Computer Networking: A Top-Down Approach (summary)

Created	@April 28, 2023 11:57 PM
Tags	
Person	 Wahba Kamaluddin

▼ Chapter 1: Introduction

▼ 1.1 What’s the Internet?

▼ “Nuts and Bolts” view - Consider internet as a machine

- Billions of connected devices, end systems=host
  - The hosts are connected to the internet via communication links (fiber, copper, radio, satellite)
- Internet is network of networks
  - the sending, receiving messages is controlled by protocols

▼ A service view

- Infrastructure that provides services to application

▼ What’s a protocol?

- Set of rules that controls the communication between computers on a network, it defines the format and sequence of messages exchanged between devices, and the actions taken by devices to response to the messages.

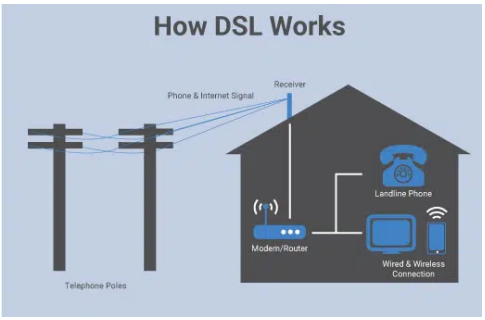
▼ 1.2 Network edge

▼ How does host send data?

- create data → break data into packets → transmits packets into network

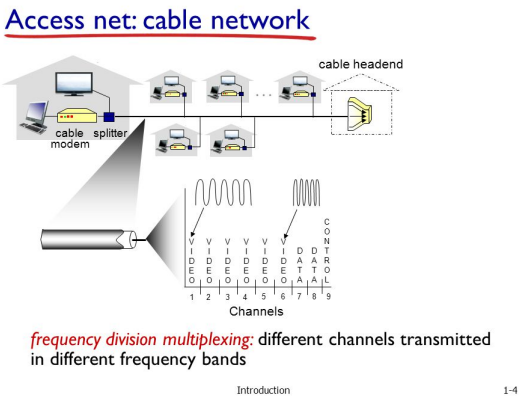
▼ Examples of access network (How user connect to the ISP)

▼ Digital Subscriber Line (DSL)



Use existing telephone line, share the same line for data and voice

▼ Cable Network



▼ Enterprise Access Networks (Ethernet)

- Connect device using physical cables (copper, fibre optic)
- Usually used in Local Area Network (LAN)
- High speed, low latency(delay that occur on data transmission (ms))

▼ Wireless Access Networks

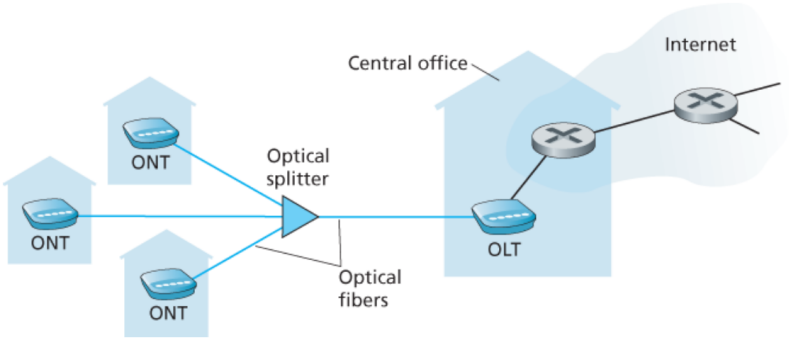
▼ Wireless LAN (WIFI)

- Uses radio waves

▼ Wide-Area Wireless Access (WAWA)

- Wireless network that covers wide geographic area
- eg:
  - Cellular Network-4g, 5g

        - Satellite Network
        - Microwave Network-travels long distance, but cant go through obstacle(used to connect between cell towers(backhaul connections))
    - ▼ Fibre To The Premise (FTTP)
        - One fibre from central office (CO)
        - ▼ Two architecture of Fibre Optic splitting:
            - Active Optical Network (AON)
            - Passive Optical Network (PON):



- ▼ What is physical media?
    - The medium used to transmit the data - sits in the physical layer of network architecture.
    - ▼ What are two types of physical media?
        - guided- uses physical wire to transmit the data
            - twisted pair cable, coaxial cable, fiber optic cable
        - unguided- transmit data wirelessly
            - radio waves, microwaves, infrared waves, bluetooth

- ▼ 1.3 Network core
    - ▼ What's a network core?
        - The central part of the network that manages the flow of data in a network
    - ▼ What are 2 common methods of transmitting data over a network ?

▼ What is **packet switching**?

- A networking technology that breaks up data into small units called packet before sending it through the network- each packets have the source and destination ip address

▼ What is store-and-forward transmission?

- The packet switch must receive the entire packet before it can begin to transmit it to appropriate output link

▼ How can delays and packet loss occur?

