Computer Networking Cheatsheet

Matteo Alberici

Computer Networking 2021
Computer Science
USI - Università della Svizzera Italiana, Lugano

Contents

1 Computer Networks and the Internet

1.1 What Is the Internet?

1.1.1 A Nuts-and-Bolts Description

A **computer network** consists of a number of interconnected machines. The Internet is a compound of interconnected networks with billions of devices called **hosts** or **end systems** throughout the world. The **Internet of things (IoT)** describes the network of objects embedded with technologies to connect and exchange data.
Hosts are connected together by **communication links** and **packet switches**. Different links can transmit at different **transmission rates** [bps].
An host having data to send segments it and adds header bytes, creating **packets**. Each packet is forwarded to a destination host by a packet switch, which can be either a **router**, for the network core, or a **link-layer switch**, for access networks. The sequence of links and switches traversed is called a **route** or **path** through the network.
Internet components run **protocols** that control the sending and receiving of information.

1.1.2 A Services Description

Internet applications run on hosts and are said to be **distributed applications** since they involve multiple hosts exchanging data.

1.1.3 What Is a Protocol?

A protocol defines the **format** and the **order** of messages exchanged between communicating devices, as well as the actions taken on events while transmitting.

1.2 The Network Edge

The Internet hosts constitute the **network edge** and are divided into two categories: **clients**, such as desktops and smartphones, and **servers**, which reside in large **data centers**.

1.2.1 Access Networks

Access Networks connect an host to an **edge router**.

1.2.2 Physical Media

A **physical link** lies between transmitter/receiver pairs.
Hosts transmit at full link capacity a message of **L bits** over a link at a **rate R** generating a **transmission delay** d_{trans}.
Physical media can be either **guided**, if the signal propagates in a solid medium, or **unguided**, if it propagates freely.

1.3 The Network Core

The **network core** consists of a mesh of interconnected routers.

1.3.1 Packet Switching

Store-and-Forward Transmission

Most switches use **store-and-forward transmission**, meaning the switch receives the entire packet before it begins to transmit, generating an **end-to-end delay** $d_{end-to-end}$.

Queuing Delays and Packet Loss

If a link is **busy**, the arriving packets wait in the link **output buffer**, generating a **queuing delay** d_{queue}. If the buffer is full, then **packet loss** occurs and either the arriving packet or one among the queued ones is dropped.

Forwarding Tables and Routing Protocols

Every host has an IP address and when it sends a packet, it includes the destination's IP address in the packet's header. Every router has a **forwarding table** that maps destination addresses to the router's outbound links: when a packet arrives, the router searches the forwarding table in order to find the appropriate link and forwards the packet. There exist special **routing protocols** which determine the shortest path from each router and use the results to configure the forwarding tables.

1.3.2 Circuit Switching

In **circuit-switched networks**, the resources needed along a path are reserved for the duration of the communication between the hosts. The switches on the path maintain the connection state establishing a **circuit**.

Multiplexing in Circuit-Switched Networks

A circuit is implemented with either **frequency-division multiplexing (FDM)** or **time-division multiplexing (TDM)**. In FDM, the link dedicates a frequency band of a certain **bandwidth** to each connection for its entire duration. In TDM, the link divides time into fixed frames, divided again into slots, and when the connection is established, the network dedicates one slot per frame to the connection.

1.3.3 A Network of Networks

The Internet is built as a network of networks:

- **Network Structure 1**: interconnects all the **Internet Service Providers (ISPs)** with a single global transit ISP, having the access ISPs as **customers** and the global one as **provider**
- **Network Structure 2**: two-tier hierarchy with providers at the top tier and clients at the bottom tier
- **Network Structure 3**: clients connect to regional ISPs which then connect to **tier-1 ISPs**
- **Network Structure 4**: routers at the same location are grouped by **PoPs (Points of Presence)** in order to connect to providers, which could connect more providers (**multi-home**); a third-party company can create an **IXP (Internet Exchange Point)** where providers can peer together
- **Network Structure 5**: **content-provider networks** peer with clients in order to connect to tier-1 ISPs.

1.4 Delay, Loss, and Throughput in Packet-Switched Networks

1.4.1 Overview of Delay in Packet-Switched Networks

As a packet travels from an host to the subsequent one along a path, it suffers from several delays at each host.

Types of Delay

- **Processing Delay** d_{proc}: time required to examine the packet's header and determine where to direct it
- **Queuing Delay** d_{queue}: time spent by a packet waiting to be transmitted
- **Transmission Delay** d_{trans}: time required to transmit all of a packet's bits
- **Propagation Delay** d_{prop}: time required to propagate from the beginning of the link to the next router
- **Total Delay** d_{total}: sum of all the delays in a specific host

Traffic can be either **periodic**, if every packet finds an empty queue, or **bursty**, if N packets arrive every $(L/R)N$ seconds, but typically, the arrival process is **random**.

1.4.2 Queuing Delay and Packet Loss

1.4.3 End-to-End Delay

Traceroute

1.4.4 Throughput in Computer Networks

The **end-to-end throughput** is the instantaneous rate at which an host receives data from another host. It corresponds to the transmission rate of the **bottleneck link**. Today, the throughput constraining factor is typically the access network.

1.5 Protocol Layers and Their Service Models

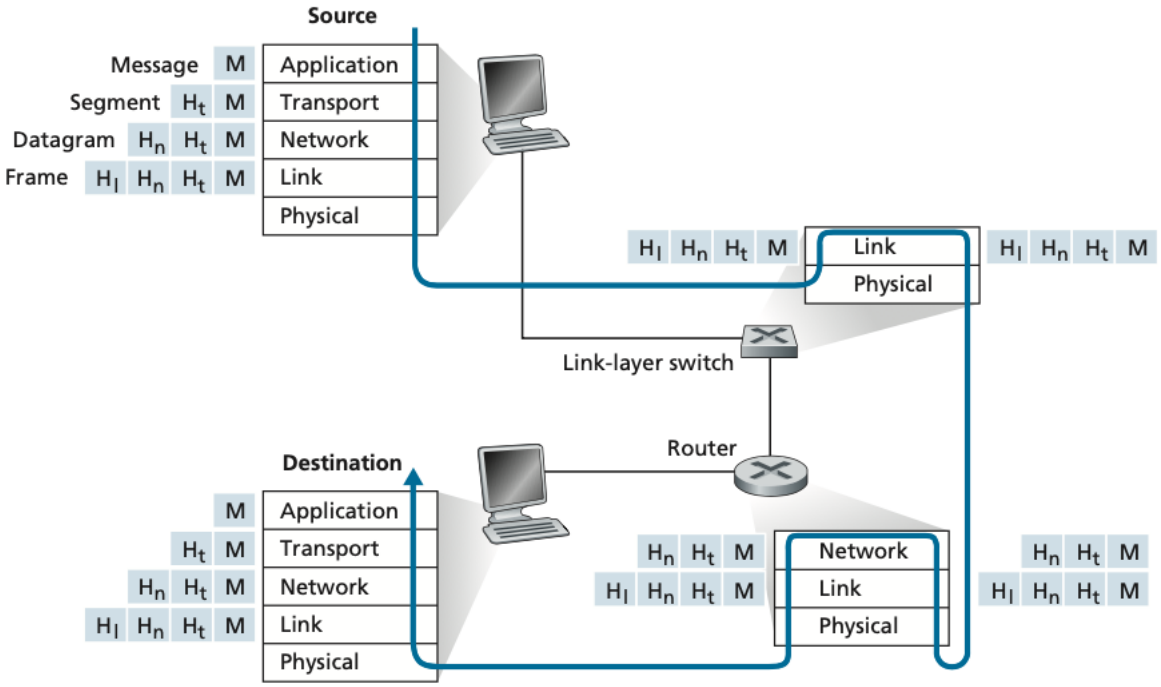
1.5.1 Layered Architecture

To provide structure to network protocols, designers organize them in **layers**, each providing specific **services** used by the upper layers.

1.5.2 Protocol Layering

All the protocols from the different layers taken together are called the **protocol stack**. The Internet protocol stack consists of five layers:

- **Application Layer**: where network applications reside; its protocols are distributed over hosts and applications use them to exchange **messages** with other hosts
- **Transport Layer**: passes application layer messages known as **segments** with a destination address to the network layer
- **Network Layer**: moves packets known as **datagrams** from an host to another
- **Link Layer**: at each node, it receives the network layer datagram known as **frame** and delivers it to the next node, where it is passed up to the network layer
- **Physical Layer**: moves the **bits** of a frame from one node to another



1.6 Encapsulation

Each layer adds some header information to the payload received in the packet **encapsulating** it.

2 Application Layer

2.1 Principles of Network Applications

Creating a network application means writing programs that run on hosts and communicate over the network.

2.1.1 Network Application Architectures

The application architecture is designed by developers and dictates how the application is structured over the various hosts.
In a **client-server architecture**, there is an always-on host called the **server** which services requests from many **clients** that do not communicate with each other. Since a server may be incapable of keeping up with all the requests, **data centers** are used to create virtual and powerful servers.
In a **P2P architecture**, application exploits direct communication between pairs of connected hosts called **peers**. One important feature is their self-scalability, meaning that they request and provide services to other peers. This architecture is cost effective, but faces challenges of security and reliability due to the decentralized structure.

2.1.2 Processes Communication

A **process** is a program running within an host. Within the same host, processes communicate through **inter-process communication**, while in different hosts they communicate by exchanging **messages**.

Client and Server Process

For each pair of communicating processes, the **client process** is the one that initiates communication, while the **server process** is the one that waits to be contacted.

The Interface Between the Process and the Computer Network

A process exchanges messages through a software interface called a **socket**, which is the interface between the application layer and the transport layer within an host, also referred to as the **Application Programming Interface (API)** between the application and the network.

Addressing Processes

If a process wants to send a message, it has to know the destination **IP address** and a **port number** specifying the receiving process.

2.1.3 Transport Services Available to Applications

Reliable Data Transfer

When a transport protocol provides **reliable data transfer**, the sending process knows that the data sent will arrive with no errors nor loss.

