and **selective repeat**.

Sliding Window Protocol

In the protocol the sender is allowed to transmit multiple packets without waiting for an acknowledgment, but is constrained to have no more than some maximum allowable number, N , of unacknowledged packets in pipeline.

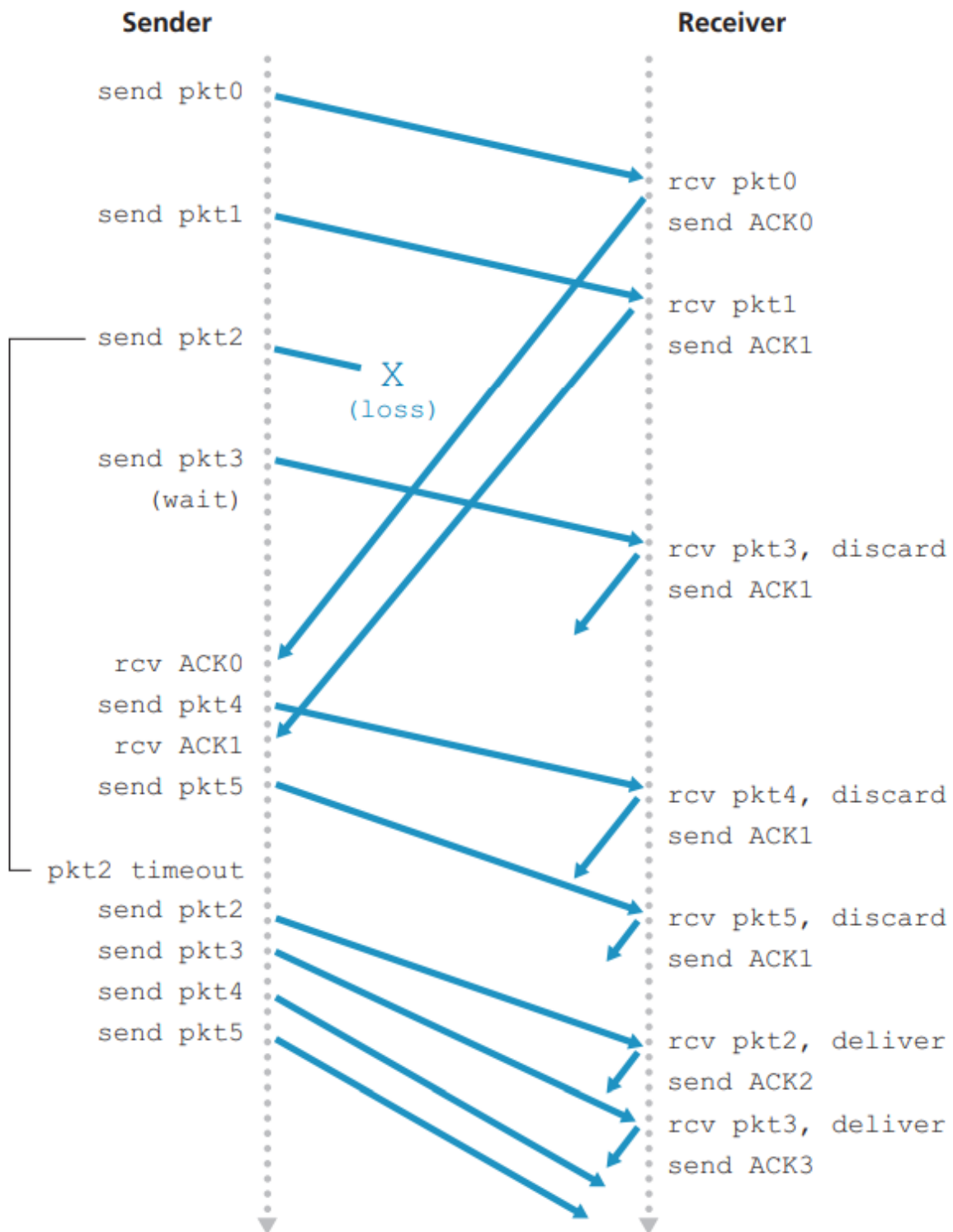
In practice, a packet's sequence number is carried in a fixed-length field in the packet header. With a finite range of sequence numbers, all arithmetic involving sequence numbers must then be done using modulo 2^k arithmetic. The TCP protocol has a 32-bit sequence number field, where TCP sequence numbers count bytes in the byte stream rather than packets.

The SW protocol must respond to three types of events:

- *Invocation from above.* The sender first checks whether the window is full. If not, a packet is created and sent, and variables are appropriately updated. If window is full, the sender simply returns the data back to the upper layer.
- *Receipt of ACK.* In the protocol, an acknowledgment for a packet with sequence number n will be taken to be a **cumulative acknowledgment**, indicating that all packets with a sequence number up to and including n have been correctly received
- *A timeout event.* If a timeout occurs, the sender resends *all* packets that have been previously sent but that have not yet been acknowledged.

On the receiver side, if a packet with sequence number n is received correctly and is in order (that is, the data last delivered to the upper layer came from a packet with sequence number $n-1$), the receiver sends an **ACK** for packet n and delivers the data portion of the packet to the upper layer.

In all other cases, the receiver discards the packet and resends an ACK for the most recently received in-order packet.



Selective Repeat

As in the sliding window, the number of packets are devoted as a window with the $base$ being the sequence number of the every first packet in the window.

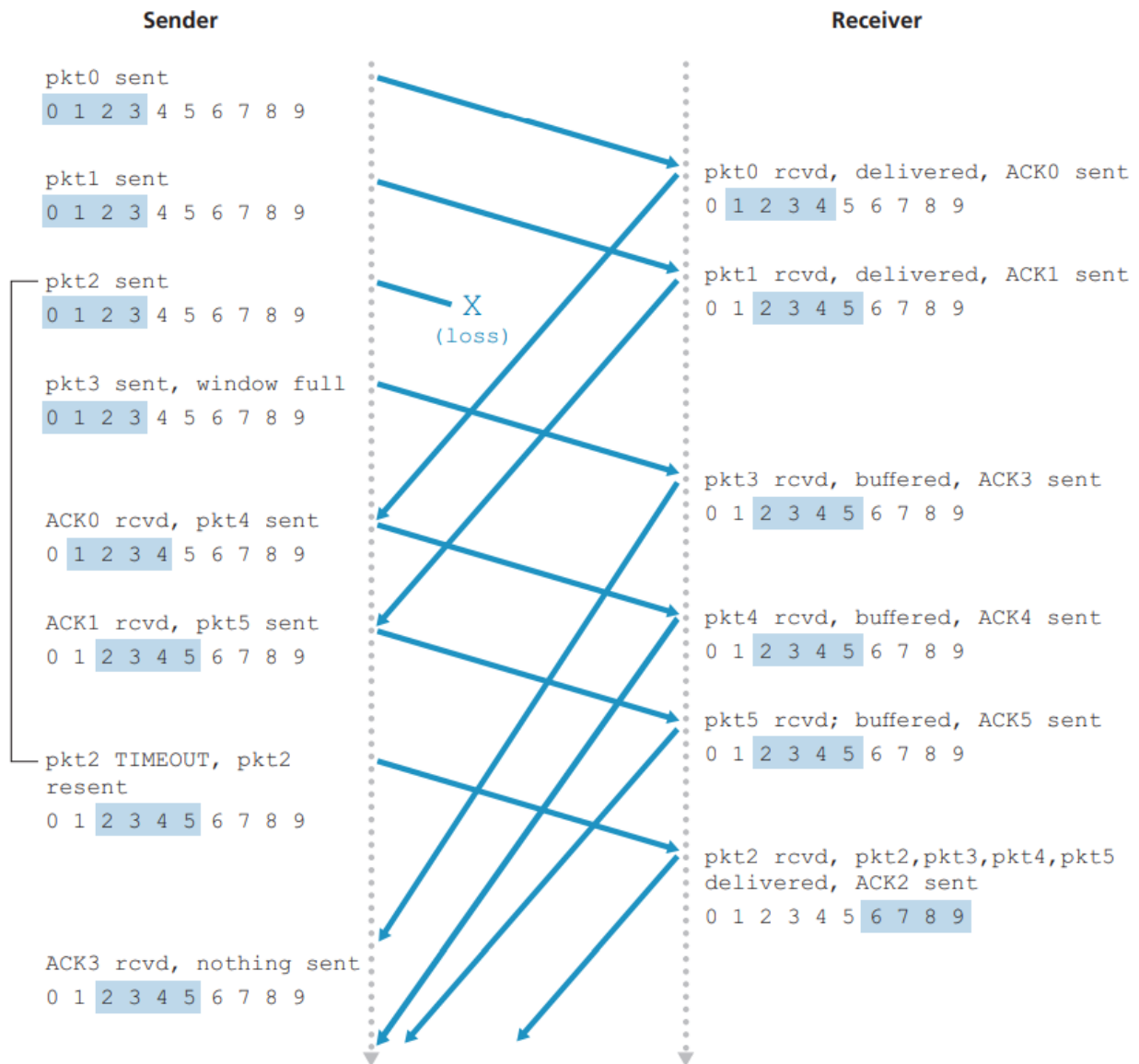
The behaviour of a sender is the following:

1. *Data received from the above.* When data is received from above, the SR sender checks the next available sequence number for the packet. If the sequence number is within the sender's window, the data is packetized and sent; otherwise it is either buffered or returned to the upper layer for later transmission.
2. *Timeout.* The principle is the same with the key difference that know each packet will have its logical timer.
3. *ACK received.* As acknowledgment received, if the packet sequence number is $base$ then the $base$ number is changed to the number of the least unacknowledged packet and untransmitted packages are transmitted.

The receiver also has a window of the packets with addresses: $[rbase, rbase+N-1]$

The behaviour of a receiver is the following:

1. If received packet falls into the window, send the selective acknowledgment. If the sequence number is $base$, then move the window,
2. If received packet's sequence number is less than $base$, then the packet is again acknowledged.
3. Any other packets are ignored.



Connection-Oriented Transport: TCP

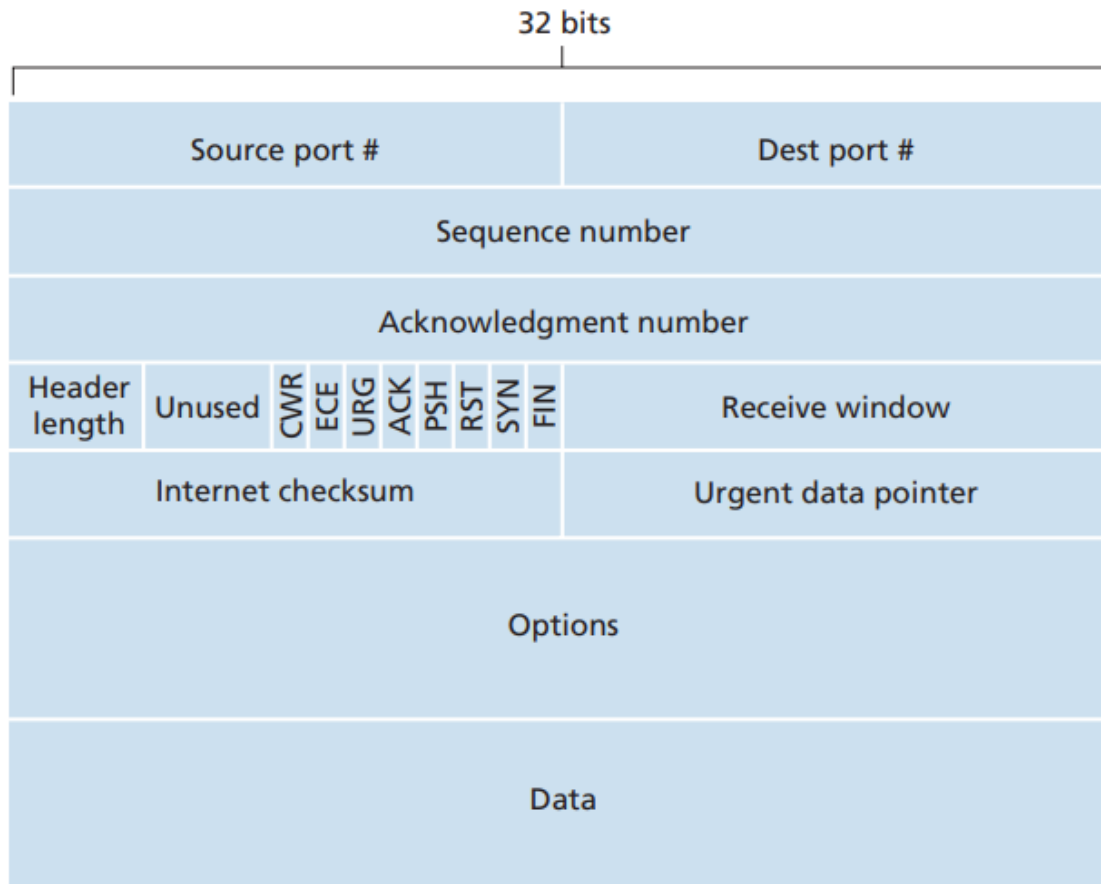
TCP is said to be connection oriented, because to transmit any data, two endpoints must firstly perform a handshake.