- Each packet switch has multiple link each with an output buffer, which stores packet.
- If the buffer is full, packet loss will occur— the arriving or the queing packets will be dropped

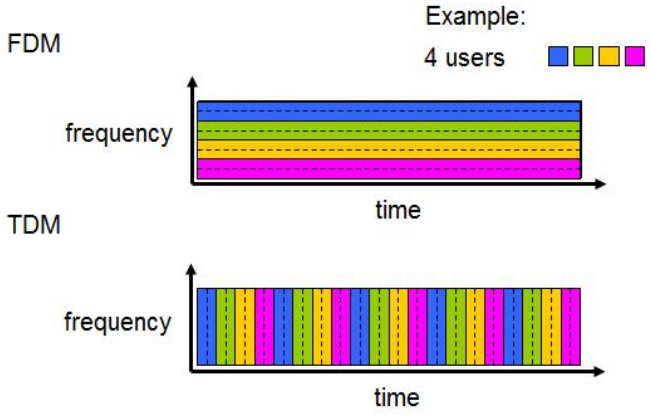
▼ How can a router determine the link to forward packet?

- Each packet has destination IP address, the router will search on forwarding table (automatically set by using routing protocol) — each router has it built in — to determine the route.
- Same like we drive to a far location, on each checkpoint, we ask the locals where to go

▼ What is **circuit switching**?

- A path is established between the sender and receiver, it remains reserved until the transmission complete.
  - The data is not broken into packets
  - Inefficient

▼ What are 2 common techniques used?



- Frequency Division Multiplexing (FDM)
  - The physical medium (Cable, etc) is divided into multiple frequency bands, each channel is assigned with different frequency band, the data is transmitted in each designated band
- Time Division Multiplexing (TDM)
  - The physical medium is divided into multiple time slots, each sender assigned to a time slot, data transmitted in the assigned time slot, time slots are repeated continuously, each sender takes turn

▼ What are 2 keys network core function?

▼ Packet forwarding

- Forwarding packets from one sender to receiver

▼ Routing

- Determining optimal path for packet to reach its destination

▼ 1.4 Delay, loss, throughput in networks

▼ How does delay occur?

- Queuing delay
  - packet arrival rate to link exceeds output link capacity, packets queue waiting for turn to be send
- Nodal processing
  - checking packet header, performing error checks

- Transmission delay
  - Times taken to transmit a packet to the communication channel- depends on length and link bandwidth
- Propagation delay
  - Time taken for data to travel from the sender to the receiver

▼ How does packet loss occur?

- Error in data transmission
- Damaged hardware
- Hardware capacity and bottlenecks

▼ What is throughput?

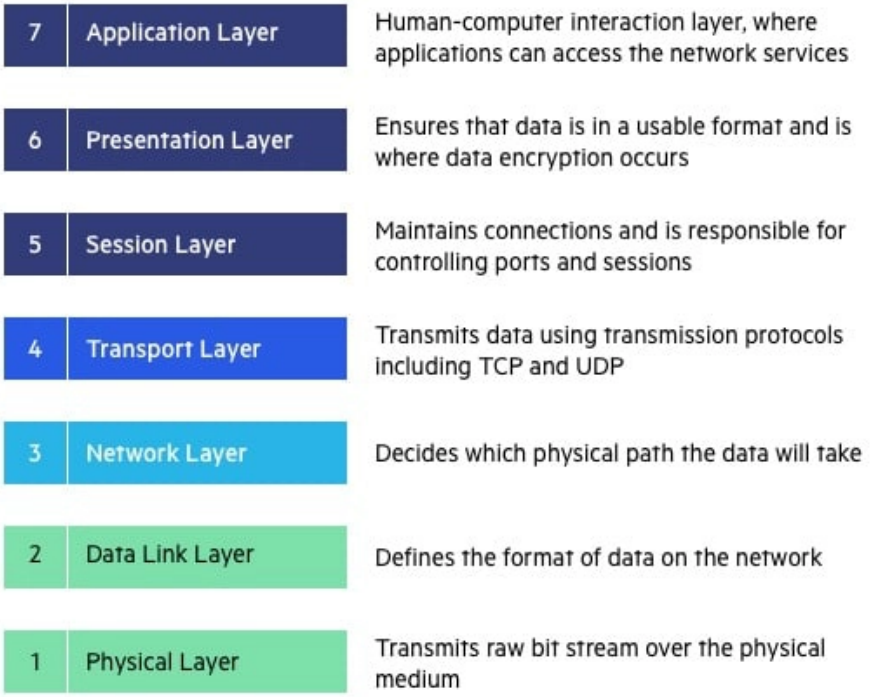
- amount of data that is transmitted over a network at a given time, kbps (bandwidth-max capacity of a communication channel)