Troughput

Bandwidth-sensitive applications need a **guaranteed available throughput** at a specified rate, while **elastic applications** make use of whatever throughput they get.

Timing

As with throughput, a transport protocol may need to provide **timing** guarantees in order to avoid long delays.

Security

Transport protocols can provide many **security** services, such as encryption and data integrity.

2.1.4 Transport Services Provided by the Internet

There exist two Internet **transport protocols**: the **Transmission Control Protocol (TCP)** and the **User Datagram Protocol (UDP)**.

TCP Services

- **Connection-oriented service**: first handshaking to create a **TCP connection** before exchanging messages
- **Reliable data transfer**: bytes stream sent is received with no errors nor loss
- **Flow control**: sender process does not overwhelm the receiver
- **Congestion control**: sender is throttled if the network is overloaded
- **NOT provided**: timing, throughput guarantees, and security services

UDP Services

UDP is a no-frills, connectionless protocol, meaning there is no handshake. It does not provide reliability, flow and congestion control, timing and throughput guarantees, nor security services.

2.1.5 Application-Layer Protocols

An **application-layer protocol** defines how processes on different hosts communicate:

- **Type**: request or response
- **Syntax**: what fields are in the messages and how they are delineated
- **Semantics**: meaning of the information in the fields
- **Rules**: when and how processes send and respond

2.2 The Web and HTTP

2.2.1 Overview of HTTP

The **HyperText Transfer Protocol (HTTP)** is at the heart of the web and is implemented both in a client side (**Web browsers**) and in a server side (**Web servers**). A **Web page** consists of a **base HTML file** and several referenced **objects**, each addressable by a **Uniform Resource Locator (URL)**.
HTTP defines how clients request web pages and how servers transfer pages to them:

1. client initiates a TCP connection to server, port 80
2. server accepts the connection
3. HTTP messages exchanged between the browser and the server
4. TCP connection is closed

HTTP is a **stateless protocol** since it maintains no information about the clients.

2.2.2 Non-Persistent and Persistent Connections

HTTP with Non-Persistent Connections

In **non-persistent HTTP**, only one object is sent over each TCP connection; thus, downloading multiple objects requires multiple connections. The **round-trip time (RTT)** is the time that elapses from the client request until the file is received: since there is a three-way handshake before the object is sent, the total time spent is two RTTs plus the the object transmission time.

HTTP with Persistent Connections

In **persistent HTTP**, multiple object are sent over a single TCP connection, thus only one RTT is spent to all the referenced objects.

2.2.3 HTTP Message Format

HTTP Request Message

```
GET /somedir/page.html HTTP/1.1
Host: www.someschool.edu
Connection: close
User-agent: Mozilla/5.0
Accept-language: fr
```

The first line is called the **request line** and has three fields: the method, the URL, and the version. Some common methods are *GET*, *POST*, *PUT*, *DELETE*, and *HEAD*.
The subsequent lines are called the **header lines**, followed by a blank line and by the **entity body**.

HTTP Response Message

```
HTTP/1.1 200 OK
Connection: close
Date: Tue, 18 Aug 2015 15:44:04 GMT
Server: Apache/2.2.3 (CentOS)
Last-Modified: Tue, 18 Aug 2015 15:11:03 GMT
Content-Length: 6821
Content-Type: text/html
(data data data data data ...)
```

The first line is called the **status line** and has three fields: the protocol version, a status code, and a status message. Some common status code are:

- **200 OK**: request succeeded
- **301 Moved Permanently**: requested object has a new URL specified in the *Location* header
- **400 Bad Request**: request not understood
- **404 Not Found**: requested document does not exist in the server
- **505 HTTP Version Not Supported**: HTTP version not supported by the server

The subsequent lines are **header lines**, and then there is the **entity body**.

2.2.4 User-Server Interaction: Cookies

It is often desirable for a Web server to identify users to restrict user access or to serve content as identity functions: **Cookies** allow sites to keep track of users. Cookies technology has four components: a header line in the HTTP response, a header line in the HTTP request, a file on the user's host managed by the browser, and a back-end database at the Web site.

2.2.5 Web Caching

A **Web cache**, also called a **proxy server**, is a network entity that satisfies HTTP requests on the behalf of an origin Web server. It stores copies of the recently requested objects:

1. browser establishes a TCP connection to the Web cache and requests the object
2. if the Web cache finds it, the object is returned within a response
3. otherwise, the Web cache establishes a TCP connection to the origin server and requests the object
4. when the object is returned, the Web cache stores a copy and sends it to the client within a response

Web caching acts as both client and server at the same time and reduces response times and network traffic.

The Conditional GET

An object housed in a Web server could have been modified since it was cached: the **conditional *GET*** mechanism verifies this adding the **If-Modified-Since** header line in the request to the Web server, which returns the object iff it is no longer the same stored in the cache.

2.3 Electronic Mail in the Internet

The Internet mail system has three major components: **user agents**, **mail servers**, and the **Simple Mail Transfer Protocol (SMTP)**, which uses TCP to transfer emails from clients to servers.

2.3.1 SMTP

The journey of a message from the sender to receiver is:

1. sender invokes its user agent providing the receiver email address and instructs it to send a message

2. sender's user agent sends the message to the sender's mail server in the **message queue**
3. SMTP client side opens a TCP connection to an SMTP server on the receiver mail server
4. the message is sent into the connection to the receiver mail server
5. SMTP server side receives the message and the receiver mail server places it in the receiver **mailbox**
6. receiver invokes its user agent to read the message

There exist many SMTP commands:

- **HELO**: starts the conversation identifying the sender server
- **EHLO**: starts the conversation with the server using Extended SMTP
- **MAIL FROM**: sender states the mail address in the *from* field
- **RCPT TO**: identifies the receiver
- **SIZE**: informs the receiver server about the mail size
- **DATA**: email content starts to be transferred
- **VRFY**: server verifies if an address exists
- **TURN**: inverts roles between client and server
- **AUTH**: client authenticates to the server
- **RSET**: alerts the server that the transmission is terminating
- **EXPN**: asks for a confirmation about the identification of a mailing list
- **HELP**: client requests useful information
- **QUIT**: terminates the conversation

Along with the commands, there exist some reply codes fore SMTP:

- $2.y.z$: command sent and completed successfully
- $3.y.z$: command accepted but not completed due to missing information
- $4.y.z$: command not accepted due to a temporary problem (such as low storage space)
- $5.y.z$: command not accepted due to an unrecoverable reason (such as a non-existing address)

2.3.2 Mail Message Format

From: alice@crepes.fr
To: bob@hamburger.edu
Subject: Searching for the meaning of life.

A mail message consists of some header lines, a blank line, and the message body.

2.3.3 Mail Access Protocol

There are two common ways to retrieve a message in the mail server: if the receiver is using a Web-based or smartphone app, then the user agent will use HTTP; otherwise, the protocol used is **Internet Mail Access Protocol (IMAP)**.

2.4 DNS - The Internet Directory Service

2.4.1 Services Provided by DNS

Internet's **domain name system (DNS)** is a directory service that translates hostnames into IP addresses. It is a distributed database implemented in a hierarchy of **DNS servers** and an application-layer protocol running on UDP that allow hosts to query the database.
If an host wants to send an HTTP request, it needs to know the Web server's IP address:

1. user machine runs the DNS application client side
2. browser extracts the hostname by the URL and passes it to the DNS client side
3. DNS client sends a query with the hostname to a DNS server
4. DNS server eventually replies with the IP address
5. browser receives the IP address and opens a TCP connection

DNS provide a few other services:

- **Host aliasing**: DNS can retrieve an host's **canonical name** given one of its alias names
- **Mail server aliasing**: DNS can retrieve a mail server's canonical name by one of its alias names
- **Load distribution**: busy sites are replicated over multiple servers with a set of IP addresses associated that rotate to distribute the traffic

2.4.2 Overview of How DNS Works

A centralized design with a single DNS server does not scale:

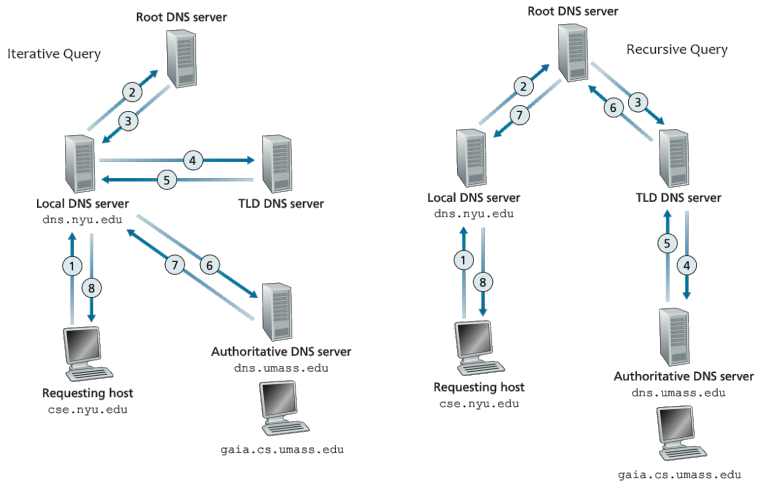
- if the server crashes, then Internet does the same
- the server should handle all DNS queries
- the server could not be close to all the querying clients
- the server should keep records for all the Internet hosts

A Distributed, Hierarchical Database

DNS uses several servers in hierarchical structure distributed around the world:

- **Root DNS server**: copies of 13 different servers; they provide TLD servers IP addresses
- **Top-level domain (TLD) servers**: there exists a TLD server for each TLD domain; they provide the authoritative DNS servers IP addresses
- **Authoritative DNS servers**: an organization with publicly accessible hosts provides accessible DNS records mapping the hostnames to IP addresses, housed in authoritative DNS servers

The **local DNS server** does not belong to the hierarchy and forwards the DNS queries from the hosts that are closed to it.



Queries can be described in two categories: the **iterative queries** return either the response or returns the server to ask, while the **recursive queries** put burden of name resolution on the contacted name server.

DNS Caching

In a query chain, a DNS server can store a hostname/IP address reply in its local memory. After some **time to live (TTL)**, cache entries timeout and disappear.