A TCP connection provides a **full-duplex** meaning that data can flow in any direction in the connection. Connection is always **point-to-point**, meaning that multicasting (delivering messages from one host to many others within one connection is impossible)

Once a TCP connection is established, the two applications processes can send data to each other. The maximum amount of data that can be grabbed and places in a segment is limited by the **maximum segment size (MSS)**. The MSS is typically set by first

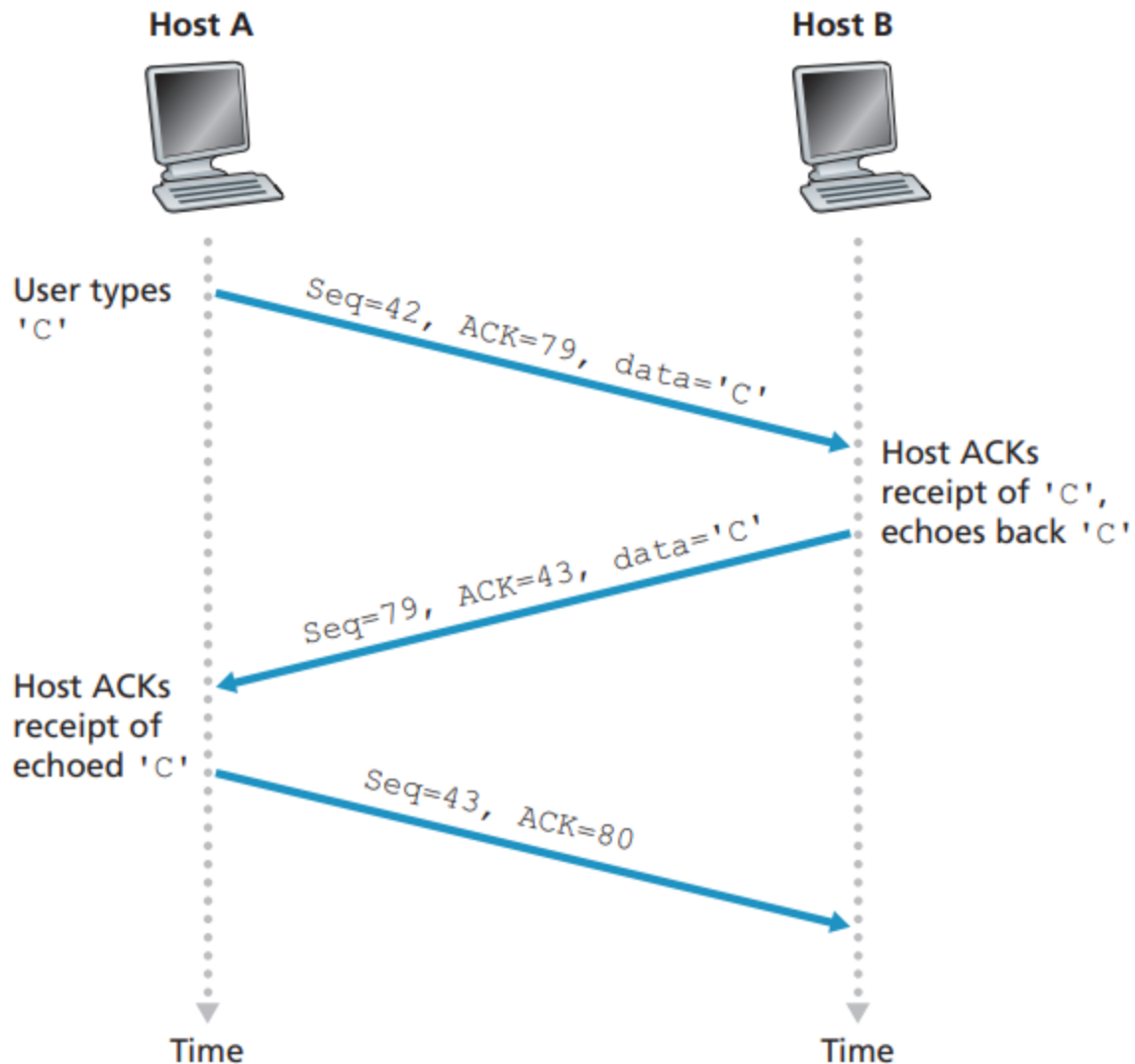
determining the length of the largest link-layer frame that can be sent by the local sending host (the so-called **maximum transmission unit, MTU**),

TCP Segment Structure



Sequence Numbers and Acknowledgment Numbers

The **sequence number** is the byte-stream number of the first byte in the segment. The **acknowledgment number** is the sequence number of the next byte that is awaited by the receiver. TCP is said to provide **cumulative acknowledgments**.



Round-Trip Time Estimation and Timeout

Although this is conceptually simple, many subtle issues arise when we implement a timeout/retransmit mechanism in an actual protocol such as TCP. Clearly, the timeout should be larger than the connection's round-trip time (RTT).

TCP measures RTT once a time, not for every single packet, and does not take retransmitted packets into consideration. Since the `SampleRTT` might fluctuate with time, it is need to keep track over the current approximation. The variable `EstimatedRTT` is responsible for this and is updated using the weighted average.

$$\text{EstimatedRTT} = (1 - \alpha) \cdot \text{EstimatedRTT} + \alpha \cdot \text{SampleRTT}$$

It is also beneficial to keep track of deviation of RTT:

$$\text{DevRTT} = (1 - \beta) \cdot \text{DevRTT} + \beta \cdot |\text{SampleRTT} - \text{EstimatedRTT}|$$

The timeout is set as:

$$\text{TimeoutInterval} = \text{EstimatedRTT} + 4 \cdot \text{DevRTT}$$

An initial `TimeoutInterval` value of *1 second* is recommended. Also, when a timeout occurs, the value of `TimeoutInterval` is doubled to avoid premature timeout occurring for a subsequent segment that will soon be acknowledged. However, as soon as a segment is received and `EstimatedRTT` is updated, the `TimeoutInterval` is recomputed using the formula above.

Reliable Data Transfer

TCP creates a **reliable data transfer** service on top of IP's unreliable best-effort service. TCP's reliable data transfer service ensures that the data stream that a process reads out of its TCP receive buffer is uncorrupted, without gaps, without duplication, and in sequence.

The TCP uses only a single retransmission timer, even if there are multiple transmitted but not yet acknowledged segments. The TCP sender acts in the following way:

- Once the data received from application above, the packet is assembled. Then, the timer is started (if not yet started) and the number of the next packet's sequence number is updated accordingly
- If the timer strikes, the not-yet-acknowledged segment with the smallest sequence number is being retransmitted.
- If acknowledgment received, the sender base number is updated if the number is greater than the current base (cumulative acknowledgment)

The TCP receiver actions can be summarized as:

Event	TCP Receiver Action
Arrival of in-order segment with expected sequence number. All data up to expected sequence number already acknowledged.	Delayed ACK. Wait up to 500 msec for arrival of another in-order segment. If next in-order segment does not arrive in this interval, send an ACK.
Arrival of in-order segment with expected sequence number. One other in-order segment waiting for ACK transmission.	Immediately send single cumulative ACK, ACKing both in-order segments.
Arrival of out-of-order segment with higher-than-expected sequence number. Gap detected.	Immediately send duplicate ACK, indicating sequence number of next expected byte (which is the lower end of the gap).
Arrival of segment that partially or completely fills in gap in received data.	Immediately send ACK, provided that segment starts at the lower end of gap.

Fast Retransmit

The timeout delay might be quite long while the TCP sender can investigate the lost packet. With the use of **duplicate ACKs** the sender might retransmit the package much earlier before the timeout.

Flow Control

TCP provides a **flow-control service** to its applications to eliminate the possibility of the sender overflowing the receiver's buffer. Flow control is thus a speed-matching service — matching the rate at which the sender is sending against the rate at which the receiving application is reading.

TCP provides flow control by having the *sender* maintain a variable called the **receive window**. Informally, the receive window is used to give the sender an idea of how much free buffer space is available at the receiver.

At sender side two variables are stored: *Last Acknowledged Byte* and *Last Sent Byte*. Their difference is the amount of unacknowledged (sent) data and thus cannot exceed the receive window.

At receiver side three variables are used: *Last Received Byte*, *Last Read Byte*, *Receiver Buffer*. The last stands for the total memory available for the data buffering. Therefore, the receive window is computed as:

$$\text{Receive Window} = \text{Receive Buffer} - (\text{Last Received Byte} - \text{Last Read Byte})$$

Since the sender learns about current receive window size via acknowledgment packets, in order to prevent blocking, it does not stop sending when the window is zero,

but sends byte per packet to maintain an ability for receiver to refresh the receive window size.

Principles of Congestion Control

To treat the cause of network congestion, mechanisms are needed to throttle senders in the face of network congestion.

The cost of a congested network can be described as:

- Large queueing delays are experienced as the packet-arrival rate nears the link capacity.
- The sender must perform retransmissions in order to compensate for dropped (lost) packets due to buffer overflow
- Unneeded retransmissions by the sender in the face of large delays may cause a router to use its link bandwidth to forward unneeded copies of a packet
- When a packet is dropped along the path, the transmission capacity that was used at each of the upstream links to forward that packet to the point at which it is dropped ends up having been wasted

Approaches to Congestion Control

1. *End-to-end congestion control.* In this approach to congestion control, the network layer provides no explicit support to the transport layer for congestion-control purposes.
2. *Network-assisted congestion control.* With this approach, routers provide explicit feedback to the sender and/or receiver regarding the congestion state of the network.

TCP Congestion Control

If a TCP sender perceives that there is a little congestion on the path between itself and the destination, then the TCP sender increases its send rate; if the sender perceives that there is congestion along the path, then the sender reduces its send rate.

The TCP congestion-control mechanism operating at the sender keeps track of an additional variable, the **congestion window**.

Because TCP uses acknowledgements to trigger its increase in congestion window size, TCP is said to be **self-clocking**.

The key principles the TCP protocol follows:

- A lost segment implies congestion, and hence, the TCP sender's rate should be decreased when a segment is lost.
- An acknowledged segment indicates that the network is delivering the sender's segments to the receiver, and hence, the sender's rate can be increased when an *ACK* arrives for a previously unacknowledged segment
- The TCP sender increases its transmission rate to probe for the rate at which congestion onset begins, backs off from that rate, and then begins probing again to see if the congestion onset rate has changed

Chapter IV. Network Layer: Data Plane

Unlike the transport and application layers, there is a piece of the network layer in each and every host and router in the network. The network layer can be decomposed into two interacting parts, the **data plane** and the **control plane**.

The primary data-plane role of each router is to forward datagrams from its input links to its output links.

The primary role of the network control plane is to coordinate these local, per-router forwarding actions so that datagrams are ultimately transferred end-to-end, along paths of routers between source and destination hosts.

Forwarding and Routing

- *Forwarding*. When a packet arrives at a input link of a router, the router must move the packet to the appropriate output link.
- *Routing*. The job of determining the route or path taken by packets as they flow from sender to a receiver. The algorithms that calculate these paths are referred to as **routing algorithms**.



A key element in every router is its **forwarding table**.

Control Plane

It is worth mentioning that the job to keep forwarding tables updated is a job of software part of the system that is the control plane.