  - the throughput from server to client will be the smallest value, either Rs or Rc (throughput = min{Rs,RC}) — called the bottleneck link

▼ 1.5 Protocol layers, service models

▼ Internet layer

▼ ISO/OSI reference model (Please Do Not Throw Sausage Pizza Away)

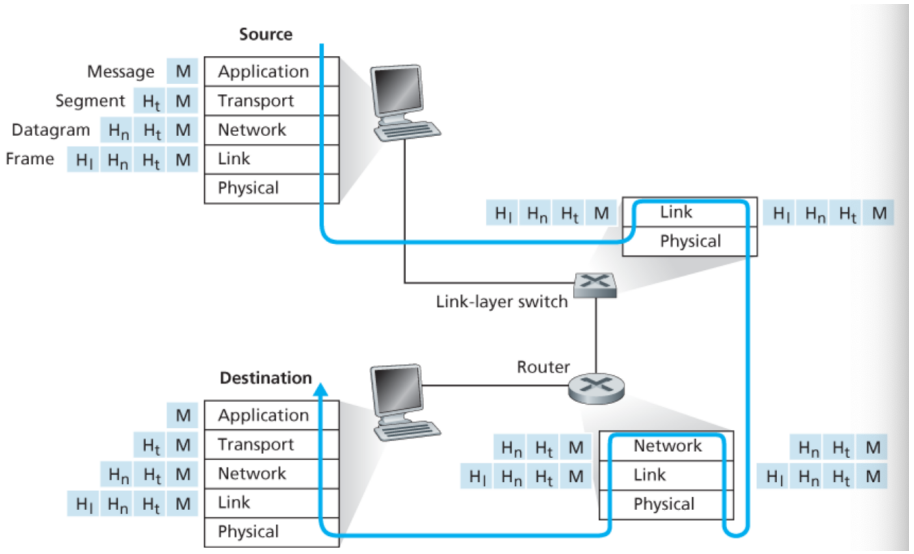


- Application Layer
  - network app, app-layer protocols reside such as HTTP, FTP, SMTP
  - The packet of information is called message
- Presentation Layer
  - Provide services that allow communication applications to interpret the data exchanged
- Transport Layer
  - transport application-layer message between two application endpoints — when arrived at the destination, it'll dictate which app request/ service it via the port number
  - The packet is called segment
  - There are two transport protocols:
    - Transmission Control Protocol, TCP
      - Provides connection-oriented service, guarantee delivery of messages, flow control (match sender/receiver speed)

            - Messages are broken into segments
        - Used Datagram Protocol, UDP
            - connectionless service, no reliability, no flow control, no congestion control
            - high speed transmission
    - Network Layer
        - facilitate the segment from transport layer to be able to be transported to appropriate host by encapsulating it using Internet Protocol (adding sender and receiver IP address)
        - Packet is called datagram
    - Link Layer
        - Facilitate the transportation on a local network
        - Packet is called frames
        - Protocols are: Ethernet, WIFI, etc — since the transportation will involves multiple link, the protocol used may changes accordingly
    - Physical Layer
        - Moving individual bits from one node to other

▼ What is encapsulation?

- As the packet goes through the layers, additional info is added as a header to make sure it arrives at the designated host



▼ 1B Network Topology

▼ What is network topology?

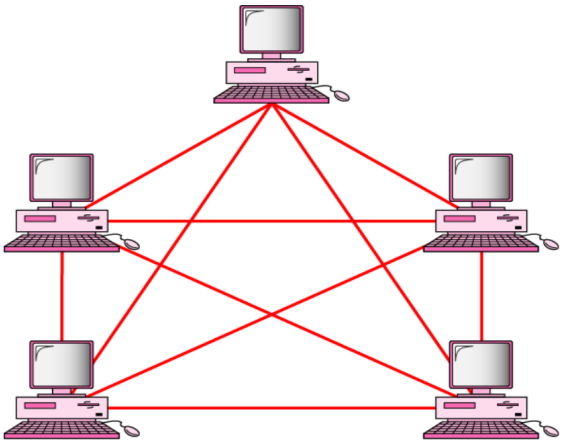
- The geometric representation of how links and devices are connected to each other

▼ What two relationships between connected devices that are possible?:

- Peer-to-peer, the devices share the link equally
- Primary-secondary, one device controls traffic, other transmit through it

▼ What are five basic topologies?