2.4.3 DNS Records and Messages

DNS servers store **resource records (RRs)**, which are four-tuples with the format:
(Name, Value, Type, TTL) .
The meaning of Name and Value depend on Type:

- Type A: Name is a hostname and Value is the IP address of the hostname
- Type NS: Name is a domain and Value is the hostname of an authoritative server which can retrieve the IP address for hosts in the domain
- Type CNAME: Name is an alias hostname and Value is the canonical name for the alias name
- Type MX: Name is mail server alias name and Value is the canonical name for the alias name

DNS Message

The semantics of the DNS message fields are:

- **Header section**, 12 bytes:
  - 16 bits: query identifier
  - 1 bit: indicates if the message is a query (0) or a reply (1)
  - 1 bit: authoritative flag
  - 1 bit: recursion-desired flag
  - 1 bit: recursion-available flag
- **Question section**:
  - Name field
  - Type field
- **Answer section**: resource records for the queried name
- **Authority section**: authoritative servers records
- **Additional section**: other helpful records

2.5 Peer-to-Peer File Distribution

2.5.1 Scalability of P2P Architectures

The **distribution time** D_{cs} is the time spent to get a copy of a file distributed among some peers.

BitTorrent

BitTorrent (BT) is a communication protocol for P2P file sharing which enables users to distribute data in a decentralized manner. It is used to transfer large digital video or audio files with BitTorrent **clients**. **Trackers** provide a list of available files and peers, known as **seeds**.
Peers in a torrent download equal-size chunks from one another and upload chunks to them. Each peer requires the missing chunks in a **rarest first** manner and sends chunks to the others that are sending it chunks at highest rate. At some time, a peer is **optimistically unchoked**, meaning that it is selected randomly by a peer and starts to receive chunks, maybe finding a new top-provider.

2.6 Video Streaming and Content Distribution Networks

Videos are sequence of images compressed to a bit-rate: the higher it is, the better the image quality.

2.6.1 Internet Video

Prerecorded videos are stored on servers and users send requests to view them **on demand**.
There are two methods to encode videos:

- **Spacial coding**: instead of sending N values of the same type, the values sent are the type and the number of occurrences
- **Temporal coding**: instead of sending the frame $i + 1$, only differences from frame i are sent

Videos can be encoded at a **Constant Bit Rate (CBR)**, with a fixed rate, or at a **Variable Bit Rate (VBR)**, with a rate changing depending on space and time.

2.6.2 HTTP Streaming and DASH

In **Dynamic Adaptive Streaming over HTTP (DASH)**, videos are encoded into several versions each with a different bit rate and URL provided by a **manifest file**. Clients measure server-to-client bandwidth and determines when to request a chunk, what rate to request, and where to request the chunk.

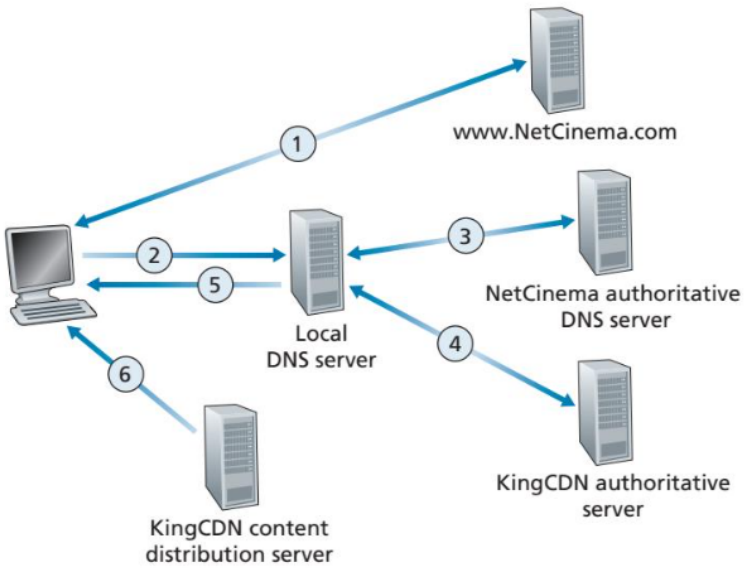
2.6.3 Content Distribution Networks

Content Distribution Networks (CDNs) are use to distribute massive amounts of video data to users around the world. It may be a **private CDN**, owned by by the provider, or a **third-party CDN**, distributing on behalf of multiple providers.
Two server placement strategies can be adopted:

- **Enter deep**: pushes CDN servers deep into many access networks, being closer to users
- **Bring home**: less larger clusters in Internet Exchange Points (IXPs) and Points of Presence (POP) near access networks

CDN Operation

1. user visits a Web page and clicks a link
2. user's host sends a DNS query
3. user's Local DNS Server (LDNS) relays the query to an authoritative DNS server which returns a hostname in the domain
4. query enters the domain private DNS infrastructure and a second query to retrieve the content server IP address is sent
5. LDNS forwards the IP address to the user's host
6. client establishes a TCP connection with the server and issues an HTTP request for the video



2.6.4 Cluster Selection Strategies

A **cluster selection strategy** is a mechanism for directing clients to a server cluster within the CDN. After learning an IP address, the CDN needs to select a cluster and employs proprietary cluster selection strategies such as assigning the client to the closest cluster or determining the best one based on the traffic conditions.

3 Transport Layer

3.1 Introduction and Transport-Layer Services

A transport-layer protocol provides **logical communication** between application processes on different hosts: the sender breaks messages into segments and passes them to the network layer, while the receiver reassembles segments into messages and passes them to the application layer.

3.1.1 Relationship Between Transport and Network Layers

The main difference is that in network-layer the **Internet Protocol (IP)** provides a best-effort delivery service (no guarantees) in logical communication between hosts instead of processes.

3.1.2 Overview of the Transport Layer in the Internet

Both TCP and UDP provide integrity checking with error detection and multiplexing, but neither of them provide delay nor bandwidth guarantees.

3.2 Multiplexing and Demultiplexing

Extending the host-to-host delivery to process-to-process delivery is fundamental for all computer networks.
The source hosts gather data from different sockets and encapsulates it with header information creating segments for the network layer in a process called **multiplexing**. The receiving host transport layer examines segments fields to identify the correct receiving socket in a process called **demultiplexing**.
Every segment has a **source port number field** indicating the source application and a **destination port number field** specifying the appropriate receiving socket.

Connectionless Multiplexing and Demultiplexing

When creating a UDP socket segment, the sender specifies the destination IP address and the destination port number; then, the receiver checks the port number and directs the segment to the corresponding socket.

Connection-Oriented Multiplexing and Demultiplexing

A TCP socket is identified by a 4-tuple:

- source IP address
- source port number
- destination IP address
- destination port number

The receiver uses all the 4 values to direct the segment.
A server may support many simultaneous TCP sockets.

3.3 Connectionless Transport: UDP

Some applications are better suited for UDP:

- Finer application-level control on what and when data is sent, thus packages and passes data immediately
- No connection establishment, thus does not introduce delay
- No connection state, thus more activities at the same time
- Small packet header overhead (8 bytes)

3.3.1 UDP Segment Structure

The UDP header has four 2-bit fields: the source and destination port numbers, the segment length, and the **checksum** for error detection.

3.3.2 UDP Checksum

The checksum determines whether segment bits have been altered: the UDP sender side performs 1s complement on the sum of all the 16 bits words in the segment (overflow is wrapped around); at the receiver, all 16 bits words are added, including the checksum and, if a bit is a 0, then errors occurred.
If the segment is altered such that the sum remains constant, the checksum will not detect the error.

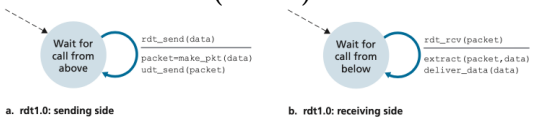
3.4 Principles of Reliable Data Transfer

The problem of a **reliable data transfer protocol** is that the layers below it may be unreliable. The sending side of the protocol is invoked by `rdt_send()` and with `deliver_data()` it passes the data to the upper layer at the receiving side, which calls `rdt_rcv()` when a packet arrives on the receiving channel.
Both the sender and receiver sides may send segments with an **unreliable data transfer protocol** by calling `udt_send()`.

3.4.1 Building a Reliable Data Transfer Protocol

Reliable Data Transfer over a Perfectly Reliable Channel: rdt1.0

In the protocol **rdt 1.0**, the **finite state machines (FSMs)** define one state for both the sender and the receiver.

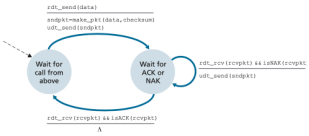


a. rdt1.0: sending side b. rdt1.0: receiving side

Reliable Data Transfer over a Channel with Bit Errors: rdt2.0

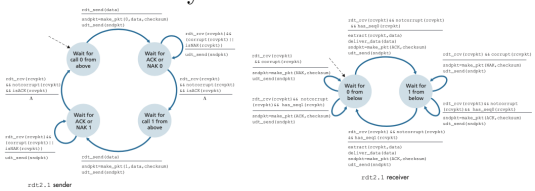
Protocol **rtd 2.0** shows a more realistic model based on an **Automatic Repeat reQuest (ARQ)** protocol:

- **Error detection**: errors are detected and eventually corrected through the checksum field
- **Receiver feedback**: receiver sends back positive (**ACK**) or negative (**NAK**) acknowledgment packets
- **Retransmission**: packets received in error are sent again



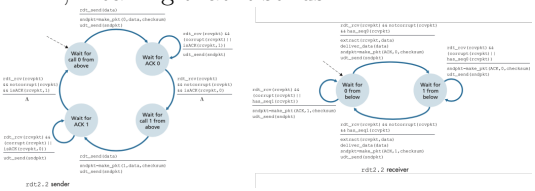
a. rdt2.0: sending side b. rdt2.0: receiving side

Protocol rdt 2.0 is a **stop-and-wait** protocol since the sender does not send new data until it receives an ACK packet. If the acknowledgment packet is corrupted, then the sender sends the packet again until a correct ACK is received, introducing duplicate packets into the channel. The solution is to add a field for a **sequence number** allowing the receiver to determine whether the packet has already been received or not. Protocol **rtd 2.1** handles this problem:



rdt2.1 sender rdt2.1 receiver

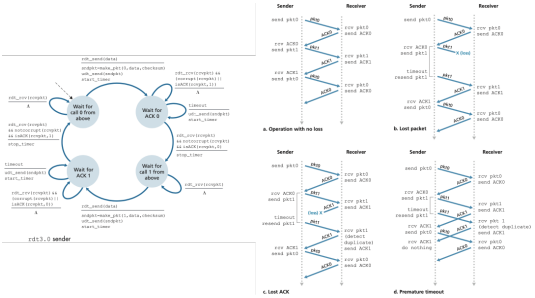
Protocol **rtd 2.2** sends an ACK packet instead of a NAK, meaning that it sends the same ACK packet twice for a duplicated packet:



rdt2.2 sender rdt2.2 receiver

Reliable Data Transfer over a Channel with Bit Errors: rdt3.0

In protocol **rdt 3.0**, the underlying channel can lose packets; thus, a **countdown timer** is started each time a packet is sent and, if after some time the ACK packet has not been received, then the sender retransmits the packet:



3.4.2 Pipelined Reliable Data Transfer Protocols

The problem of rdt3.0's performance is that it is a stop-and-wait protocol: the sender **utilization** U_{sender} is very low compared with the transmitting time. The solution is **pipelining**, which is sending multiple packets before having to wait for acknowledgments.