Network Service Model

The network might provide the following services:

 Service	 Description
<u>Guaranteed delivery.</u>	The service guarantees that a packet send by a source host will eventually arrive at the destination port
<u>Guaranteed delivery with bounded delay.</u>	The same as above, but delivers within a specified host-to-host delay bound
<u>In-order packet delivery.</u>	The service guarantees that packets arrive at the destination in the order that they were sent
<u>Guaranteed minimal bandwidth</u>	Preserves the constant transmission rate between hosts
<u>Security</u>	The service ensures that all datagrams will be encrypted

The Internet's network layer protocol provides no services

Internals of Router

A high-level view of a generic router architecture is shown in the following figure

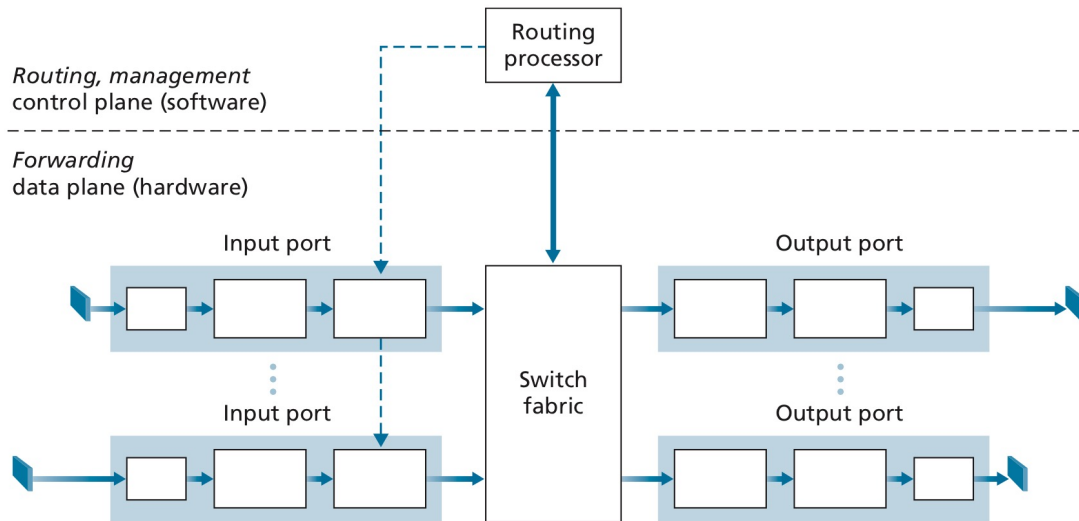


Figure 4.4 ♦ Router architecture

Four components can be identified:

- *Input ports.* An **input port** performs several key functions: the physical layer function of terminating an incoming physical link at a router, the link layer function of interoperating with the link layer at the other side of the incoming link, a lookup and forwarding function.
- *Switching fabric.* The switching fabric connects the router's input ports to its output ports.
- *Routing processor.* The routing processor performs control-plane functions
- *Output ports.* An **output port** also perform some key functions: storing packets from switching fabric, transmitting it to an outgoing link applying certain link-layer and physical-layer procedures

Input Port Processing and Destination-Based Forwarding

The forwarding table is either computed and updated by the routing processor (using a routing protocol to interact with the routing processors in other network routers) or is received from a remote SDN controller.

The router uses the **longest prefix matching** rule when matches incoming port with outgoing. Also, it compresses the forwarding table by only storing the common prefixes of the addresses.

Once a packet's output port has been determined via the lookup, the packet can be sent into the switching fabric. In some designs, a packet may be temporarily blocked from entering the switching fabric if packets from other input ports are currently using the fabric.

Switching

The switching fabric is at the very heart of a router. Switching can be accomplished in a number of ways:

- *Switching via memory.* The switching on earliest routers was done under direct control of CPU (routing processor). Input and output ports functioned as traditional I/O devices in a traditional operating system. An input port with an arriving packet first signaled the routing process via an interrupt. Then the packet is copied to the processor memory where the processor performs necessary actions with it
- *Switching via a bus.* In this approach, an input port transfers a packet directly to the output port over a shared bus without any intervention. It is usually done by attaching a special label to the packet. All output ports receive this packet, and all but one discards it using the attached label.
- *Switching via an interconnection network.* A crossbar switch is an interconnection network consisting of $2N$ busses connecting N input ports to N output ports. This method is **non-blocking** but can force packets to await before the bus is free.

Output Port Processing

The output port processing task is storing packets in the output port's memory and retransmitting it over the link. This includes selecting and de-queueing packets for transmission, and performing the needed link-layer and physical-layer transmission functions.

Queueing

The packets at the input ports might be queued due to the corresponding outgoing links now occupied or some other packet that goes before is blocked due to that reason (Head-of-Line blocking).

The packet at the output ports might experience queueing congestion due to that the rate of incoming packets is larger than the transmission rate of the outgoing link.

When there is not enough memory to buffer an incoming packet, a decision must be made to either drop the arriving packet (drop-tail policy) or drop some of the already queued packets. Sometimes the packets are dropped before the queue is full to provide a congestion signal to the sender.

Packet Scheduling

Scheduling algorithms:

- *First-Come-First-Served (FIFO)*
- *Priority Queueing*
- *Round Robin*. Divides input packets into groups, then circularly switches from one group to another.
- *Weighted Fair Queueing*. Same as round-robin, but assigns weights to the groups letting packets from one of them be sent more frequently.

The Internet Protocol (IP): IPv4, Addressing, IPv6

There are two versions of IP in use today: IPv4 [RFC 791] and IPv6 [RFC 2460; RFC 4291].

IPv4 Datagram Format

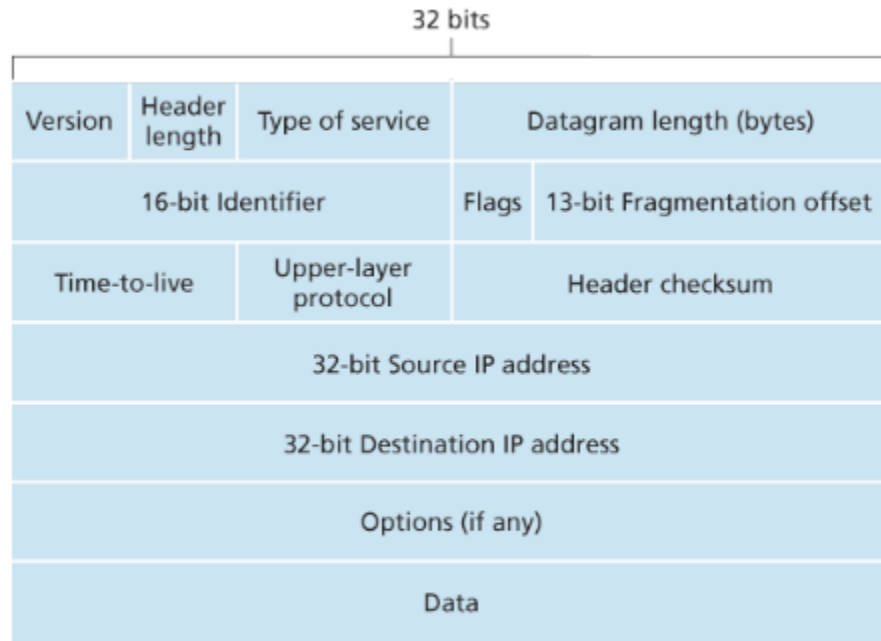


Figure 4.16 IPv4 datagram format

- *Version number*. By looking at the version the node can understand how to interpret the datagram.
- *Header length*. The header can contain variable number of fields.
- *Type of Service*. Helps in congestion and flow control.
- *Identifier, flags, fragmentation offset*. These three fields have to do with so-called IP fragmentation
- *Data length*. 16-bit field: the total length of the IP datagram
- *Protocol*. This field is typically used by upper layer to determine to which protocol data portion of this datagram should be passed.
- *Header checksum*. Necessary for error-detecting of the datagram's header.

Addressing

IPv4 address in 32-bit number usually written in dotted-decimal notation. The address is associated with the interface rather than with router or host.

Isolated networks are often referred as **subnet**. IP addressing is assigning an address to the subnet using the subnet mask.

The Internet's address assignment strategy is known as **Classless Interdomain Routing (CIDR)**. CIDR generalises the notion of subnet addressing. As with subnet addressing, the 32-bit IP address is divided into two parts and again has the dotted decimal form $a.b.c.d/x$ where x indicates the number of bits in the first part of the address.

The most significant bits of an address of the form $a.b.c.d/x$ constitute the network portion of the IP address, and are often referred to as the **prefix** (or *network prefix*). Only these bits are considered by routers outside the organization's network.

Obtaining a Host Address: The Dynamic Host Configuration Protocol

Host addresses are typically configured using **Dynamic Host Configuration Protocol (DHCP)** [RFC 2131]. DHCP allows a host to obtain an IP address automatically. A network administrator can configure DHCP so a host can obtain the same IP address from time to time, or it may be assigned a **temporary address** for each session.

The DHCP work is described in the following steps:

1. *Server discovery*. New client connected to the server sends UDP segment containing destination address `255.255.255.255:67` and source address as `0.0.0.0`. The broadcasted message is received by DHCP server.
2. *Server offer(s)*. The server allocates a new address for the client and broadcasts the message with necessary configuration to the network, attaching an identification to it to be recognized by requesting client.
3. *Request*. The client sends *DHCP request message* with the configuration it has received.
4. *Acknowledgment*. DHCP receives the message from the just configured client and sends an acknowledgment.

DHCP also specifies lease time that points out how long the given IP address will be valid. It also offers an interface for the lease time prolongation.

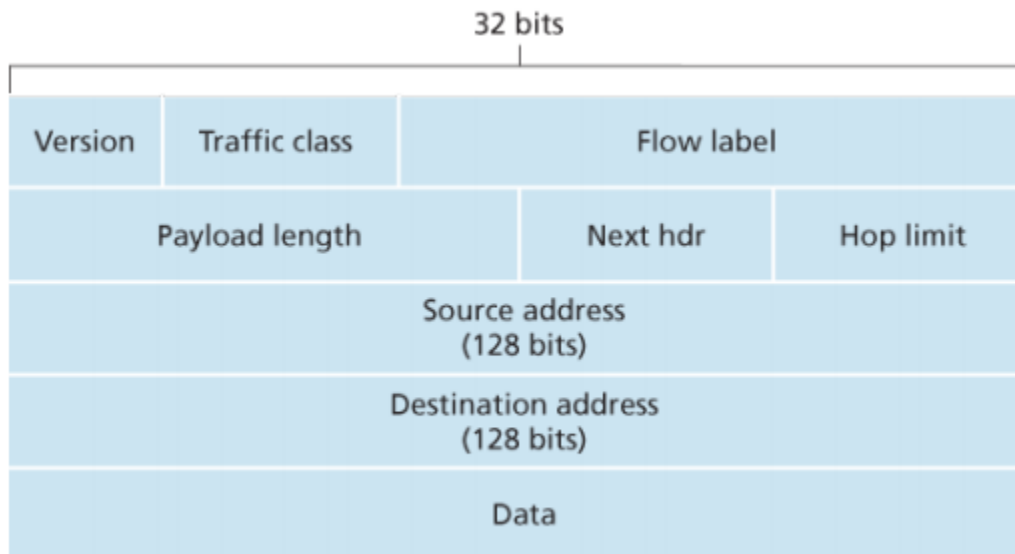
Network Address Translation (NAT)

To provide an ability to allocate larger amount of addresses, the new technology named Network Address Translation has been introduced. As defined in [RFC 1918], the address blocks

10.0.0.0	-	10.255.255.255	(10/8 prefix)
172.16.0.0	-	172.31.255.255	(172.16/12 prefix)
192.168.0.0	-	192.168.255.255	(192.168/16 prefix)

defines **private networks address space**. The devices in private network can be assigned with one of these addresses. When a making a request to the host located outside the private network, the gateway router will change the IP address and port of the packet to the real one, and send it to the outside host. When receives the packet back, router will again match the port and private IP address and retranslate the packet to the device in private network

IPv6 Addressing



The sixth version of IP protocol experienced considerable changes:

- No error checking, since it is provided by lower and upper layers and is frequently redundant
- Fixed header size — 40 bytes
- Enlarged addresses sizes
- No fragmentation: router sends ICMP message “Packet too big” if it cannot store or forward it.

- No options field (next header makes it possible to distinguish between UDP and TCP segments)

The **transition** between IPv4 and IPv6 are performed using tunneling (e.g. storing IPv6 header in IPv4 payload)

Generalized Forwarding

The forwarding may be performed in match-plus-action manner, where match can be performed over any of the fields of transport, network and link layer; and action can be either forwarding, dropping or modifying (or any combination).