▼ Mesh



- Every device has a dedicated—the link carries traffic only between the two devices it connects—point-to-point link to every other device
- fully connected mesh:
  - total no. of dedicated links = (n*(n-1))/2
  - total ports on each devices = n-1

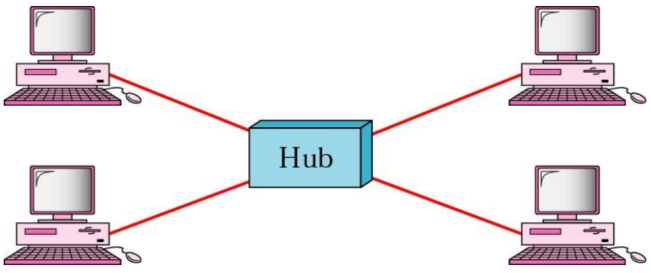
▼ Advantages

- Use dedicated links - eliminates traffic problem
- Robust - one device error doesn't effect other connections
- Privacy/ security - each connection between devices are dedicated
- Fault identification is easy

▼ Disadvantages

- Amount of cable and number of I/O port, expensive
- Installation and reconfiguration is difficult

▼ Star



- Each device has a dedicated point-to-point link only to a central controller (hub)
- Controller act as an exchange — receives and relays data (forwards to intended recipient)

▼ Advantages

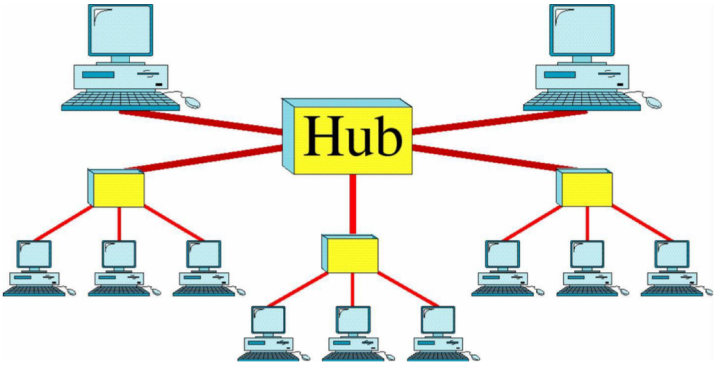
- Lower cost compared to mesh - less links, I/O ports

- Robust - if one link fails, others aren't affected

▼ Disadvantages

- more cabling required compared to tree, ring, bus topologies

▼ Tree

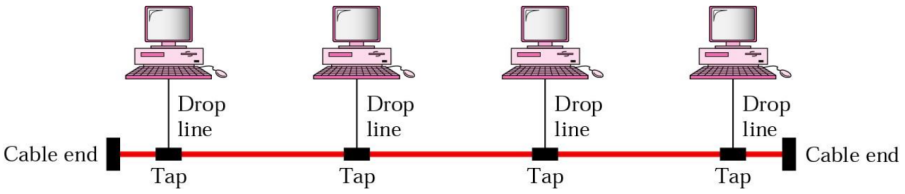


- A variation of star topology
- Nodes are connected to central hub (also called active hub) that controls traffic
- There are secondary hub (can be passive or active) that is connected to the central hub
- Term:
  - Active hub: AKA switch, provides physical connection, capable of amplifying the signal via repeater
  - Passive hub: only provides physical connection

▼ Advantages

- Same as star topology
- Secondary hub allows:
  - more devices to be attached
  - the network to isolate and prioritise communication

▼ Bus



- multipoint
- One long cable acts as a backbone to link all teh devices in the network
- Nodes are connected to the backbone via drop-lines and taps

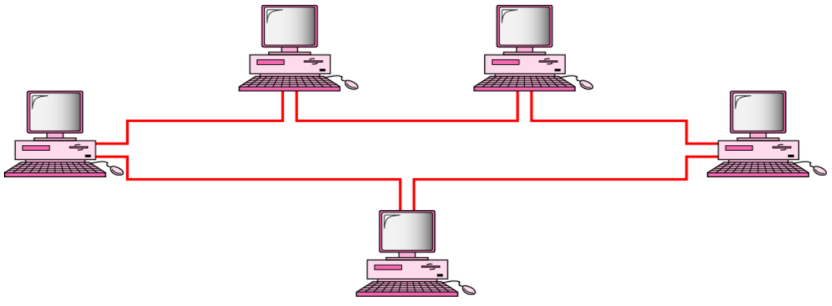
▼ Advantages

- Easy for installation

▼ Disadvantages of bus topology

- Difficult for a reconfiguration and fault isolation (identifying where is the error point)
- Signal reflection at the taps can cause degradation in quality

▼ Ring



- Each device has a dedicated p2p link with adjacent node
- signal passed along the ring in one direction
- Each devices incorporates a repeater

▼ Advantages

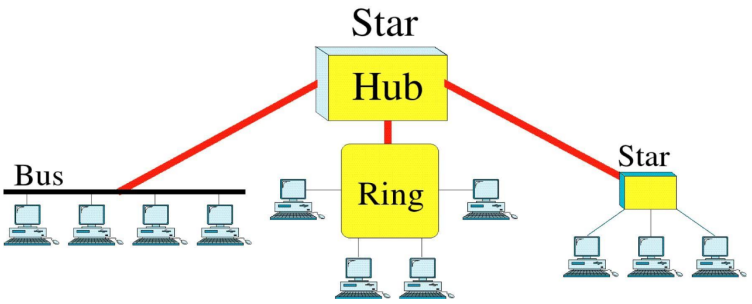
- Easy to install and configure

▼ Disadvantages

- unidirectional traffic

▼ What topologies are used in a larger settings?