3.4.3 Go-Back-N (GBN)

In a **Go-Back-N (GBN)** protocol, the sender sends multiple packets without acknowledgment, but with a maximum of N unacknowledged packets in the pipeline.
Given that the **base** is the oldest acknowledged sequence number and that **nextseqnum** is the smallest unused sequence number:

- [0, base -1] : packets already transmitted and acknowledged
- [base, nextseqnum -1] : packets transmitted but not acknowledged
- [nextseqnum, base $+ N - 1$] : packets available to send
- $\geq$ base $+ N$] : packets unavailable at the time

Since a single packet error could cause the retransmission of many packets, the **selective repeat** method is used to allow the receiver to individually acknowledge specific packets.

3.5 Connection-Oriented Transport: TCP

3.5.1 The TCP Connection

TCP is a **connection-oriented** protocol since before sending data, processes must handshake with each other. It provides a **full-duplex** service, meaning that data from host A to host B flows as data from B to A, and is **point-to-point**, meaning that it is between a single pair sender/receiver.
Since three segments are sent between the hosts, the connection-establishment is a **three-way handshake**. Once the connection is established and the data has been sent, TCP directs it to the **send buffer**, from where it will be grabbed

and passed to the network layer. The amount of data that can be grabbed an placed in a segment is limited by the **maximum segment size (MSS)**, which is based on the largest frame possible, called the **maximum transmission unit (MTU)**.

3.5.2 TCP Segment Structure

A TCP segment consists of data fields, containing a data chunk, and header fields, such as port numbers, checksum, and more:

- **Sequence number**, 32 bits
- **Acknowledgment number**, 34 bits
- **Receive window**, 16 bits
- **Length**, 4 bits
- **Options**: used when sender and receiver negotiate the MSS
- **Flag**, 6 bits

Sequence Numbers and Acknowledgment Numbers

The sequence number for a segment is the **byte-stream number** of the **first byte** in the segment, while the acknowledgment number is the sequence number of the next byte the host is expecting.
TCP provides **cumulative acknowledgment**, meaning that it acknowledges bytes up to the first missing byte in the stream.

3.5.3 Round-Trip Time Estimation and Timeout

Estimating the Round-Trip Time

The **SampleRTT** is the amount of time between when a segment is sent and its acknowledgment. It is measured at some point in time on a single-transmission segment. TCP maintains a sample RTT average value called **EstimatedRTT**. The new value of `EstimatedRTT` is a weighted combination of its previous value and the new `SampleRTT`. It is an **exponential weighted moving average (EWMA)**, meaning an average that puts more weight on recent samples.
It is valuable to have a measure of the **DevRTT**, which is an EWMA indicating the RTT variation.

Setting and Managing the Retransmission Timeout Interval

The **timeout interval** should be greater than the `EstimatedRTT`, or unnecessary retransmissions would be sent, but not too much larger, or TCP would not quickly retransfer a lost segment. An initial `TimeoutInterval` value of 1 second is recommended.

When a timeout occurs, the TimeoutInterval value is doubled in order to avoid a premature timeout for a subsequent segment. It is computed every time a segment is received and the `EstimatedRTT` is updated.

3.5.4 Reliable Data Transfer

1. TCP receive data from an application, encapsulates it in a segment, and passes it to IP starting the timer
2. TCP responds to a timeout retransmitting the segment that caused it and restarts the timer
3. TCP compares the ACK value received with its **SendBase** variable holding the sequence number of the oldest unacknowledged byte and either acknowledges a new segment or restarts the timer

3.5.5 Flow Control

TCP provides **flow control** in order to eliminate the receiver's buffer overflowing. The sender maintains a **rwnd** variable with the free **RcvBuffer** space and the following variables are defined:

- **LastByteRead**: number of the last byte read from the receiver buffer
- **LastByteRcvd**: number of the last byte arrived at the receiver buffer

The sender keeps track of two different variables: **LastByteSent** and **LastByteAcked**.

3.5.6 TCP Connection Management

The three-way handshake needs three steps:

1. client-side TCP sends a **SYN segment** to the server-side TCP containing a `SYN` bit set to 1
2. server-side TCP extracts the segment, allocates buffers and variables, and sends back a connection-granted **SYNACK segment**
3. client-side allocates buffers and variables and sends a segment with `SYN` set to 0

3.6 Principles of Congestion Control

Congestion happens when there is too much data traffic for the network to handle.

3.6.1 The Causes and the Costs of Congestion

Causes and costs of congestion need some insights:

- Throughput never exceeds capacity
- Delay increases as the maximum capacity is approached
- Packet loss and retransmission decrease the throughput
- Unneeded duplicates decrease the throughput
- Upstream transmission capacity and buffering are wasted for packets lost downstream

3.6.2 Approaches to Congestion control

Congestion-control approach is based on whether the network layer assists the transport layer:

- **End-to-end**: no support from the network and TCP decreases its window size according to packet loss
- **Network-assisted**: routers provide feedbacks about congestion

4 The Network Layer: Data Plane

4.1 Overview of Network Layer

The network layer transports segment between hosts: the sender encapsulates segments into datagrams and passes them to the link layer, while the receiver delivers segments to the transport layer.

4.1.1 Forwarding and Routing: The Data and Control Planes

Two important functions can be identified:

- **Forwarding**: routers must move arriving packets to the appropriate output link using **forwarding tables** with links interfaces
- **Routing**: the packets' path through the network is determined by some **routing algorithms**

Control Plane: The Traditional Approach

The **Data Plane** is a per-router function that determines how datagrams arriving to routers' input port are forwarded to the output port, while the **control plane** is a network wide logic determining how datagrams are routed along their path.
In the traditional approach, routing algorithms are implemented in routers.

Control Plane: The SDN Approach

In the **software-defined network (SDN)** approach, the controller which computes forwarding tables and interacts with routers is implemented in software.

4.1.2 Network Service Model

The **network service model** defines the end-to-end packets delivery characteristics:

- **Guaranteed delivery**: a packet source will eventually arrive at destination
- **Guaranteed delivery with bounded delay**: packets are delivered within a specified delay bound
- **In-order packet delivery**: packets arrive in the order that they were sent
- **Guaranteed minimal bandwidth**: as long as packets are transmitted below a specified bit rate, then they eventually arrive to destination
- **Security**: datagrams can be encrypted at the source and decrypted at destination

The Internet's network layer only provides a **best-effort service**, where there is no guarantee at all but some reflections:

- Simplicity has made Internet widely deployed adopted
- Sufficient provisioning of bandwidth allows sufficient performances
- Replicated application-layer services allow services to be provided from multiple locations
- Congestion control helps

4.2 What's Inside a Router

Four router components can be identified:

- **Input ports**: terminates an incoming physical link, performs link-layer functions to operate with the one on the other side, and consults the forwarding table to move the packet
- **Switching fabric**: connects input ports to output ports
- **Output ports**: stores packets and transmits them on the appropriate link
- **Routing processor**: executes routing protocols and computes or receives forwarding tables

Let's consider the information required for packet forwarding:

- **Destination-based forwarding**: forwarding process is based only on destination IP address
- **Generalized forwarding**: forwarding process is based on any set of field values

4.2.1 Input Port Processing and Destination-Based Forwarding

When looking for a forwarding table entry, if there are more matching values, the **longest prefix match** is chosen and the packet is forwarded to the associated interface.

4.2.2 Switching

Switching fabric transfers packets from input links to output link at specific **switching rate** in different manners:

- **Switching via memory**: packet copied to processor's memory, then the destination address is extracted and copied to the appropriate output port's buffer
- **Switching via bus**: input ports pre-pend a switch-internal label and transfer packets via a bus to all the output ports, where it is either kept or discarded
- **Switching via interconnection network**: a crossbar switch connects N input ports to N output ports through $2N$ buses and closes crosspoints while transferring packets

4.2.3 Output Port Processing

Output port processing takes packet stored in its memory and transmits them, including selecting and dequeuing functions.

4.2.4 Where Does Queuing Occur?

Input Queuing

If the switch fabric is slower than input ports combined, queuing may occur at input queues, eventually implying queuing delays and packet loss.
If a queued datagram at the front of the queue prevents others from moving forward even with different output link associated, then **head-of-line (HOL) blocking** occurs.

Output Queuing

At output buffers, **buffering** is required when datagrams arrive from a switch fabric which is faster than the link transmission rate, eventually implying congestion and packet loss: when there is not enough space, packets are drop depending on the policy adopted, which is either **drop-tail** or **drop-priority**. Every time that a packet is transferred, a **packet scheduler** must choose packets to transmits among those queued.
A larger buffer is not always the best option, since delays could increase with too much buffering and performances could be lowed by long RTTs.