Middleboxes

Any intermediary box performing functions apart from normal, standard functions of an IP router on the data path between a source host and destination host is referred as middlebox

Three categories can be identified:

- *NAT Translation.*
- Security Services
- Performance Enhancement


Chapter V. The Network Layer: Control Plane

In this chapter the methods for computing forwarding and flow tables is discussed.

There are two possible approaches:

- *Per-router control.* Routing algorithm runs in each and every router; both a forwarding and a routing function are contained within each router. The OSPF and BGP protocols are base on this approach
- *Logically centralized control.* The center node computes and distributes forwarding tables.

Routing Algorithms

<u>Aa</u> Classification	 Description
<u>Location</u>	Centralized routing algorithms compute the path using the global knowledge of the network layout. Decentralized routing algorithm only operates with the data that is available to a single router
<u>Stability</u>	Static algorithms assumes network does not change rapidly, while dynamic algorithms are performed at frequently changed networks.
<u>Load Sensitivity</u>	The algorithms that change the behavior based on the congestion level in the network is called load-sensitive, while those which are not are called load-insensitive.

Intra-AS Routing in the Internet: OSPF

The routers can be organized into **autonomous systems (AS)** with each AS consisting of a group of routers that are under the same administrative control. Routers within the same AS all run the same routing algorithm and have information about each other.

Open Shortest Path First (OSPF)

OSPF is a link-state protocol that uses flooding of link-state information and a Dijkstra's least cost path algorithm. With OSPF, each router constructs a complete topological map of the entire autonomous system.

Some of the advances embodied in OSPF include the following:

- *Security.* Exchanges between OSPF routers can be authenticated to prevent malicious injections
- *Multiple same-cost paths.* When multiple paths to a destination have the same cost, OSPF allows multiple paths to be used.
- *Support for multicasting*
- *Hierarchy support*

Routing Among the ISPs: BGP

Border Gateway Protocol (BGP) is used to determine forwarding tables between autonomous systems.

BGP provides each router a means to:

1. *Obtain prefix reachability information from neighboring ASs*
2. *Determine the 'best' routes to the prefixes*

Advertising BGP Route Information

Each subnet has a router that is dedicated to communicate with the routers in other ASs. Such routers are called **gateway routers**, while routers that communicate inside the network are called **internal**.

BGP uses a TCP connection over the port 179. Each TCP connection, along with all the BGP messages sent over the connection, is called **BGP connection**.

BGP connection can be **external (eBGP)** and **internal (iBGP)**.

When advertising a prefix along the network, BGP includes some attributes to the prefix. Prefix with its attributes is called a *BGP route*.

Two of the more important attributes are **AS-PATH** and **NEXT-HOP**.

1. **AS-PATH** contains the list of ASs through which the advertisement has passed.
2. **NEXT-HOP** is the *IP address of the router interface that begins the AS-PATH*.

Hot-Potato Routing

In hot potato routing, the route chosen (from among all possible routes) is that route with the least cost to the **NEXT-HOP** router beginning that route.

Route-Selection Algorithm

For any given destination prefix, the input into BGP's route-selection algorithm is the set of all routes to that prefix that have been learned and accepted by the router. The following steps are performed until the single route is remained:

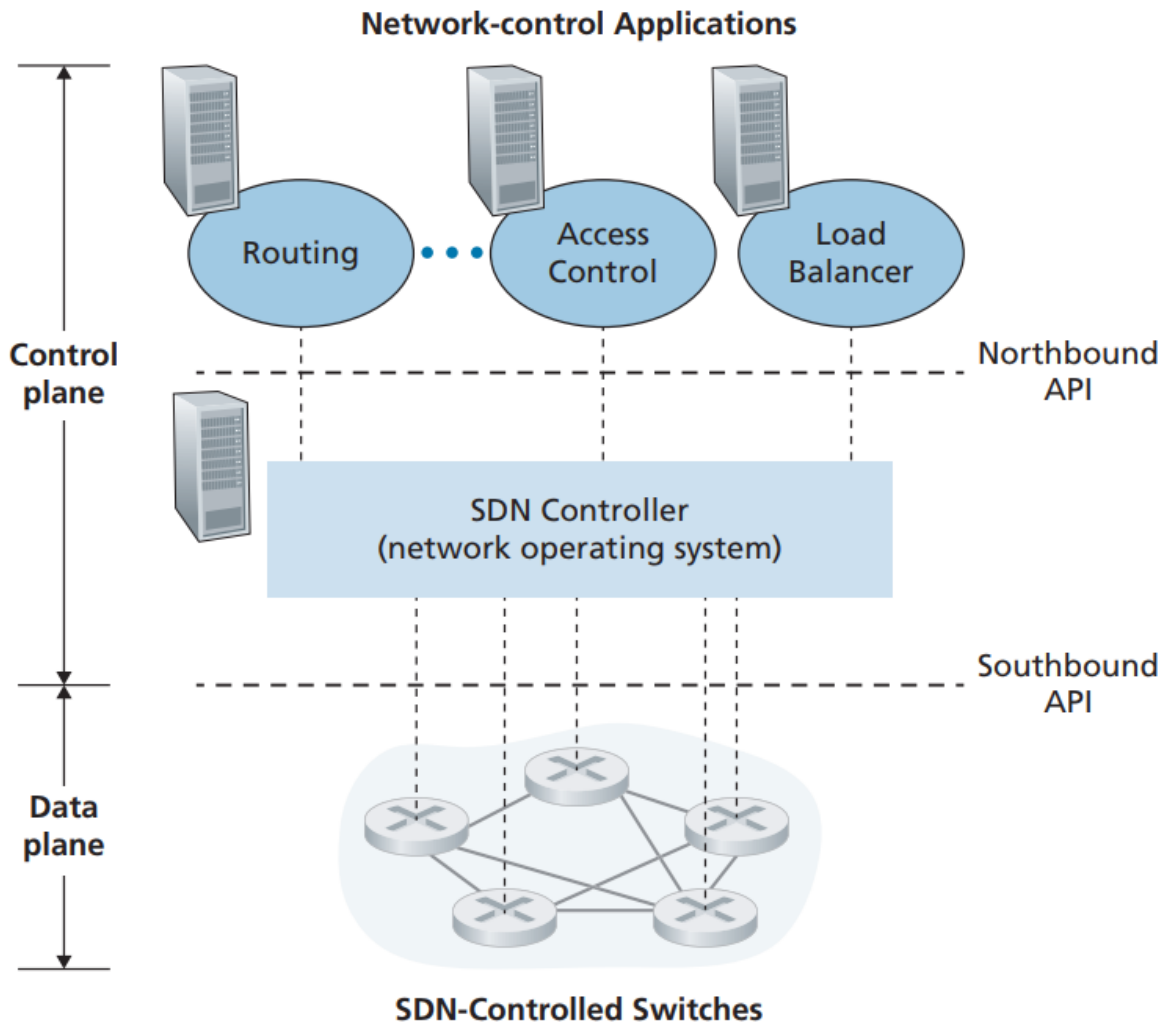
1. A route is assigned a **local preference** value as one of the attributes. The meaning of the value is determined by AS administrator. The routes with the highest local preference value are chosen
2. From the remaining routes the route with shortest AS-PATH is selected
3. Then it is the hot potato routing applied to the set of remained routes

4. If more than one route remained, the BGP identifier is used to choose one

The SDN Control Plane

Key characteristics:

1. *Flow-based forwarding.* Packet forwarding by SDN-controlled switches can be based on any number of header field values in the transport-layer, network-layer, or link-layer header
2. *Separation of data plane and control plane.* The data plane consists of the network's switches—relatively simple (but fast) devices that execute the “match plus action” rules in their flow tables. The control plane consists of servers and software that determine and manage the switches' flow tables.
3. *Network control functions: external to data-plane switches.* Unlike traditional routers, however, this software executes on servers that are both distinct and remote from the network's switches
4. *A programmable network.* The network is programmable through the network control applications running in the control plane. These applications represent the “brains” of the SDN control plane, using the APIs provided by the SDN controller to specify and control the data plane in the network devices



Chapter VI: The Link Layer and LANs

The link layer is responsible for communication along a single link between two network entities.

Introduction

Any device running a link-layer protocol is referred as **node**. The communication path between any two nodes is referred as a **link**.

Over a given link, a transmitting node encapsulates the datagram in a **link-layer frame** and transmits the frame into the link.

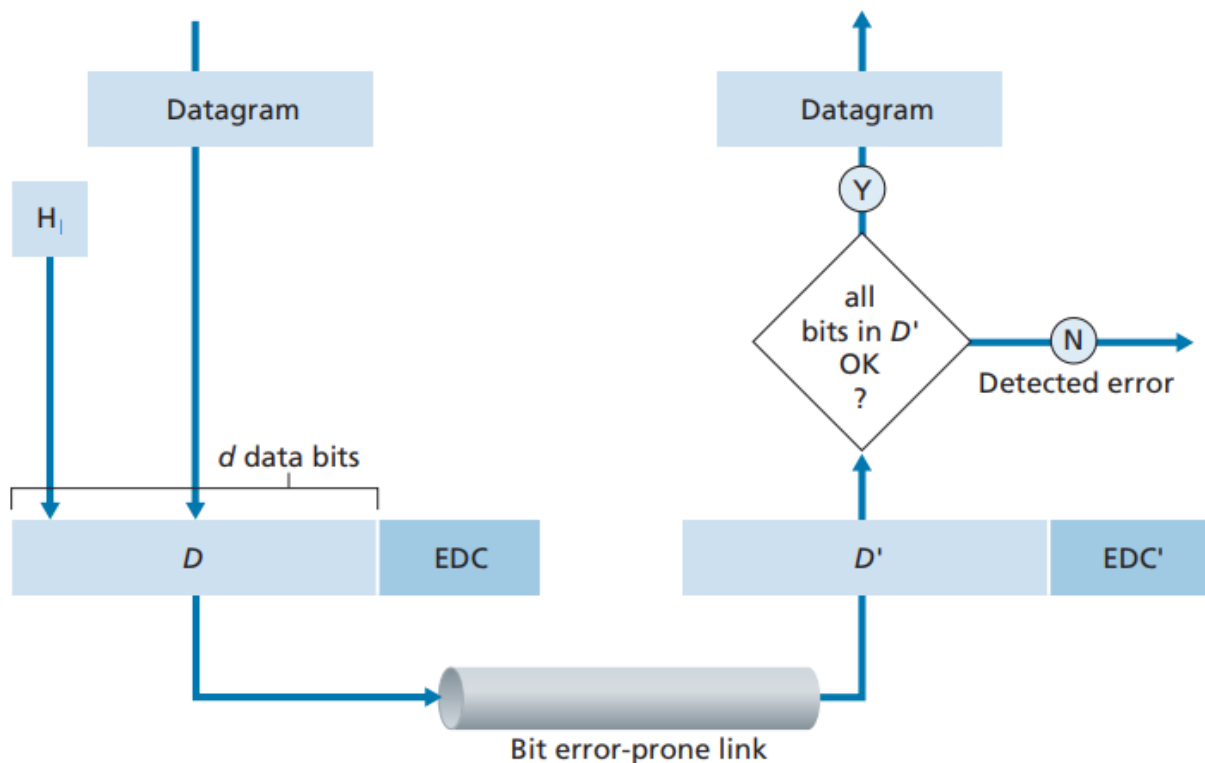
Services

- *Framing.* Almost all link-layer protocols encapsulate each network-layer datagram within a link-layer frame before transmission over the link
- *Link access.* A medium access control (MAC) protocol specifies the rules by which a frame is transmitted onto the link.
- *Reliable transfer.* When a link-layer protocol provides reliable delivery service, it guarantees to move each network-layer datagram across the link without error.
- *Error detection and correction*

Implementation

For the most part, the link layer is implemented on a chip called the network adapter, also sometimes known as a network interface controller (NIC)

Error-Detection and -Correction Techniques



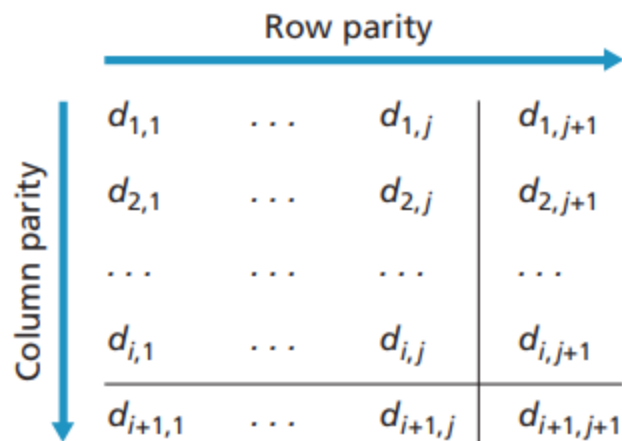
Parity Checks

Suppose the datagram D that has to be sent contains d bits.