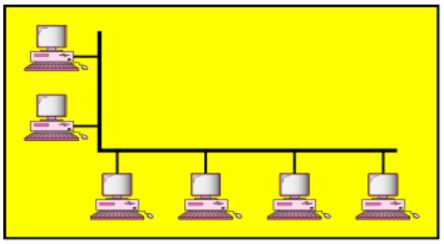
Hybrid topology



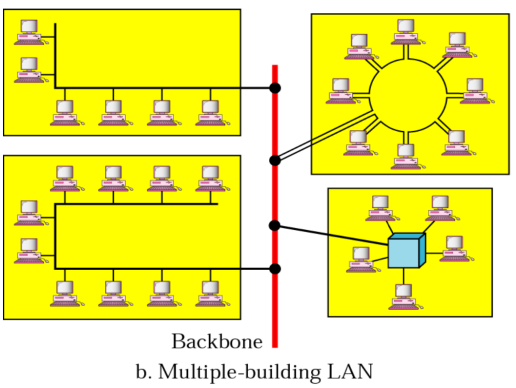
- Combinations of several topologies as subnetworks linked together in a larger topology
- Several topology can be connected to each other via a central controller in a star topology

▼ What are the categories of Networks?

▼ LAN (Local Area Network)



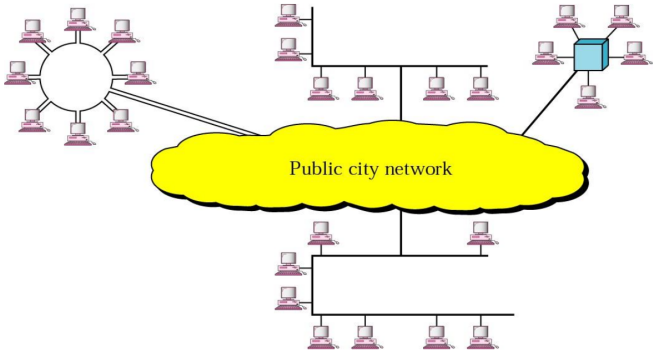
a. Single-building LAN



b. Multiple-building LAN

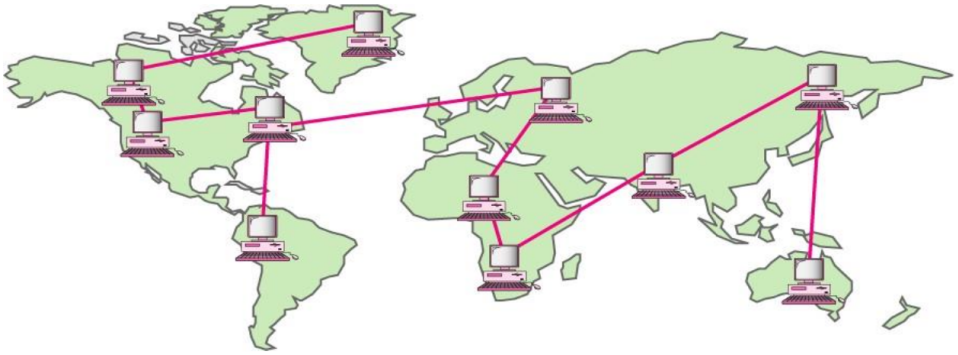
- Privately owned
- Link devices in a single office, building or campus
- limited to a few kilometres
- Commonly uses bus, ring, star topology
- Data rate/ speed up to 100Mbps

▼ MAN (Metropolitan Area Network)



- Designed to extend over an entire city
- eg: cable TV network, phone network

▼ WAN (Wide Area Network)



- Provides long distance transmission of data, voice, image, video conference over large geographic areas
- May utilised as public, leased or private communication equipment

- Enterprise network - refer to WAN that wholly owned and used by a single company