4.2.5 Packet Scheduling

Packet scheduling is the operation of deciding which packets to send next on a link.

First-in-First-Out (FIFO)

In FIFO scheduling, also known as **FCFS**, packets are transmitted in the order that they arrive to the output port.

Priority Queuing

In **priority scheduling**, packets are classified by priority classes: the transmitting order is based on the packets priority value.

Round Robin and Weighted Fair Queuing (WFQ)

In **round robin (RR)** scheduling, arriving packets are queued by class: the server cyclically and repeatedly scans the queues sending one complete packet for each of them in turn.
The **weighted fair queuing (WFQ)** works as the RR scheduling, but each class gets a weighted amount of service in each cycle, with a minimum bandwidth guarantee per class.

4.3 The Internet Protocol (IP): IPv4, Addressing, IPv6, and More

4.3.1 IPv4 Datagram Format

The **IPv4** datagram fields are:

- **Version number**, 4 bits: IP protocol version
- **Header length**, 4 bits: where the payload begins
- **Type of service (TOS)**: distinguishes datagrams from each other
- **Datagram length**, 16 bits: total datagram length

- **Identifier**, **flags**, **fragmentation offset**: three fields having to do with fragmentation
- **Time-to-live (TTL)**
- **Protocol**: indicates the transport-layer protocol to which the datagram must be passed
- **Header checksum**: aids routers in detecting bit errors
- **Source and destination IP addresses**
- **Options**: more options needed by the datagram
- **Data**: contains the payload

4.3.2 IPv4 Addressing

The boundary between the host and its physical link is called an **interface** and has a unique IP address.
An IPv4 address is 32 bits long, meaning that there are 2^{32} possible addresses, and written in **dotted-decimal notation**.
Detaching an interface from its host creating an island of isolated networks, with the interface terminating their end points means identifying a **subnet**.
Internet's address assignment is known as **Classless Interdomain Routing (CIDR)**: the IP address is divided into two parts and has the format `a.b.c.d/x`, where `x` indicates the first part bits number and the `x` most significant bits constitutes the **prefix**.
Before CIDR, **classful addressing** divided subnets with 24-, 16-, and 8-bit addresses into **classes A**, **B**, and **C** respectively.
If a datagram is sent to the **broadcast address** `255.255.255.255`, it is delivered to all hosts in the subnet.

Obtaining a Host Address: The Dynamic Host Configuration Protocol

Hosts address are configured by the **Dynamic Host Configuration Protocol (DHCP)**: an host obtains an IP address from a network when it joins it. The four steps are:

1. **DHCP server discovery**: client broadcasts a **DHCP discover message** containing an IP address of `0.0.0.0`
2. **DHCP serve offers**: server broadcasts a **DHCP offer message** containing the proposed IP address and a **lease time** for it
3. **DHCP request**: client responds to the best offer with a **DHCP request message** containing the connection parameters
4. **DHCP ACK**: server responds with a **DHCP ACK message** confirming the parameters

4.3.3 Network Address Translation (NAT)

With **Network Address Translation (NAT)**, all devices in a private network have 32-bit addresses in a private address space and they can be changed without notifying the outside world. The NAT router must:

- replace the outgoing datagrams IP source address and port number with the NAT IP address and a new port number
- record every translation pair in a **NAT translation table**
- replace the incoming datagrams NAT IP destination address and port number with the corresponding IP address and port number stored in the NAT table

The NAT router appears as a single device with a single IP address to the rest of the world.

4.3.4 IPv6

IPv6 Datagram Format

There are some important changes in the format:

- **Expanded addressing capabilities**: IP addresses size increased to 128 bits and introduced the **anycast address**, which can deliver a datagram to any one of a group of hosts
- **Header length**: many fields have been dropped or made optional allowing faster processing
- **Flow labeling**: differentiates among the flows depending on their type

The following fields are defined:

- **Version**: IP version number
- **Traffic class**: gives priority to a certain type of datagrams
- **Flow label**: identifies a flow of datagrams
- **Payload length**
- **Next header**: protocol to which the data field must be delivered
- **Hop limit**: decrements by one at each router forwarding the datagram
- **Source and destination addresses**
- **Data**

Some IPv4 fields are no longer present:

- **Fragmentation**: no intermediate fragmentation for speed up forwarding
- **Checksum**: redundant because of transport- and link- layers checksums
- **Options**: one of the possible next headers pointed to from within the header

4.3.5 Transitioning from IPv4 to IPv6

The transition approach involves **tunneling**: a **tunnel** is an IPv4 router connected to IPv6 nodes. The sender node puts the IPv6 datagram in the data field of an IPv4 datagram, the router forwards it as a normal datagram, and the receiver extracts the IPv6 datagram.

4.4 Generalized Forwarding and SDN

In **generalized forwarding** each router contains a **flow table** which works with a "**match + action**" abstraction.
OpenFlow is a communications protocol that gives access to the forwarding plane of a switch or a router.

4.4.1 Match

OpenFlow's match is made on a set of selected fields.

4.4.2 Action

The most important actions are:

- **Forwarding**: packets can be forwarded, broadcasted, multicasted, encapsulated, and more
- **Dropping**: no entry means that a matched packet must be dropped
- **Modify-field**: fields values may be re-written before the packet is forwarded

4.5 Middleboxes

A **middlebox** is an intermediary box performing functions apart from standard ones on a network path. They provide three services:

- **NAT translation**: NAT boxes implement private network addressing
- **Security**: **firewalls** block traffic based on field values or redirect packets for additional processing, e-mail filters block dangerous mails, intrusion detection systems (IDSs) detect predetermined patterns and filter packets
- **Performance**: compression, caching, load balancing of requests to servers

They were initially proprietary hardware solutions, but nowadays they are **whiteboxes** implementing open APIs with programmable local actions.
With the **network function virtualization (NFV)** approach, services for computation, networking, and storage are implemented over whiteboxes.

5 The Network Layer: Control Plane

5.1 Introduction

There are two approaches for computing forwarding and flow tables:

- **Per-router control**: each router has a component communicating with the others to compute the tables
- **Logically centralized control**: logically centralized controller computes and distributes the tables

5.2 Routing Algorithms

Routing algorithms determine the fastest and least congested path through the network and are formulated using graphs. A **graph** $G = (N, E)$ is a set of N nodes and a collection of E edges. One way to classify algorithms is:

- **Centralized**: takes as input the connectivity between all nodes and all link costs, which are global state information (**link-state -LS- algorithms**)
- **Decentralized**: there is no global state information and the calculation is carried out in an iterative, distributed manner

A second way to classify them is:

- **Static**: routes change slowly over time
- **Dynamic**: routes change quickly in response to topology or link costs

5.2.1 The Link-State (LS) Routing Algorithm

Dijkstra's algorithm is a centralized, iterative algorithm that computes the k best paths for k nodes after k iterations:

```
Initialization:
N' = u    # computes the shortest paths from u to all other nodes
for all nodes v
  if v adjacent to u    # u only knows path costs to its neighbors
    D(v) = c_{u,v}    # set the estimated shortest path to v
Loop for all nodes in N':
find w not in N' such that D(w) is a min
add w to N'
```

```
update D(v) for all nodes adjacent to w and not in N':
    D(v) = min(D(v), D(w) + c_{w,v})
```

Since at each iteration it needs to check all the nodes not in N', the complexity is $O(n^2)$, but there is a more efficient implementation with a complexity of $O(nlogn)$.

5.2.2 The Distance-Vector (DV) Routing Algorithm

The **distance-vector (DV) algorithm** is an iterative, distributed, and self-stopping algorithm in which every node sends its own estimated distance vector to its neighbors (when it changes), which update their own DV using the **Bellman-Ford equation** and notify their neighbors if necessary.

```
At each node x:
Initialization:
for all destinations y in N:
  D_x(y) = c(x,y)    # inf if y is not a neighbour
for each neighbor w:
  D_w(y) = ? for all y in N
for each neighbor w:
  send D_x = [D_x(y): y in N] to w
Loop:
  wait until something happens
  for each y in N
    D_x(y) = min_v c(x,v) + D_v(y)
if D_x(y) changed for any y
  send D_x = [D_x(y): y in N] to all neighbors
forever
```

A Comparison of LS and DV Routing Algorithms

- **Message complexity**
  - LS: n routers = $O(n^2)$ messages sent
  - DV: convergence time varies
- **Speed of convergence**
  - LS: $O(n^2)$, but may have oscillations
  - DV: routing loops and count-to-infinity problem
- **Robustness**
  - LS: each router computes its own table and advertise incorrect links costs
  - DV: routers can advertise incorrect paths costs and their tables are used by others

5.3 Intra-AS Routing in the Internet: OSPF

LS and DV algorithms assume all routers identical in a flat networks. An **autonomous system (AS)** consists of a group of routers under the same administrative control that runs the same routing algorithm, which is called an **intra-autonomous system** protocol.

5.3.1 Open Shortest Path First (OSPF)

The **Open Shortest Path First (OSPF)** is a link-state protocol that uses flooding of link-state information and Dijkstra's algorithm in order to construct a complete topological map of the AS. Some of the advantages are:

- **Security**: all messages are authenticated to prevent malicious intrusion
- **Multiple same-cost paths**: multiple same-cost paths are used at same time
- **Integrated support**: multicast OSPF (MOSPF) adds a new type of link-state advertisement
- **Support for hierarchy within a single AS**: an AS can be configured hierarchically into areas, each running its own algorithm, and with a backbone area which routes traffic between the others

5.4 Routing Among the ISPs: BGP

All ASs run the same **inter-autonomous system routing protocol** known as **Border Gateway Protocol (BGP)** and is "*the glue that keeps the Internet together*".