<u>Aa</u> Even Parity	<u>≡</u> Odd Parity
If there is odd number of one bits in the » <u>augment it with a one bit, else augment it with</u> <u>zero bit</u>	If there is even number of one bits in the » augment it with a one bit, else augment it with zero bit.

On the receiver side the parity of the number of 1 bits in the received data is checked. If the parity does not correspond to expected one, the odd number of bit errors is detected. The method does not detect even number of bit errors.

In *two-dimensional* parity check scheme the data D is divided into i rows and j columns and $i+j+1$ bits are added to the data. With multidimensional parity check, receiver can detect the wrong bit and correct it.

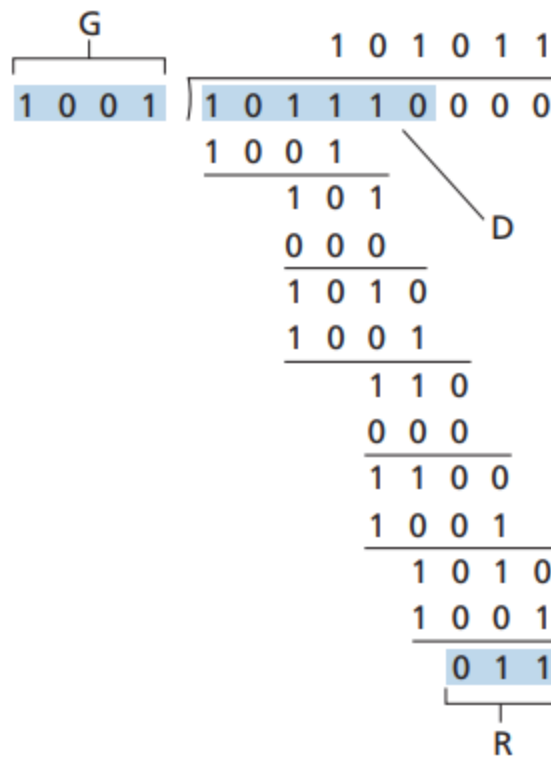


No errors							Correctable single-bit error						
1	0	1	0	1	1		1	0	1	0	1	1	
1	1	1	1	0	0		1	0	1	1	0	0	Parity error
0	1	1	1	0	1		0	1	1	1	0	1	
0	0	1	0	1	0		0	0	1	0	1	0	

Cyclic Redundancy Check (CRC)

The data of d bits can be viewed as a polynomial with 0-1 coefficients. The receiver and sender agree with $r+1$ bit pattern known as generator G . For a given piece of data, D , the sender will choose r additional bits, R , and append them to D such that the resulting $d+r$ bit pattern is exactly divisible by G using modulo-2 arithmetic.

The process of error checking with CRCs is thus simple: The receiver divides the $d+r$ received bits by G . If the remainder is nonzero, the receiver knows that an error has occurred; otherwise the data is accepted as being correct.



Multiple Access Links and Protocols

Some links can have multiple sending and receiving nodes all connected to the same, single, shared broadcast channel.

Because all nodes are capable of transmitting frames, more than two nodes can transmit frames at the same time. When this happens, all of the nodes receive multiple frames at the same time; that is, the transmitted frames **collide** at all of the receivers

Channel Partitioning Protocols

<u>Aa</u> Protocol	Description
<u>TDMA</u>	Divides the time into slots allowing a single node transmit in the allocated time slot
<u>FDMA</u>	Divides the channel into several frequency slots, allowing several nodes transmit at different slots simultaneously
<u>CDMA</u>	Assigns a different code to each node allowing them transmit simultaneously without any collision by encoding the data with the given code

Random Access Protocols

The second broad class of multiple access protocols are random access protocols. In a random access protocol, a transmitting node always transmits at the full rate of the channel, namely, R bps. When there is a collision, each node involved in the collision repeatedly retransmits its frame (that is, packet) until its frame gets through without a collision. But when a node experiences a collision, it doesn't necessarily retransmit the frame right away. Instead it waits a random delay before retransmitting the frame

Slotted ALOHA

- All frames consist of exactly L bits
- Time is divided into slots of size L/R seconds
- Nodes start to transmit frames only at the beginnings of slots
- The nodes are synchronized so that each node knows when the slot begins
- If two or more frames collide in a slot, then all the nodes detect the collision event before the slot ends

The nodes try to send the frame during some slot, if the slot is occupied, then they reschedule transmission to the next slot with probability p .

For the N active nodes, the efficiency of the protocol is $Np(1-p)^{N-1}$

ALOHA

Works similar as the slotted version, but does not have slots and is fully decentralized. When a frame first arrives, the node immediately transmits the frame in its entirety into the broadcast channel. If the collision is detected, the node awaits for transmission time and then retries to send the frame, or with probability of p it retransmits immediately after it finished transmission of collided frame.

For the N active nodes, the efficiency of the protocol is $Np(1-p)^{2(N-1)}$

CSMA/CD

The protocol exploits the notion of **carrier sensing** — ability of the node to 'sense' the channel and determine whether it is busy or not.

The CSMA/CD protocol can be summarized as:

1. As the frame arrives, it is prepared for transmission
2. As the channel is sensed free, the transmission begins
3. While transmitting, the adapter monitors for the presence of signal energy coming from other adapters using the broadcast channel
4. If the adapter transmits the entire frame without detecting signal energy from other adapters, the adapter is finished with the frame. If, on the other hand, the adapter detects signal energy from other adapters while transmitting, it aborts the transmission (that is, it stops transmitting its frame)
5. After aborting, the adapter waits a random amount of time and then returns to step 2.

When transmitting a frame that has already experienced n collisions, a node chooses the value of K at random from $\{0, 1, 2, \dots, 2^n - 1\}$. Thus, the more collisions experienced by a frame, the larger the interval from which K is chosen.

In order to present a closed-form approximation of the efficiency of Ethernet, let d_{prop} denote the maximum time it takes signal energy to propagate between any two adapters. Let d_{trans} be the time to transmit a maximum-size frame. The efficiency is then:

$$\text{Efficiency} = \frac{1}{1 + 5 \frac{d_{\text{prop}}}{d_{\text{trans}}}}$$

Taking Turns Protocols

The protocols of this kind can be divided into: polling algorithms and token-passing protocol. In polling protocol the **master node** is dedicated to control the flow of the channel access among the nodes. In token-passing protocols, nodes notify the successor nodes that they can now retransmit.

Switched LANs

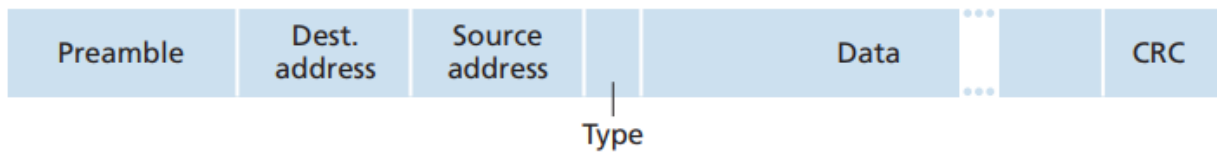
MAC Address

Each node (or being precisely, network interface) has a designated 6-byte address. This address is unique across the world and assumed to be fixed. It is used to determine whether the frame sent to the link is intended for the certain device by comparing MAC addresses. The broadcast MAC address for Ethernet and Wi-Fi is `FF:FF:FF:FF:FF:FF`

Address Resolution Protocol

To match IP address and MAC address, the ARP protocol request packet is broadcasted by the host who desires to retrieve the MAC address corresponding to certain IP address. The packet is received by all hosts and then is proceeded by the one whose IP address matches with the one requested. Then this host sends ARP response message with a proper MAC address.

Ethernet



The Ethernet frame begins with an 8-byte preamble field. Each of the first 7 bytes of the preamble has a value of 10101010; the last byte is 10101011. The Ethernet protocol is a CSMA/CD link-layer protocol.

Link-Layer Switches

The switch is transparent to the hosts and routers in the subnet. Filtering is the switch function that determines whether a frame should be forwarded to some interface or should just be dropped. Forwarding is the switch function that determines the interfaces to which a frame should be directed, and then moves the frame to those interfaces. Both functions are performed using **switch-table**.

Address	Interface	Time
62-FE-F7-11-89-A3	1	9:32
7C-BA-B2-B4-91-10	3	9:36
....

Chapter VII: Wireless and Mobile Networks

Introduction



We can identify the following elements in a wireless network:

- *Wireless hosts.*
- *Wireless links.*
- *Base station.* The **base station** is a key part of the wireless network infrastructure. Unlike the wireless host and wireless link, a base station has no obvious counterpart in a wired network. A base station is responsible for sending and receiving data to and from a wireless host that is associated with that base station.

Hosts associated with a base station are often referred to as operating in **infrastructure mode**. In **ad hoc networks**, wireless hosts have no such infrastructure with which to connect. In the absence of such infrastructure, the hosts themselves must provide for services such as routing, address assignment, DNS-link name translation, and more.

When a mobile host moves beyond the range of one base station and into the range on another, it will change its point of attachment into the larger network (**handoff and handover**)

Taxonomy

 Infrastructure-based	 Ad-hoc
<u>Single hop. Any communication performed over just one hop</u>	Single hop ad-hoc networks chooses one single device to operate as a base station
<u>Multi hop. Some wireless nodes may have to relay their communication through other wireless nodes in order to communicate via base station</u>	Multi hop ad-hoc networks consists of dozen independent wireless hosts that can relay the communication among them to reach the destination

Wireless Links and Network Characteristics

As compared with wired networks, wireless communication has the following properties:

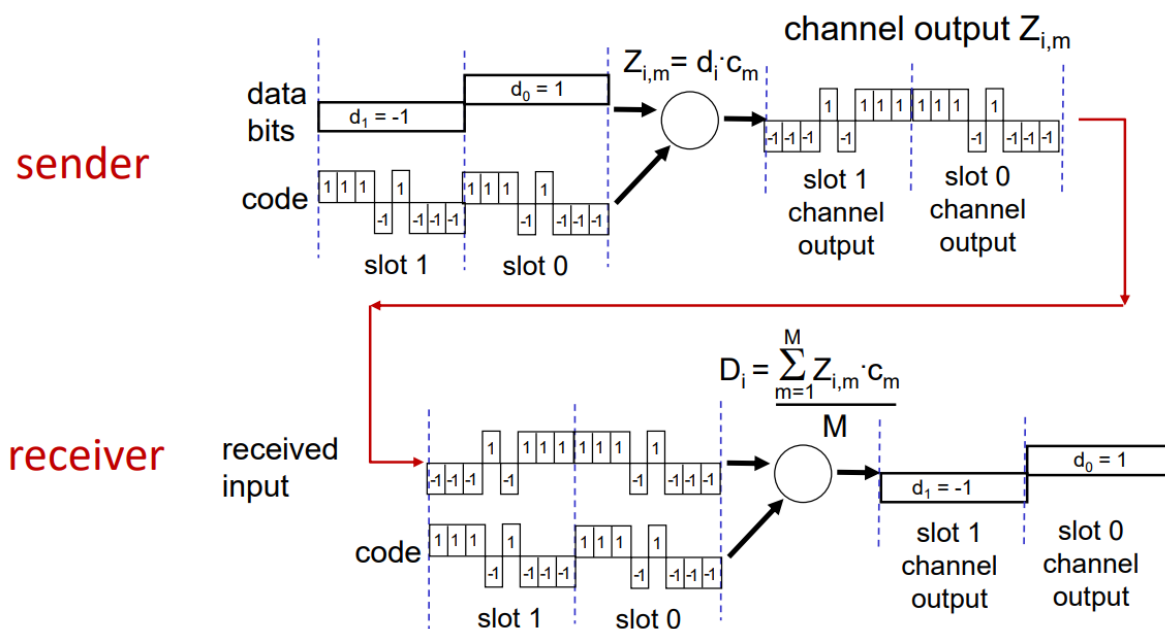
- Signal strength decreases with distance
- Signals from different sources may interfere

- Signals may reflect off objects and the ground causing asynchronization and errors

The **signal-to-noise ratio (SNR)** is a relative measure of the strength of the received signal and this noise.