▼ Chapter 2: Application Layer

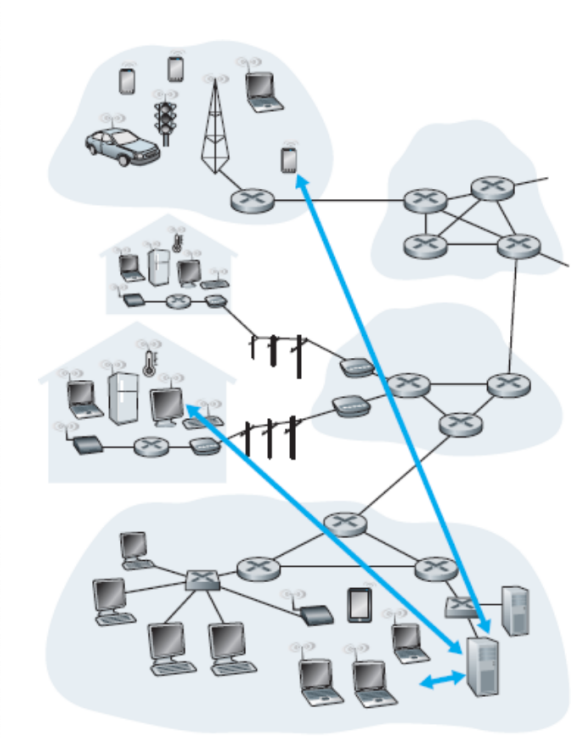
▼ 2.1 Principles of Network Applications

▼ 2.1.1 What is an application architecture?

- Structural design and organisation of software app, defines the overall conceptual model for building an application

▼ What are 2 most famous app architecture?

▼ Client-server architecture

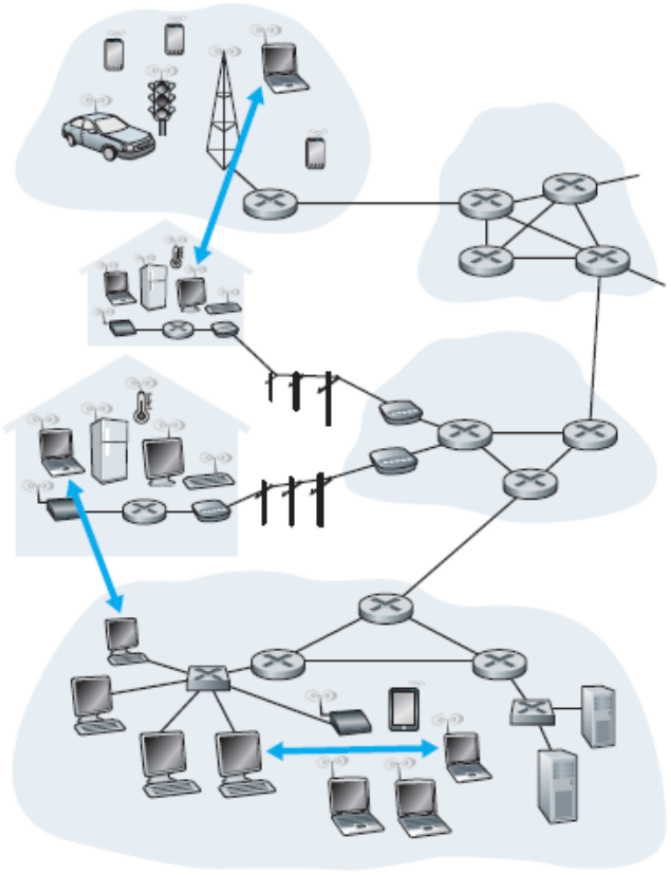


a. Client-server architecture

- There is server and client, client will request from the server
- The server has a fixed IP address
- eg:
  - Web
  - FTP
  - Telnet

    - e-mail
- Since a single server is incapable of tonnes of request, there is a data centre that houses thousands of server

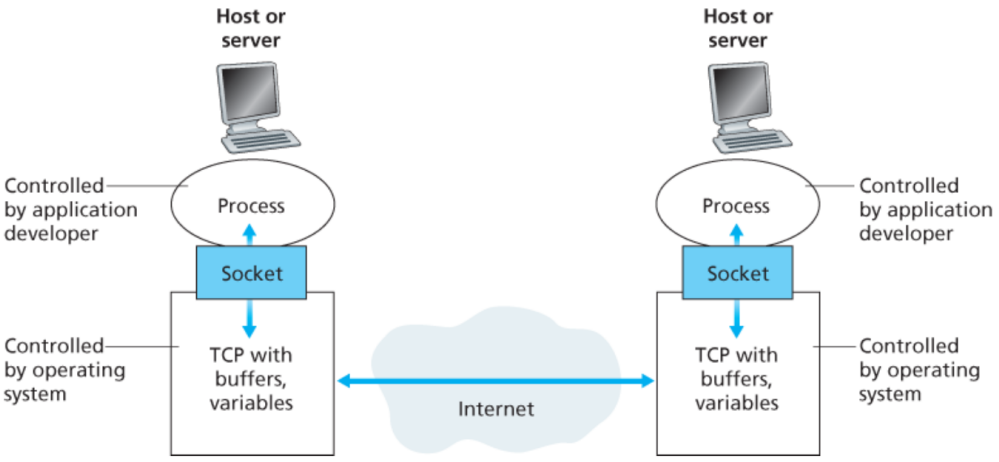
▼ Peer-to-peer architecture (P2P)



b. Peer-to-peer architecture

- Doesn’t require server, since the hosts (aka peer) are directly connected without passing a server.
- eg:
    - BitTorrent
    - Skype
- Self-scalability
- Cost effective
- Face security, performance, reliability, performance challenges

▼ 2.1.2 How do the app on different end systems communicate?