5.4.1 The Role of BGP

BGP provides to each router a means to:

- **Obtain prefix reachability information from neighboring ASs**: allows each subnet to advertise its existence to the Internet
- **Determine the best routes to the prefixes**: best route is determined based on policy and reachability information

5.4.2 Advertising BGP Route Information

For each AS, each router is either a **gateway router**, connected to other ASs, or an **internal router**, connecting routers within an AS. BGP routers exchange messages over a semi-permanent TCP connection called a **BGP connection**. The messages types are:

- **OPEN**: opens a BGP connection and authenticates the sender

- **UPDATE**: advertises new paths or withdraw old ones
- **KEEPALIVE**: acknowledges the OPEN request and keep the connection alive
- **NOTIFICATION**: reports error in messages or closes the connection

5.5 Determining the Best Routes

A router advertising a prefix over a BGP connection includes with it **BGP attributes**. The two most important attributes are: **AS-PATH**, indicating the ASs traversed by the prefix advertisement, and **NEXT-PATH**, indicates to a specific router the next-hop AS.

Hot Potato Routing

In **hot potato routing**, the route chosen is that with the least cost to the `NEXT-HOP` router beginning that route.

5.6 The SDN Control Plane

5.7 ICMP: The Internet Control Message Protocol

The **Internet Control Message Protocol** is used by nodes to communicate network-level information. An ICMP message consists of:

- Type
- Code: ICMP subtype
- Checksum: calculated over ICMP header and data
- Data: copy of the ICMP header and at least 8 bytes of IP data

6 The Link Layer and LANs

6.1 Introduction to the Link Layer

We refer to any device that runs a link-layer protocol as a **node**, such as hosts and routers, and to communication channels between adjacent nodes as **links**. A transmitting node encapsulates the datagram in a **link-layer frame** and transmits the frame into the link.

6.1.1 The Services Provided by the Link Layer

Provided services can vary from one link-layer protocol to the next:

- **Framing**: encapsulating each datagram within a link-layer frame consisting of a data field, in which the network-layer datagram is inserted, and a number of header fields
- **Link access**: a medium access control (MAC) protocol specifies the rules by which a frame is transmitted and coordinates the transmissions if many nodes share a broadcast link
- **Reliable delivery**: each network-layer datagram is moved without error with acknowledgments and retransmissions
- **Error detection** and correction: the transmitting node includes error-detection bits in the frame and the receiving node performs an error check

6.1.2 Where Is the Link Layer Implemented?

The link layer is implemented on a chip called the **network adapter**, also known as **network interface controller (NIC)**, which can implement many services such as framing and link access. On the sending side, the controller takes a datagram created and stored in host memory by the higher layers, encapsulates it in a link-layer frame, and transmits it into the communication link. On the receiving side, a controller receives the entire frame and extracts the network-layer datagram.

6.2 Error-Detection and -Correction Techniques

At the sending node, data D to be protected is augmented with error-detection and -correction bits EDC. At the receiving node, a sequence of bits D! and EDC! is received and may differ from the original as a result of in-transit bit flips. The receiver may detect bit errors, but there still may be **undetected bit errors** allowing the receiver to deliver a corrupted datagram to the network layer.

6.2.1 Parity Checks

The simplest form of error detection is the inclusion of a single **parity bit**: in even parity schemes, its value is chosen such that the number of 1s in the d + 1 bits is even, while in odd parity schemes, it is chosen such that there is an odd number of 1s. The receiver counts the 1s in the received bits and if an odd number is found with an even parity scheme, it knows that an odd number of bit errors occurred. If an even number of bit errors occur, it would result in an undetected error. Errors are often clustered together in bursts and the probability of undetected errors with single-bit parity can approach 50%. In **two-dimensional parity scheme**, the d bits are divided into i rows and j columns and a parity value is computed for each of them: the resulting i + j + 1 parity bits comprise the link-layer frame's error-detection bits. If a single bit error occurs, the parity of both the column and the row containing the flipped bit will be in error and the receiver can use the indices to identify and correct the corrupted bit. The ability of the receiver to both detect and correct errors is known as **forward error correction (FEC)**. It decreases the number of sender retransmissions required and allow for immediate correction of errors at the receiver.

6.2.2 Checksumming Methods

In checksumming techniques, the d bits are treated as a sequence of k-bit integers. One simple checksumming method sums the integers and use the result as the error-detection bits. The receiver checks the checksum by taking the 1s complement of the sum of the received data, indicating an error if any of the resulting bits are 1.

6.2.3 Cyclic Redundancy Check (CRC)

A technique used widely in computer networks is based on **cyclic redundancy check (CRC)** codes, also known as polynomial codes. Consider the initial d bits: sender and receiver must first agree on an r + 1 bit pattern, known as a **generator G**, with the most significant bit = 1. The sender appends the r bits to the d bits such that the d + r bit pattern is divisible by G using modulo-2 arithmetic. The receiver divides the d + r bits by G and if the remainder is nonzero, an error occurred. All calculations are done in modulo-2 arithmetic without carries or borrows, meaning that addition and subtraction are identical and are equivalent to the exclusive-or.

6.3 Multiple Access Links and Protocols

A **point-to-point link** consists of a single sender at one end of the link and a single receiver at the other end. A **broadcast link** can have multiple sending and receiving nodes connected to a single broadcast channel and when a node transmits a frame, each of the others receives a copy. Computer networks have

protocols called **multiple access protocols** by which nodes regulate transmissions into the broadcast channel. All of the nodes receive multiple frames at the same time, meaning that frames **collide** at all of the receivers and are lost. It is possible to classify any multiple access protocol in three categories: **channel partitioning protocols**, **random access protocols**, and **taking-turns protocols**. A protocol for a broadcast channel of rate R bps should have the following characteristics:

1. a single node having data to send has a throughput of R bps
2. each of M nodes having data to send has an average throughput of R/M bps
3. the protocol is decentralized, meaning that there is no master node representing a point of failure for the network
4. the protocol is simple and inexpensive to implement

6.3.1 Channel Partitioning Protocols

Time-division multiplexing (TDM) and frequency-division multiplexing (FDM) can be used to partition a broadcast channel's bandwidth. TDM divides time into **time frames** and again into **time slots**, each assigned to one node. Whenever a node has a packet to send, it transmits during its assigned time slot in the revolving TDM frame. Each node gets a dedicated transmission rate of R/N bps during each frame time, but there are two drawbacks: a node is limited to that average rate even if it is the only node with packets to send and a node must always wait for its turn in the transmission sequence. FDM divides the channel rate into different frequencies, each with a bandwidth of R/N, and assigns each of them to one node creating smaller channels of R/N bps. FDM shares both the advantages and drawbacks of TDM. A third partitioning protocol is **code division multiple access (CDMA**). It assigns a different code to each node, which uses it to encode the data bits to send. CDMA networks have the property that different nodes can transmit simultaneously having their respective receivers correctly receive encoded data bits in spite of interfering transmissions by other nodes.

6.4 Random Access Protocols

The second broad class of multiple access protocols are random access protocols. A transmitting node always transmits at the full rate of the channel and when there is a collision each node involved retransmits its frame waiting a random delay.

Slotted ALOHA

The simplest random access protocols is the **slotted ALOHA** protocol. Let's assume the following:

- all frames consist of L bits
- time is divided into slots of L/R seconds
- nodes start to transmit frames only at the beginnings of slots
- nodes are synchronized so that each node knows when the slots begin
- all nodes detect a collision event before the slot ends
- if a node has a frame to send waits until the beginning of the next slot
- if there isn't a collision, the node has successfully transmitted its frame
- if there is a collision, the node detects the collision before the end of the slot and the frame is retransmitted in each subsequent slot with probability p until there is not a collision

This protocol allows a node to transmit at the full rate R when that node is the only active node. It is highly decentralized since each node detects collisions and independently decides when to retransmit.

6.4.1 Taking-Turns Protocols

The **polling protocol** designs a node as a master node which **polls** each node in a round-robin fashion. The master node sends a message to a node allowing its transmission up to a number of frames, eliminating collisions and empty slots that plague random access protocols. The protocol has a few drawbacks: it introduces a polling delay, which is the amount of time required to notify a node, and if the master node fails, the channel becomes inoperative. Another protocol is the **token-passing protocol**, where a special-purpose frame known as a **token** is exchanged among the nodes in a fixed order. When a node receives a token, it holds onto it if it has some frames to transmit, sends up to a maximum number of frames, and forwards the token to the next node. There are a few drawbacks: the failure of one node can crash the channel and if a node accidentally neglects to release the token, then some recovery procedure must be invoked.

6.4.2 DOCSIS: The Link-Layer Protocol for Cable Internet Access

The **Data-Over-Cable Service Interface Specifications (DOCSIS)** specifies the cable data network architecture and its protocols, using FDM to divide the downstream and upstream network segments into multiple frequency channels. A downstream channel is 24 MHz to 192 MHz wide with a max throughput of 1.6 Gbps, while each upstream channel is 6.4 MHz to 96 MHz wide with a max throughput of 1 Gbps. Each channel is a broadcast channel: frames transmitted on the downstream channel are received by all cable modems receiving that channel with no access problem, while in the upstream direction multiple modems share the same channel and collisions can occur. The CMTS grants

permission to individual cable modems to transmit during specific mini-slots by sending a MAP message on a downstream channel specifying which modem can transmit. The mini-slot-request frames are transmitted in a random access manner and so may collide with each other and when a collision is inferred, a modem uses binary exponential backoff to defer the retransmission.

6.5 Switched Local Area Networks

6.6 Link-Layer Addressing and ARP

MAC Addresses

A host with multiple network interfaces will have multiple link-layer addresses and IP addresses associated with it, while link-layer switches do not have link-layer addresses associated with their interfaces connected to hosts since carrying datagrams between hosts is done without the hosts having to explicitly address the frame to the intervening switch. A link-layer address is called a **LAN address**, a **physical address**, or a **MAC address**, and is 6 bytes long, each expressed as a pair of hexadecimal numbers. No two adapters have the same address: when a company wants to manufacture adapters, it purchases a chunk of the address space consisting of 224 addresses which are allocated by IEEE fixing the first 24 bits of a MAC address and letting the company create unique combinations of the last 24 bits for each adapter. An adapter's MAC address has a flat structure and never changes. An adapter inserts the destination MAC address into a frame and then sends it into the LAN. A switch can broadcast the incoming frame onto all of its interfaces, having an adapter receiving a frame that isn't addressed to it. If there is a match between the frame and the destination address, the adapter extracts the enclosed datagram and passes it up the protocol stack, otherwise the frame is discarded. If an adapter wants all the other adapters to receive a frame inserts a special **MAC broadcast address**.