CDMA (Code-division multiple access)

In a CDMA protocol, each bit being sent is encoded by multiplying the bit by a signal (the code) that changes at a much faster rate (known as **chipping rate**).



WiFi: 802.11 Wireless LANs

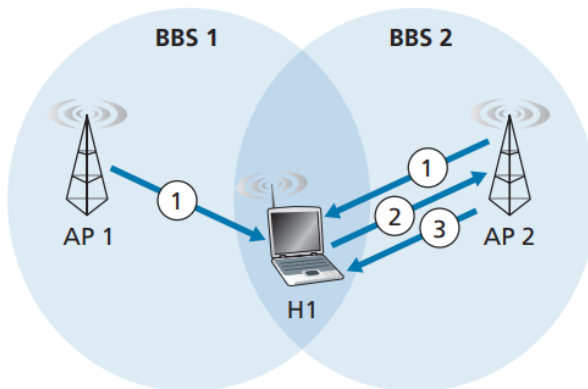
The fundamental building block of the 802.11 architecture is the **basic service set (BSS)**. A BSS contains one or more wireless stations and a central base station, known as an **access point (AP)** in 802.11 parlance.

When a network administrator installs an AP, the administrator assigns a one-or two-word **Service Set Identifier (SSID)** to the access point. WiFi works within 85 MHz band, divided into 11 channels that are partially overlapped with each other.

Architecture

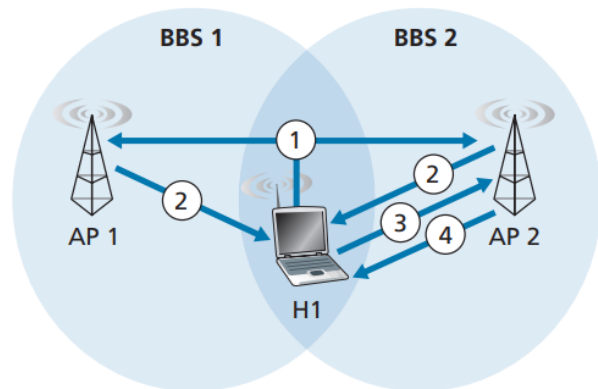
The 802.11 standard requires APs to periodically send **beacon frames** each of which includes the AP's SSID and MAC address. Using this beacon frames a wireless host

can connect to certain AP.



a. Passive scanning

1. Beacon frames sent from APs
2. Association Request frame sent: H1 to selected AP
3. Association Response frame sent: Selected AP to H1

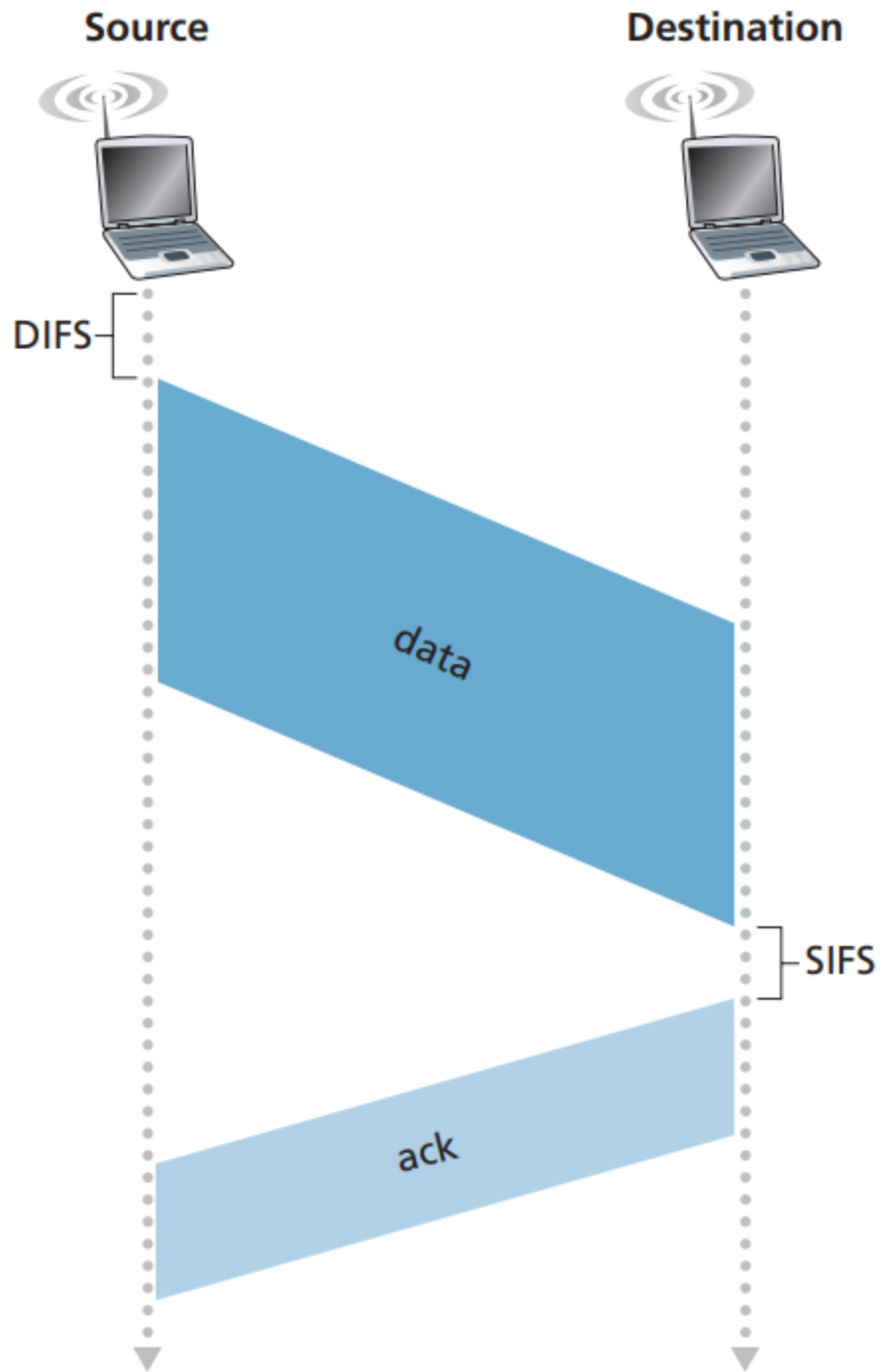


a. Active scanning

1. Probe Request frame broadcast from H1
2. Probes Response frame sent from APs
3. Association Request frame sent: H1 to selected AP
4. Association Response frame sent: Selected AP to H1

MAC protocol

802.11 uses the random access protocol that is referred as **CSMA with collision avoidance (CSMA/CA)**. The protocol senses the channel before transmission and refrains transmission if the channel is busy. Also, this protocol uses acknowledgment/retransmission scheme to provide insurance for data transmission.



As transmission began, the whole data frame is transmitted. This is possible due to collision avoidance technique used in the protocol.

When the destination station receives a frame that passes the CRC, it waits a short period of time known as the **Short Inter-frame Spacing (SIFS)** and then sends back an

acknowledgment frame.

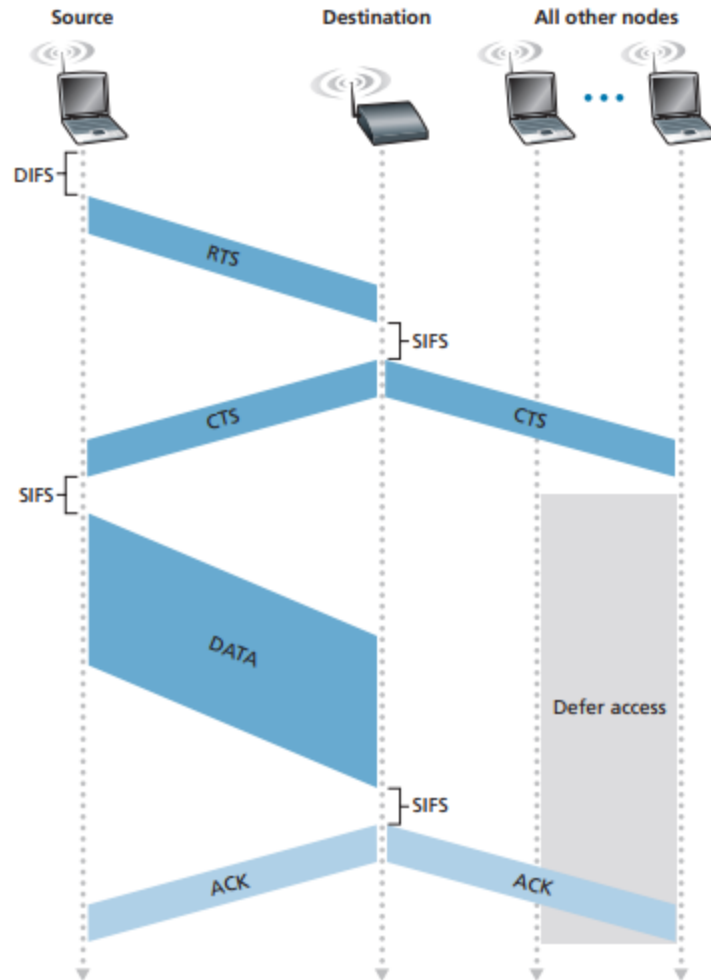
If the transmitting station does not receive an acknowledgment within a given amount of time, it assumes that an error has occurred and retransmits the frame, using the CSMA/CA protocol to access the channel. If an acknowledgment is not received after some fixed number of retransmissions, the transmitting station gives up and discards the frame.

The 802.11 MAC protocol also includes a nifty (but optional) reservation scheme that helps avoid collisions even in the presence of hidden terminals.

1. If initially the station senses the channel idle, it transmits its frame after a short period of time known as the Distributed Inter-frame Space (DIFS)
2. Otherwise, the station chooses a random backoff value using binary exponential backoff and counts down this value after DIFS when the channel is sensed idle. While the channel is sensed busy, the counter value remains frozen.
3. When the counter reaches zero (note that this can only occur while the channel is sensed idle) the station transmits the entire frame and then waits for an acknowledgment.
4. If an acknowledgment is received, the transmitting station knows that its frame has been correctly received at the destination station. If the station has another frame to send, it begins the CSMA/CA protocol at step 2. If the acknowledgment isn't received, the transmitting station reenters the backoff phase in step 2, with the random value chosen from a larger interval

In order to avoid the problem when two wireless hosts are unaware of each other, the IEEE 802.11 protocol allows a station to use a short **Request to Send (RTS)** control frame and a short **Clear to Send (CTS)** control frame to reserve access to the channel.

Although the RTS/CTS exchange can help reduce collisions, it also introduces delay and consumes channel resources. For this reason, the RTS/CTS exchange is only used (if at all) to reserve the channel for the transmission of a long DATA frame.



802.11 Frame

Frame (numbers indicate field length in bytes):

2	2	6	6	6	2	6	0-2312	4
Frame control	Duration	Address 1	Address 2	Address 3	Seq control	Address 4	Payload	CRC

Frame control field expanded (numbers indicate field length in bits):

2	2	4	1	1	1	1	1	1	1	1
Protocol version	Type	Subtype	To AP	From AP	More frag	Retry	Power mgt	More data	WEP	Rsvd

Addresses

1. Address 1 is a MAC address of the wireless station that is to receive the frame.

2. Address 2 is a MAC address of the station that is sending the frame.
3. Address 3 contains MAC address of the router interface that is used to determine the frame's destination.

Bluetooth

Bluetooth networks operate in the unlicensed 2.4 GHz Industrial, Scientific and Medical (ISM) radio band along with other home appliances such as microwaves, garage door openers, and cordless phones. As a result, Bluetooth networks are designed explicitly with noise and interference in mind.