▼ It communicate via *process*

- In this context, *process* that initiate the communication is the client, the *process* that wait to begin session is the server.

▼ What is process?

- A software component that runs on a device to provide network functionality
- It communicate by exchanging messages (in application layer, the data is encapsulated into a packet called message before encapsulated at transport layer into segment)
  - sending *process* creates, sends message to the network
  - receiving *process* receives the message, may send respond

▼ A process send and receive messages through socket

▼ What is a socket?

- An interface between application layer and transport layer in a host — also referred as Application Programming Interface (API) between the app an the network
- Analogy:
  - Process is a house, socket is a door, each component entering or exiting the *process* must

go through the door (socket)

- When an app wants to communicate over a network, it'll create a socket
- It act as a communicator between the app and the network

- In order for the message to arrive at the correct place, 2 address must be added:
  - IP address — to identify the host
  - port number — to identify which app the packet is designated to
    - Popular apps've been assigned specific port numbers
      - Http (web server): 80
      - Https (web server): 443
      - SMTP (email): 25

▼ 2.1.3 What are the services that transport-layer protocol provides to application?

▼ Reliable data transfer

- Packet can get loss during the transmission
- Some app — such as e-mail, file transfer, financial app — cant tolerate data loss
- Some app, mostly multimedia app can tolerate some amount of data loss — its called loss-tolerant applications

▼ Throughput

- Some transport protocol allow for a throughput request of r bits/sec — it'll ensure the throughput'll be at least r bits/sec
- App that have throughput requirements are called bandwidth-sensitive applications — mostly multimedia app
- App that doesn't have throughput requirements are called elastic applications — eg: email, file transfer, web transfer

▼ Timing

- Each bits must arrive no more than x msecs later
- Useful for real-time app, such as internet telephony, multiplayer games

▼ Security

- In the sending host, a transport protocol can encrypt data, then decrypt it at receiving host

▼ 2.1.4 What are the transport service provided by the Internet?

▼ Transmission Control Protocol (TCP)

- A connection-oriented service
  - Before the application-level messages are transmitted, the client and server'll exchange transport-layer control info.
  - It is called the three-way handshake
- Reliable data transfer service
  - The TCP'll ensure all data are transferred with no corrupted or missing data
- Congestion-control mechanism
  - Will decrease the size of segments if there is congestion in network — for the general welfare of the internet rather
- Enhanced version of TCP, Secure Socket Layer (SSL)
  - Same process with TCP, but provides more security features:
    - encryption
    - data integrity
    - end-point authentication
  - When an app uses SSL, it'll pass the cleartext data to SSL socket, SSL'll encrypt, pass to TCP socket
  - The receiving end'll receive the segment in the TCP socket, pass to SSL socket to decrypt the data

▼ User Datagram Protocol (UDP)

- Connectionless

- Unreliable — no guarantee of data arrival, data may arrive out of order
- no congestion-control mechanism

Application	Application-Layer Protocol	Underlying Transport Protocol
Electronic mail	SMTP [RFC 5321]	TCP
Remote terminal access	Telnet [RFC 854]	TCP
Web	HTTP [RFC 2616]	TCP
File transfer	FTP [RFC 959]	TCP
Streaming multimedia	HTTP (e.g., YouTube)	TCP
Internet telephony	SIP [RFC 3261], RTP [RFC 3550], or proprietary (e.g., Skype)	UDP or TCP

Web applications the protocol used

▼ 2.1.5 What are the responsibility of application-layer protocols?

- It defines:
  - Type of messages exchanged; request, response messages
  - Syntax of messages types; the fields in the message, how the fields arranged
  - The meaning of the information in the fields
  - Rules on when and how a *process* sends messages and responds to messages