Address Resolution Protocol (ARP)

Since there are both network-layer addresses and link-layer addresses, the Internet **Address Resolution Protocol (ARP)** translates between them. An ARP module in a sending host takes an IP address on the same LAN as input and returns the corresponding MAC address. Each host has an **ARP table** containing mappings of IP addresses to MAC addresses and a **time-to-live (TTL)** value indicating when each mapping will be deleted from the table. If the ARP table doesn't have an entry for the destination, the sender constructs a special packet called an **ARP packet**, which includes the sending and receiving IP and MAC addresses, to query all the other hosts to determine the MAC address corresponding to the resolved IP address.

6.6.1 Ethernet

Ethernet Frame Structure

The sending adapter encapsulates the IP datagram within an Ethernet frame and passes it to the physical layer, while the receiving adapter receives the frame from the physical layer, extracts the IP datagram, and passes it to the network layer. The Ethernet frame fields are:

- Data field (46 to 1,500 bytes): carries the IP datagram; if it exceeds 1,500 bytes, then the host has to fragment the datagram, while if it is less than 46 bytes, the data field has to be stuffed
- Destination address (6 bytes): contains the MAC address of the destination adapter
- Source address (6 bytes): contains the MAC address of the adapter transmitting the frame onto the LAN
- Type field (2 bytes): permits to multiplex network-layer protocols
- Cyclic redundancy check (CRC) (4 bytes): allows the receiving adapter to detect bit errors in the frame
- Preamble (8 bytes): the first 7 bytes of the preamble wake up the receiving adapters and synchronize their clocks to that of the sender's one, while the last 2 bits of the last byte alert the receiver that the data is coming Ethernet technologies provide connectionless service to the network layer, meaning without first handshaking, and an unreliable service to the network layer.

6.6.2 Link-Layer Switches

Switches receive incoming link-layer frames and forward them, being **transparent** to the hosts. They have buffers, since the rate at which frames arrive to a switch output interface may exceed the link capacity of that interface.

Forwarding and Filtering

Filtering determines whether a frame should be forwarded or dropped. **Forwarding** determines the interfaces to which a frame should be directed and moves the frame to those interfaces. Both the functions are done with a switch table containing entries for some of the hosts on a LAN: an entry contains a MAC address, the switch interface that leads toward that MAC address, and the time at which the entry was placed in the table. Suppose a frame with destination address DD-DD-DD-DD-DD-DD arrives at the switch on interface x, then there are three possible cases:

- no entry: the switch broadcasts the frame

- entry exists with x: there being no need to forward the frame to any of the other interfaces, the switch performs the filtering function by discarding the frame
- entry exists with y: the switch performs the forwarding function by putting the frame in an output buffer preceding interface y

Self-Learning

Switches are **self-learning**, meaning that the tables are built automatically, dynamically, and autonomously:

1. switch table is initially empty
2. for each frame received on an interface, the switch stores in its table the MAC address, the interface from which the frame arrived, and the current time
3. an address is deleted if no frames are received with that address as the source address after a period called the **aging time**

Switches are **plug-and-play** devices, since they require no intervention from a network administrator or user, and full-duplex, meaning any switch interface can send and receive at the same time.

Properties of Link-Layer Switching

- Elimination of collisions: switches buffer frames and never transmit more than one frame on a segment at any time
- Heterogeneous links: since a switch isolates one link from another, the different links in the LAN can operate at different speeds and can run over different media
- Management: switches ease network management, for example by detecting a malfunctioning adapter or gathering statistics on bandwidth usage, collision rates, and traffic types

Switches Versus Routers

A switch is different from a router in that it forwards packets using MAC addresses. Whereas a router is a layer-3 packet switch, a switch is a layer-2 packet switch. Pro and cons of switches:

+ plug-and-play
+ high filtering and forwarding rates
+ process frames only up through layer 2
- large ARP tables and substantial ARP traffic

- susceptible to broadcast storms

Pro and cons of routers:

+ packets do not cycle through routers even if the network has redundant paths
+ allowed the Internet to be built with a rich topology
+ firewall protection against layer-2 broadcast storms
- process datagrams up through layer 3
- not plug-and-play
- larger per-packet processing time

Switches suffice for small networks as they localize traffic and increase aggregate throughput without requiring any IP address configuration, while larger networks include routers.

6.6.3 Virtual Local Area Networks (VLANs)

Three drawbacks can be identified in LAN configuration:

- Lack of traffic isolation: broadcast traffic traverses the entire institutional network and limiting its scope would improve LAN performance and could be required for privacy
- Inefficient use of switches: if each group were small, then a single switch would be large enough to accommodate everyone, but it would not provide traffic isolation
- Managing users: if an employee moves between groups, the physical cabling must be changed to connect the employee to a different switch

7 Wireless and Mobile Networks

7.1 Introduction

In a wireless network, the following elements can be identified:

- Wireless hosts: end-systems that run applications
- Wireless links: connects a host to a base station or to another host
- Base station: sends and receives data to and from associated hosts, which are within the communication distance and uses the base to relay data, and are referred to as operating in infrastructure mode

- Network infrastructure: larger network with which a host may wish to communicate

In ad hoc networks, hosts have no infrastructure with which to connect, so they provide for services such as routing and address assignment. When a mobile host moves into the range of another base station, it changes its point of attachment in a process referred to as handoff or handover. It is possible to classify wireless networks according to two criteria based on the number of hops crossed by a packet and on the existence of an infrastructure:

- Single-hop, infrastructure-based: the base station is connected to the larger wired network and all communication is between the base station and a host over a single wireless hop
- Single-hop, infrastructure-less: no base station connected to a wireless network and one host may coordinate the transmissions of the others
- Multi-hop, infrastructure-based: the base station is wired to the larger network and some hosts may have to relay their communication through others to reach it
- Multi-hop, infrastructure-less: no base station exists and hosts may have to relay messages among several other nodes to reach a destination; nodes may be mobile with connectivity changing among hosts (mobile ad hoc networks - MANETs), or vehicles (vehicular ad hoc network - VANET)

7.2 Wireless Links and Network Characteristics

Wireless links differ from their wired counterparts in different ways:

- Decreasing signal strength: electromagnetic radiation attenuates as it passes through matter; in free space, the signal strength decreases due to path loss
- Interference from other sources: radio sources transmitting in the same frequency band and electromagnetic noise within the environment can result in interference
- Multipath propagation: occurs when portions of the electromagnetic wave reflect off objects and the ground, taking paths of different lengths between a sender and receiver and resulting in the blurring of the received signal

Wireless link protocols employ powerful CRC error detection codes and link-level reliable-data-transfer protocols that retransmit corrupted frames. A host receives an electromagnetic signal as a combination of a degraded form of the original signal transmitted by the sender and background noise in the environment: the signal-to-noise ratio (SNR) is a relative measure of its strength. There are several physical-layer characteristics important in understanding wireless communication protocols:

- the higher the SNR, the lower the bit error rate (BER); a sender can increase the SNR by increasing its transmission power and decrease the probability that a frame is received in error
- a modulation technique with a higher bit transmission rate has a higher BER
- dynamic selection of the physical-layer modulation technique is used to adapt the modulation technique to channel conditions: SNR and BER due to mobility or changes in the environment

Suppose that station A is transmitting to station B and that station C is transmitting to B. With the hidden terminal problem, physical obstructions in the environment may prevent A and C from hearing each other's transmissions. With the signal's strength fading, collisions at the receiver may be undetectable.

7.2.1 CDMA

When hosts communicate over a shared medium, a protocol is needed so that the multiple signals do not interfere. These protocols are divided into three classes: channel partitioning, random access, and taking turns. **Code division multiple access (CDMA)** belongs to channel partitioning protocols. In a CDMA protocol, each bit is encoded by multiplying the bit by a signal that changes at a faster rate known as the **chipping rate**. Let di be the value of the data bit for the i-th bit slot and let's represent the data bit with a 0 value as -1. Each bit slot is further subdivided into M mini-slots: the CDMA code consists of a sequence of M values each taking a +1 or -1 value. For the m-th mini-slot of the bit-transmission time of di, the output of the CDMA encoder is the value of di multiplied by by the m-th bit in the assigned CDMA code: With no interfering senders, the receiver would receive the encoded bits and recover the original data bit by computing:

7.3 WiFi: 802.11 Wireless LANs

7.3.1 The 802.11 Wireless LAN Architecture

The fundamental component of the 802.11 architecture is the **basic service set (BSS)**, which contains one or more wireless stations and a central base station, known as an access point (AP). Each station has a 6-byte MAC address in the adapter's firmware. Each AP also has a MAC address for its wireless interface. As with Ethernet, these MAC addresses are administered by IEEE and are (in theory) globally unique. LANs that deploy APs are referred to as **infrastructure wireless LANs**. Stations can also group themselves together to form an ad hoc network without a central control nor connections to the "outside world, formed "on the fly," by devices in proximity to each other.

Channels and Association

A wireless station needs to associate with an AP before sending or receiving data. When a administrator installs an AP, a one- or two-word **Service Set Identifier (SSID)** and a channel number are assigned to it. Channel number 802.11 defines 11 partially overlapping channels in the frequency range of 2.4 GHz to 2.4835 GHz. Any two channels are non-overlapping iff they are separated by four or more channels. A **WiFi jungle** is a physical location where a station receives a sufficiently strong signal from two or more APs. To gain Internet access, devices need to join one of the subnets and to **associate** with one AP. Devices perform **passive scanning**: APs periodically send **beacon frames**, each including their SSID and MAC address, and the devices scans the 11 channels, seeking beacon frames from any APs. The algorithm for selecting which of the available APs to associate with is left up to the designers. Devices can also perform **active scanning** by broadcasting a probe frame that will be received by all APs within the device's range, generating a probe response frame. After selecting the AP, devices send an association request frame, and the AP responds with an association response frame. This second handshake is needed with active scanning, since after the first probe request an AP doesn't know which of the responding APs the device will choose. Once associated with an AP, the device sends a DHCP discovery message into the subnet via the AP in order to obtain an IP address and to join it. A device may be required to authenticate itself to the AP, which communicates with an authentication server relaying information with the device using a protocol such as RADIUS or DIAMETER.