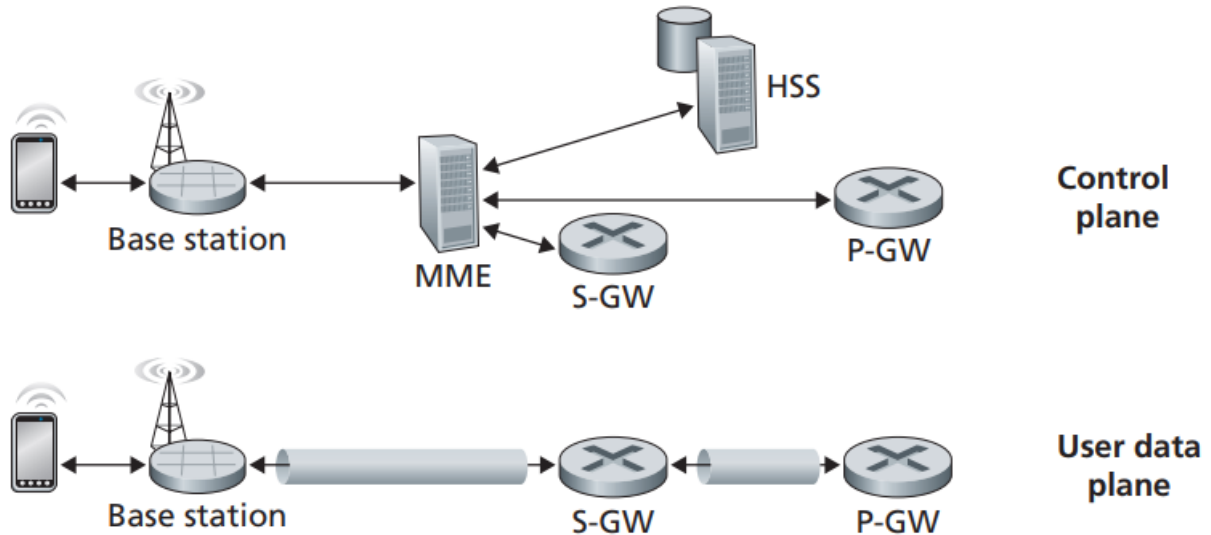
The Bluetooth wireless channel is operated in a TDM manner, with time slots of 625 microseconds. During each time slot, a sender transmits on one of 79 channels, with the channel (frequency) changing in a known but pseudo-random manner from slot to slot.

Cellular Networks

Mobile networks are built in the form of the connected **cells**, each is responsible for a certain area around it.

Each cell contains a base station that transmits signals to, and receives signals from, the mobile devices currently in its cell. The coverage area of a cell depends on many factors, including the transmitting power of the base station, the transmitting power of the devices, obstructing buildings in the cell, and the height and type of the base station antennas

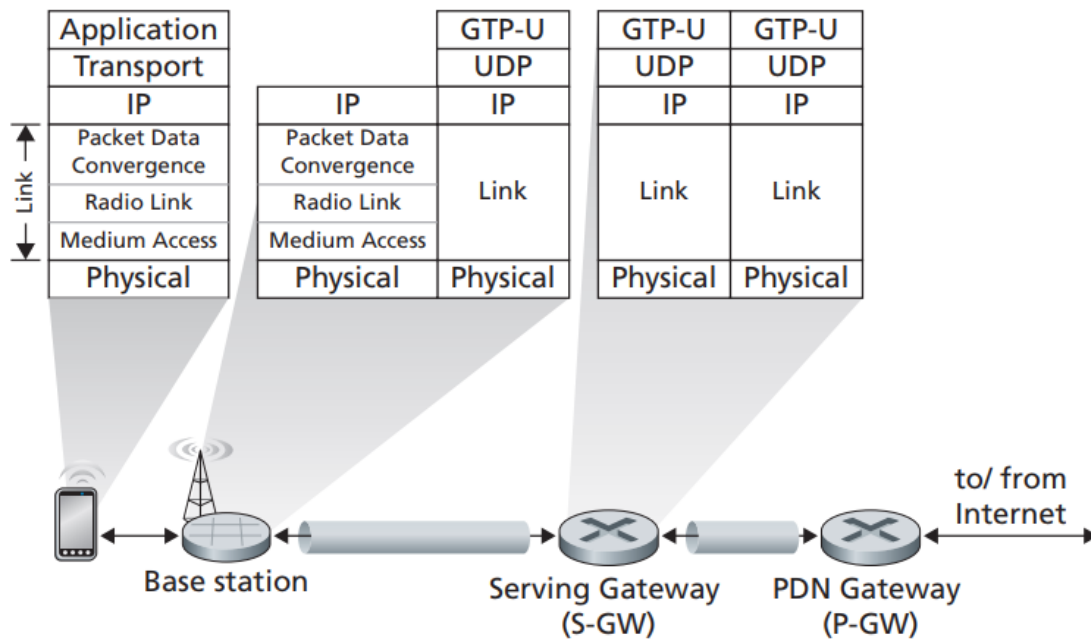
4G LTE



The network elements:

- *Mobile device.* The mobile device also has a globally unique 64-bit identifier called the **International Mobile Subscriber Identity (IMSI)**, which is stored on its **SIM (Subscriber Identity Module)** card.
- *Base station*
- *Home Subscriber Server (HSS).* As shown in Figure 7.18, the HSS is a control-plane element. The HSS is a database, storing information about the mobile devices for which the HSS's network is their home network.
- *Serving Gateway (S-GW), Packet Data Network Gateway (P-GW), and other network routers.*
- *Mobility Management Entity (MME).* The MME is also a control-plane element, serving for a numerous purposes
 - Authentication. It is important for the network and the mobile device attaching to the network to mutually authenticate each other—for the network to know that the attaching device is indeed the device associated with a given IMSI, and for the mobile device to know that the network to which it is attaching is also legitimate cellular carrier network
 - Path setup
 - Cell location tracking. As the device moves between cells, the base stations will update the MME on the device's location.

LTE Protocol Stack



LTE divides the mobile device's link layer into three sublayers:

- *Packet Data Convergence*. This uppermost sublayer of the link layers encrypts and compress datagrams.
- *Radio Link Control* performs two important functions: fragmenting and reliable data transfer
- *Medium Access Protocol* performs error detection and transmission slots requests.

LTE Radio Access Network

LTE uses a combination of frequency division multiplexing and time division multiplexing on the downstream channel, known as orthogonal frequency division multiplexing (OFDM)

Slot (re)allocation among mobile devices can be performed as often as once every millisecond.

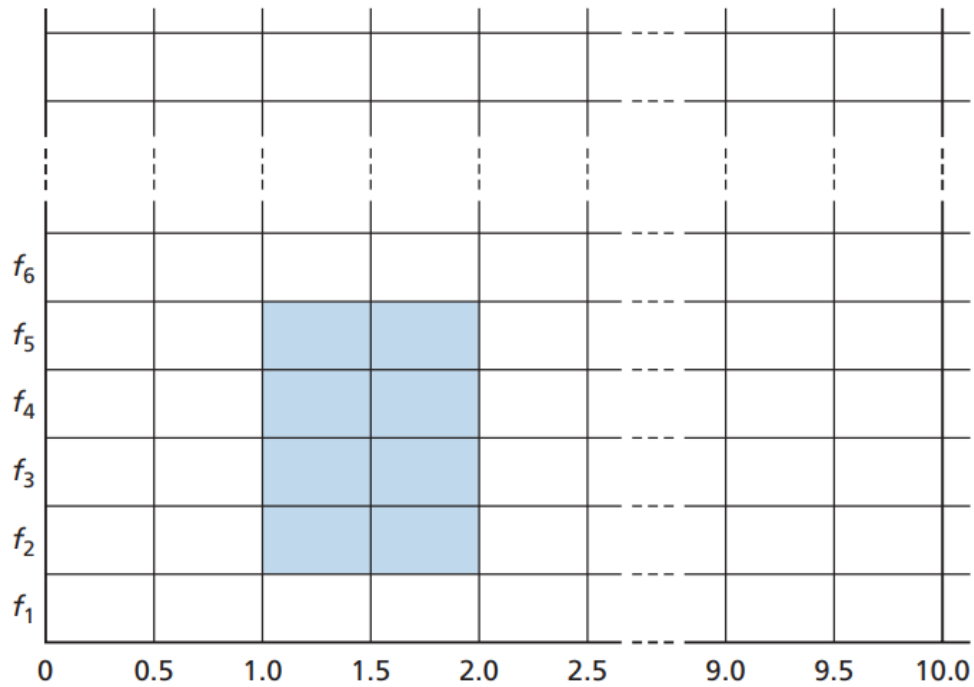
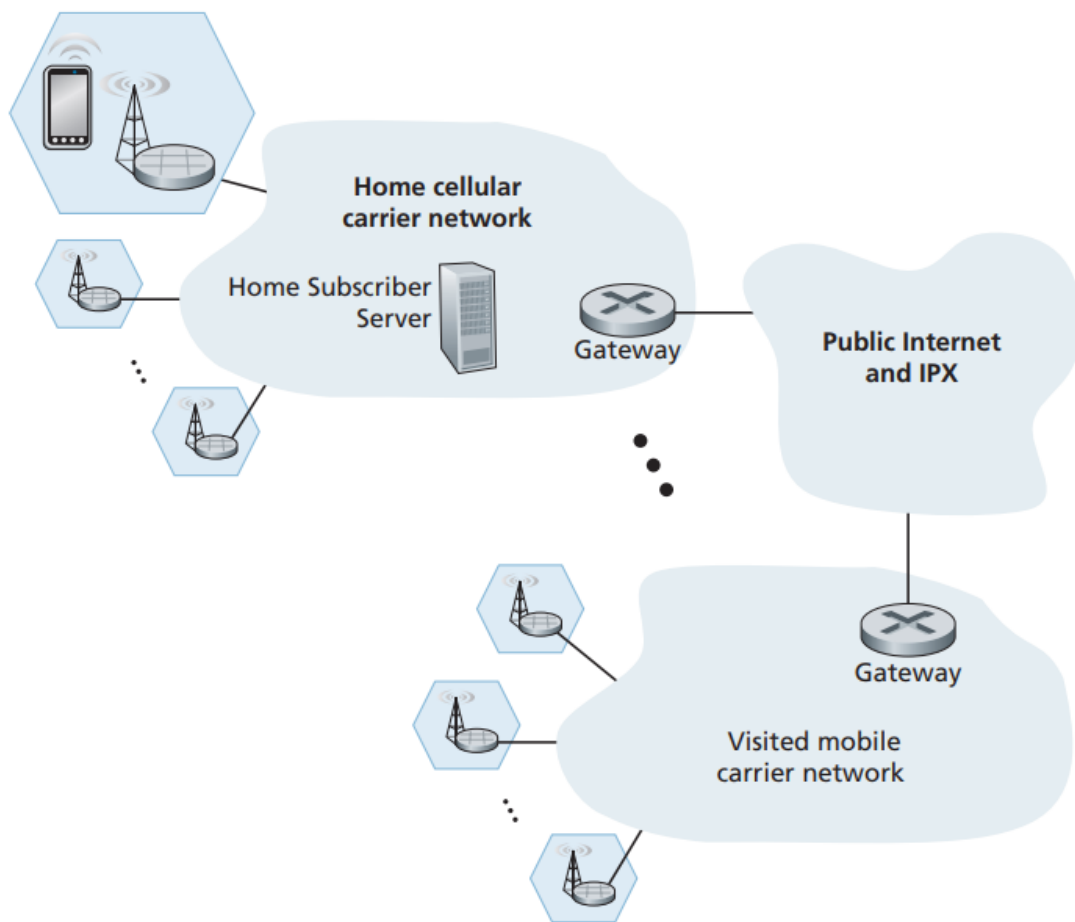


Figure 7.22 ♦ Twenty 0.5-ms slots organized into 10 ms frames at each frequency. An eight-slot allocation is shown shaded.