7.3.2 The 802.11 MAC Protocol

Once a device is associated with an AP, it can start sending and receiving data frames, but since multiple devices or the AP itself may want to transmit frames at the same time over the same channel, a multiple access protocol is needed. The designers chose a random access protocol referred to as **CSMA with collision avoidance (CSMA/CA)**: each station senses the channel before transmitting and refrains from transmitting when the channel is sensed busy. 802.11 uses collision-avoidance techniques and a link-layer acknowledgment/retransmission (ARQ) scheme. There are two important reasons for not implementing collision detection:

- requires the ability to send the station's own signal and receive to determine whether another station is transmitting at the same time and since the signal strength is very small compared with Ethernet, it is costly to build hardware that can detect a collision
- even if the adapter could transmit and listen at the same time, it would not be able to detect all collisions due to the hidden terminal problem and fading

Once a station begins to transmit, it transmits the frame in its entirety. When the destination receives a frame that passes the CRC, it waits a short period of time known as **Short Inter-frame Spacing (SIFS)** and sends back a **link-layer 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. If an acknowledgment is not received after a given number of retransmissions, the transmitting station discards the frame. Suppose that a station has a frame to transmit:

1. if the station senses the channel idle, it transmits its frame after a short period of time known as **Distributed Inter-frame Space (DIFS)**

2. otherwise, the station chooses a random value using binary exponential backoff and counts down after DIFS; while the channel is sensed busy, the counter is frozen

3. when the counter reaches zero, the station transmits the entire frame and waits for an acknowledgment

4. if an acknowledgment is received, the frame has been correctly received and 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 a larger interval

Let's consider a scenario in which two stations have a frame to transmit, but neither station transmits immediately since each senses a already transmitting station. Both the stations would each transmit as they detect that the third station has finished causing a collision. In CSMA/CD, both stations would abort the transmissions avoiding useless transmissions of the remainders. In 802.11, if the stations sense the channel busy, they enter random backoff hopefully choosing different values and once the channel becomes idle, one station will transmit before the other, while the "losing station" will hear the other's signal, freeze its counter, and refrain from transmitting. Collisions can still occur if the two stations are hidden from each other, or they choose close enough random backoff values.

Dealing with Hidden Terminals: RTS and CTS

Let's consider two stations and one access point, both within range of an AP and associated with it. Due to fading, each station is hidden from the other. Suppose station H1 is transmitting a frame and, halfway through the transmission, station H2 wants to send a frame. Since H2 does not hear H1's transmission, it will wait a DIFS interval and then transmit the frame, resulting in a collision. The channel will be wasted during the entire period of H1 and H2's transmissions. 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: when a station wants to send a frame, it can first send an RTS frame indicating the total time required to transmit the data

frame and the ACK frame. When the AP receives the RTS frame, it responds by broadcasting a CTS frame which gives the sender permission to send and also instructs the other stations not to send for the duration. RTS and CTS frames can improve performance in two ways:

- hidden station problem is mitigated, since a long frame is transmitted after the channel has been reserved
- since they are short, a collision will last only for the duration of the frame and once they are transmitted, the following data and ACK frames should be transmitted without collisions

The RTS/CTS exchange introduces delay and consumes channel resources, thus it is only used for long data frames: each wireless station can set an RTS threshold such that the RTS/CTS sequence is used only when the frame is longer than the threshold.

Using 802.11 as a Point-to-Point Link

If two nodes have a directional antenna, they can point them at each other and run the 802.11 protocol over a point-to-point link.

7.3.3 7.3.3 The IEEE 802.11 Frame

802.11 frame contains some specific fields:

Payload and CRC Fields

The payload can be as long as 2,312 bytes (typically ¡ 1,500 bytes) and consists of an IP datagram or an ARP packet.

Address Fields

The frame has four address fields, each holding a 6-byte MAC address:

- Address 1: MAC address of the station that receives the frame
- Address 2: MAC address of the station that transmits the frame
- Address 3: MAC address of the router interface connecting the subnet with other subnets
- Address 4: used when APs forward frames to each other in ad hoc mode

7.3.4 Sequence Number, Duration, and Frame Control Fields

Since acknowledgments can get lost, a station may send multiple copies of a frame. The sequence number field allow the receiver to distinguish between a newly transmitted frame and a retransmission. The time for which the channel is reserved is stored in the duration field. The control field includes many subfields:

- Type, Subtype: distinguish the association, RTS, CTS, ACK, and data frames
- to, from: define the meanings of the different address fields
- WEP: indicates whether encryption is being used or not

7.3.5 Mobility in the Same IP Subnet

In order to increase the physical range of a wireless LAN, it is possible to deploy multiple BSSs within the same IP subnet. This raises the issue of mobility among the BSSs: when stations move between subnets, some mobility management protocols will be needed. If a router moves to another BSS, then it would have to obtain a new IP address in the subnet and the address change would disrupt any on-going TCP connections. Otherwise, as the station wanders away from the first AP, it starts to scan for a stronger signal and receives beacon frames from a second AP. The station disassociates with the first AP and associates with the new one keeping its IP address and maintaining its ongoing TCP sessions. Since switches are "self-learning", they can handle occasional moves.

7.3.6 Advanced Features in 802.11

802.11 Rate Adaptation

Suppose that a user becomes mobile, walking away from the base station, with the SNR falling as the distance from the base station increases: if the modulation technique does not change, the BER will become unacceptably high and eventually no transmitted frames will be received. Some implementations have a rate adaptation capability that selects the underlying physical-layer modulation technique to use based on current channel characteristics. If a station sends two frames in a row without receiving an acknowledgment, the transmission rate falls back to the next lower rate.

Power Management

The 802.11 standard provides power-management capabilities that allow stations to minimize the amount of time that their sense, transmit, receive functions and other circuitry need to be on. A station alternates between sleep and wake states and indicates to the access point that it will be going to sleep by setting the power-management bit in the header of a frame to 1. A timer in the station is then set to wake it up just before the AP is scheduled to send its beacon frame. The AP buffers any frames destined for the sleeping host for later transmission.

7.3.7 Personal Area Networks: Bluetooth

Bluetooth networks operate over short ranges at low power and at low cost and are referred to as **wireless personal area networks (WPANs)** or **piconets**.

They operate in the unlicensed 2.4 GHz Industrial, Scientific and Medical (ISM) radio band along with other home appliances. The Bluetooth wireless channel is operated in a TDM manner with time slots of 625 μs and, during each of them, a sender transmits on one of 79 channels, with the frequency changing in a pseudo-random manner. This form of channel hopping is known as **frequency-hopping spread spectrum (FHSS)** and is used so that interference will only occur with communications in at most a subset of the slots. Bluetooth data rates can reach up to 3 Mbps. Bluetooth networks are ad hoc networks with one device designated as master and the others as clients. The master determines the slot-to-slot frequency hopping sequence, controls entry of the clients, controls the power at which client transmit, and uses polling to grant clients permission to transmit once admitted. There can also be up to 255 "parked" devices in the piconet in some form of "sleep mode" to conserve energy and will awaken to receive beacon messages from the master. These networks must be **self-organizing**: when a master wants to form a network, it first determines which devices are within range in a process known as **neighbor discovery** where the master broadcasts 32 inquiry messages, each on a different frequency channel, and repeats the sequence up to 128 times. When a client hears a message, it backs off a random amount of time from 0 s to 0.3 s and then responds with a message containing its ID. Once the master has discovered all of the potential clients, it invites them in a process called the **Bluetooth paging**, where the master informs the client of the frequency-hopping pattern and the sender's clock.

8 Appendix A - Formulae

- Transmission delay : $d_{trans} = L/R$
  - L : packet length [bits]
  - R : transmission rate [bps]
- Queuing delay : $d_{queue} = La/R$
  - a : traffic arriving rate [packets/s]
- Propagation delay : $d_{prop} = d/s$
  - d : distance [m]
  - s : propagation speed [m/s]
- Total delay : $d_{total} = d_{proc} + d_{queue} + d_{trans} + d_{prop}$
- End-to-end delay : $N \cdot d_{total}$
  - N : number of links
- End-to-end throughput : $F/min\{R_1, R_2, ..., R_N\}$

  - F : file size [bits]

- Distribution time (client-server) : $D_{cs} = max\left\{\frac{N \cdot F}{u_s}, \frac{F}{d_{min}}\right\}$
  - N : number of peers
  - u_s : server upload rate [bps]
  - d_i : ith peer download rate [bps]
- Distribution time (P2P) : $D_{cs} = max\left\{\frac{F}{u_s}, \frac{F}{d_{min}}, \frac{N \cdot F}{u_s + \sum_{i=1}^{N} u_i}\right\}$
- Utilization : $U_{sender} = \frac{L/R}{RTT + L/R}$
- Estimated RTT: $EstimatedRTT = (1-\alpha) \cdot EstimatedRTT + \alpha \cdot SampleRTT$
  - $\alpha = 0.125$
- RTT variation : $DevRTT = (1-\beta) \cdot DevRTT + \beta \cdot |SampleRTT - EstimatedRTT|$
  - $\beta = 0.25$
- Timeout interval : $TimeoutInterval = EstimatedRTT + 4 \cdot DevRTT$
- Receive window : $rwnd = RcvBuffer - (LastByteRcvd - LastByteRead)$
- Buffering : $B = RTT \cdot C/N$
  - C : link capacity
  - N : independent TCP flows
- WFQ : $WFQ_i = \frac{w_i}{\sum_{i=1} i}$
  - i : WFQ class
  - w : class weight
- Bellman-Ford : $D_x(y) = min_v\{c_{x,v} + D_v(y)\}$
  - $D_x(y)$: cost of the best path from x to y
  - $c_{x,v}$: direct link cost from x to v
  - w : class weight

9 Appendix B - Port Numbers

- SMTP : 25
- DNS : 53
- DHCP : 67
- HTTP : 80
- POP3 : 110
- IMAP : 143