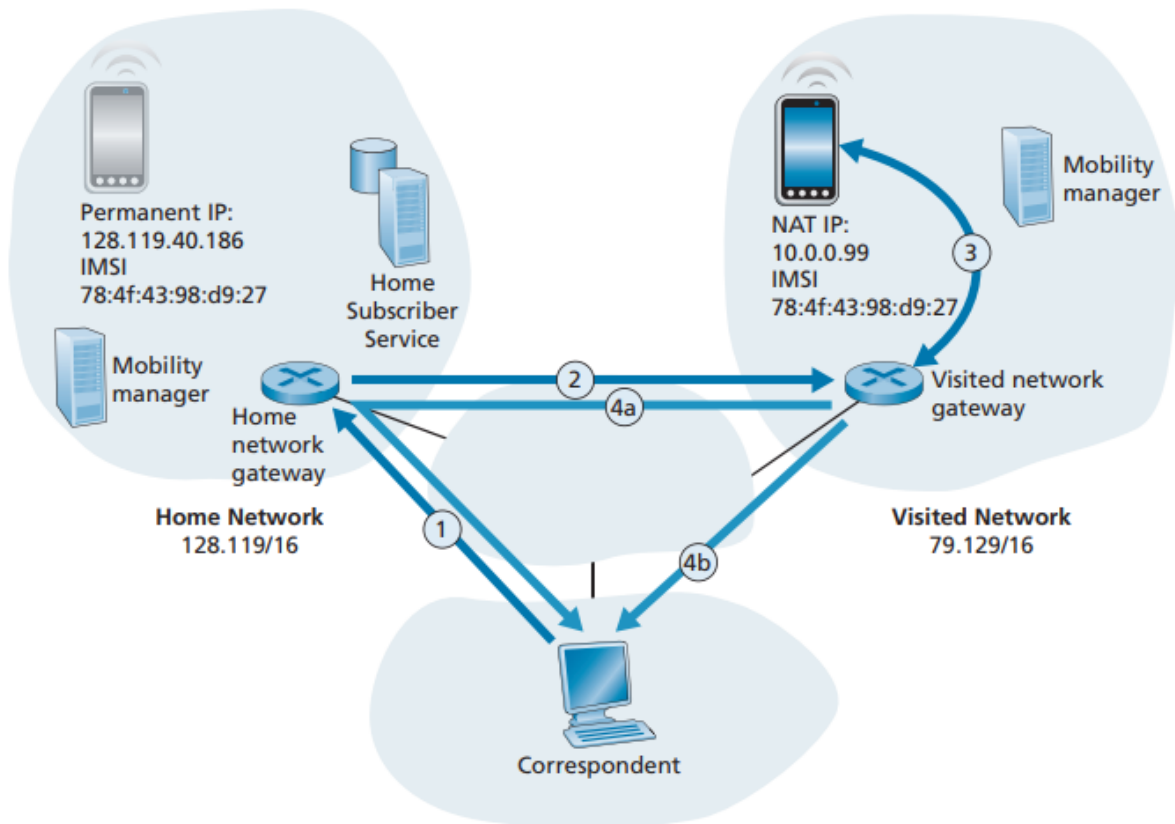
LTE Networks of Networks



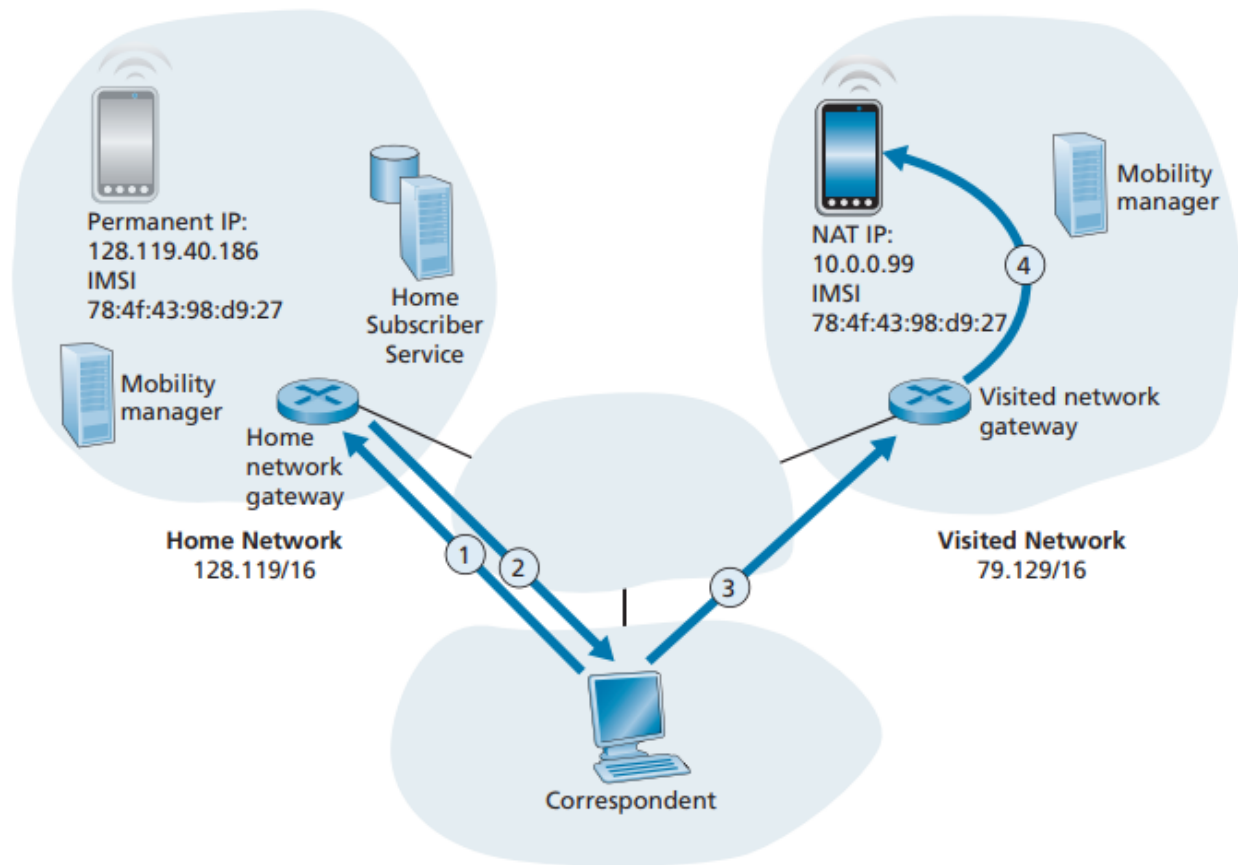
LTE Data-Link Layer

Home Subscriber Service (HSS) stores information about each of its subscribers, including a globally unique device ID (embedded in a subscriber's SIM card), information about services that the subscriber may access, cryptographic keys to be used for communication, and billing/charging information. When a device is connected to cellular network, other than its **home network**, that device is said to be roaming on a **visited network**. When a mobile device attaches to, and roams on, a visited network, coordination will be required between the home network and the visited network.

The notion of a mobile device having a home network provides two important advantages: the home network provides a single location where information about that device can be found, and (as we will see) it can serve as a coordination point for communication to/from a roaming mobile device.



Indirect routing



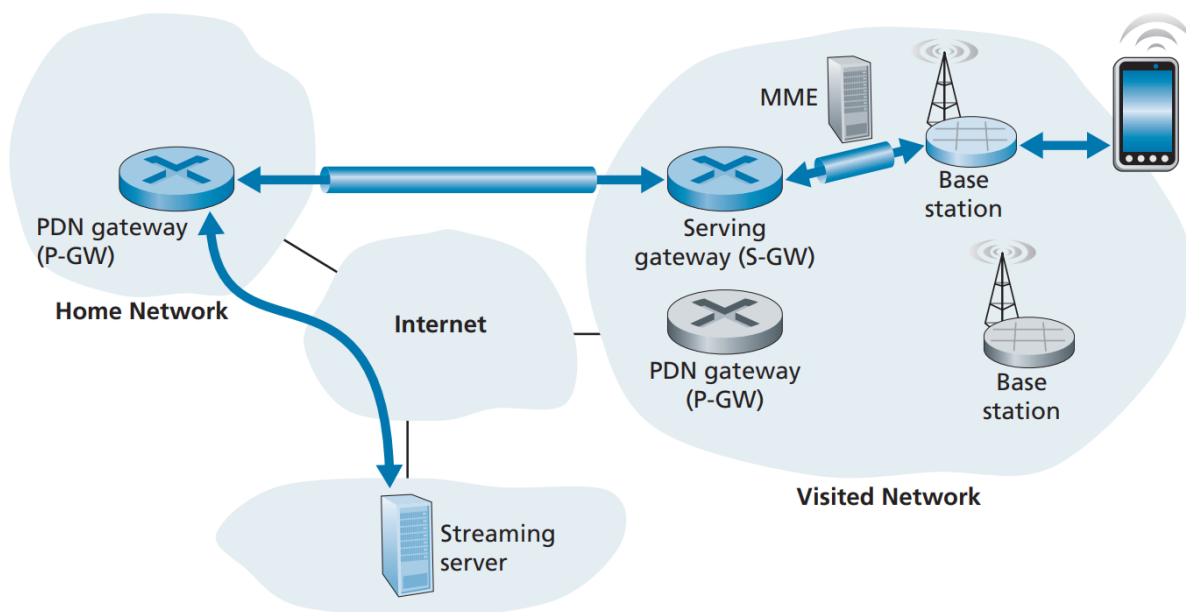
Direct Routing

Mobility Management

Four stages can be distinguished:

1. **Base station association.** The mobile device listens on all frequencies for primary signals being transmitted by base stations in its area. The mobile device acquires progressively more information about these base stations, ultimately selecting the base station with which to associate, and bootstrapping a control-signaling channel with that base station supplying IMSI
2. **Control-plane configuration of LTE network elements for the mobile device.** After connection is established, the MME (Mobile Management Entity) will consult and setup the network elements in both home and visited network. Precisely,

- a. The MME will use IMSI (International Mobile Subscriber Identity) and other information provided by the mobile device to retrieve authentication, encryption, and available network service information
 - b. The MME informs the HSS in the mobile device's home network that the mobile device is now resident in the visited network, and the HSS updates its database
 - c. The base station and the mobile device select parameters for the data-plane channel to be established between the mobile device and the base station
3. **Data-plane configuration of forwarding tunnels for the mobile device.** The MME next configures the data plane for the mobile device. Two tunnels are established. One tunnel is between the base station and a Serving Gateway in the visited network. The second tunnel is between PDN Gateway in home network and Serving Gateway in the visited network.



4G LTE uses indirect routing: all the to/from traffic goes through the mobile's home network. 4G/5G tunnels use the GRPS Tunneling Protocol (GTP)

1. **Handover Management.** In handover process the source base station will constant the target base station with handover request. After getting acknowledgement, the source base will contact the mobile device to reconfigure its station which it should contact and will deliver incoming datagrams to the target base station (4) instead. After informing the MME about changes, target base sends a message to source

one to free resources allocated to mobile phone (6) and the new communication channel (7) is finally set up.

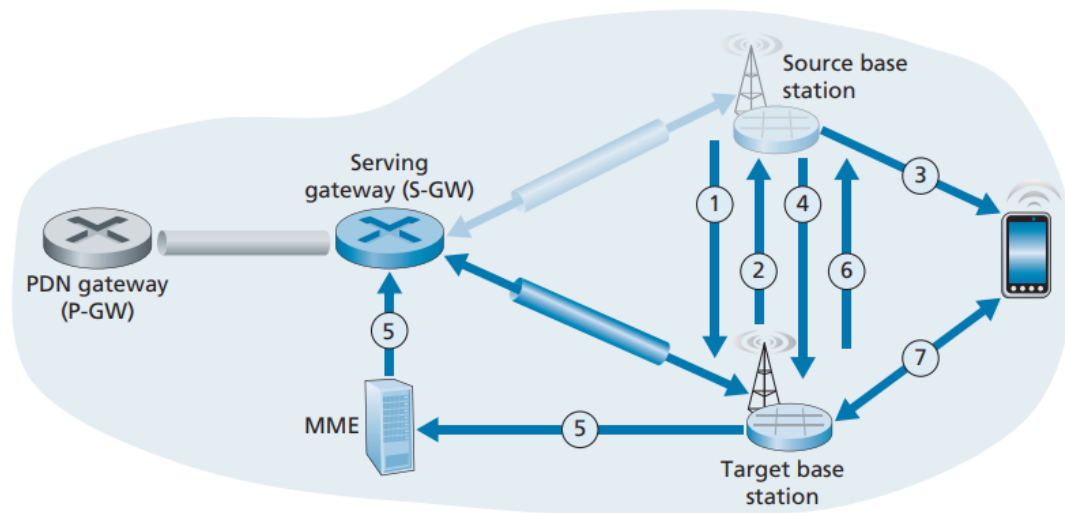


Figure 7.30 ♦ Steps in handing over a mobile device from the source base station to the target base station

Mobile IP

Mobile IP is Internet protocol served for mobile hosts, but was not quite spread as 4G/5G. Much of the notions in mobile IP are the same.

It consists of three parts:

1. *Agent discovery.* Mobile IP defines the protocols used by a foreign agent to advertise its mobility services to a mobile device that wishes to attach to its network. Those services will include providing a **care-of-address** to the mobile device for use in the foreign network, registration the mobile host with the home agent in the mobile device's home network, and forwarding of datagrams to/from mobile device, among other devices
2. *Registration with the home agent.* Mobile IP defines the protocols used by the mobile device and/or foreign agent to register and deregister a care-of-address with a mobile device's home agent.
3. *Indirect routing of datagrams*