▼ 2.2 The Web and HTTP

▼ 2.2.1 Overview of HTTP

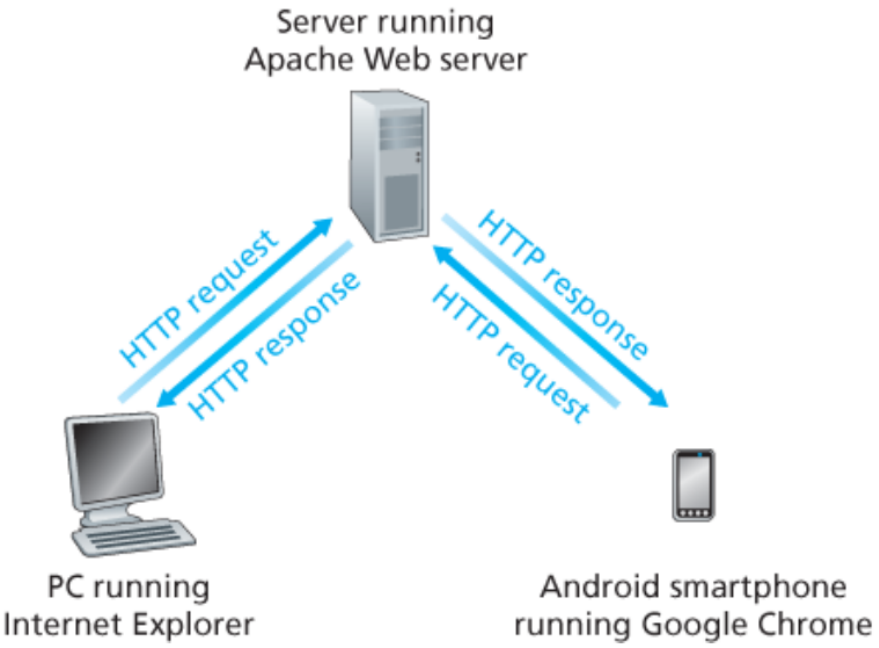
- HTTP is implemented in two programs; client program (web browser) and server program
- A web page consists of objets — objects is a file; JPEG image, video clip
  - URL (Uniform Resource Locator) have two components, hostname and object path

```
http://www.someSchool.edu/someDepartment/picture.gif
```

www.someSchool.edu- hostname

/someDepartment/pitcure.gif- path name

- HTTP defines how web client request web pages form web servers, and how servers transfer Web pages to clients.
- When a user made a request, the browser will send HTTP GET request to the server — the message is sent via TCP



- HTTP is stateless protocol; it doesn't store info about user; it will send the requested resource even if the client already asked for it

▼ 2.2.2 What are 2 types of HTTP connection?

▼ Non-persistent Connection

- TCP connection is closed after the server sends object — the connection doesn't persist for other objects.
- If there is 10 requests, there'll be10 TCP connections

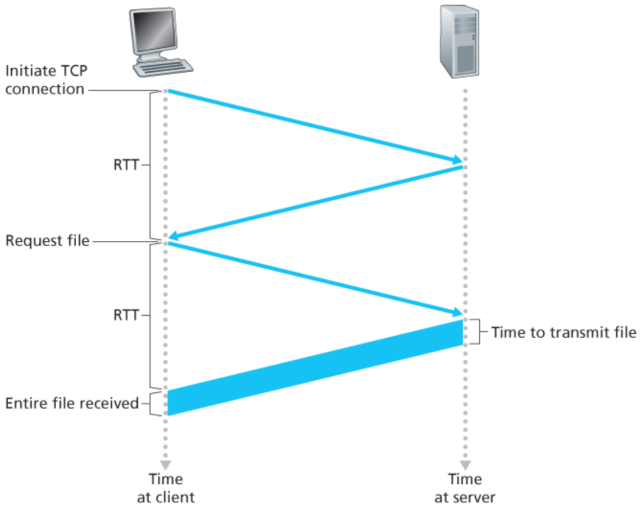
▼ Persistent Connection

- Entire web page can be sent on single TCP connection

▼ What is RTT?

- Round Trip Time, time taken for a packet to travel from client to server then back to client

- When client access a web page, TCP SYN request will be sent, and server will respond with SYN ACK — one RTT
- Then client send HTTP GET and ACK to server, then server sends HTML file to client — one RTT

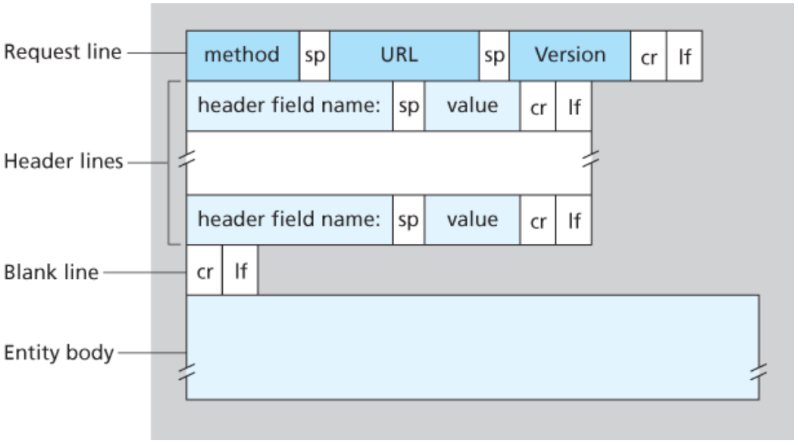


▼ 2.2.3 HTTP Message Format

▼ What are two types of HTTP messages?

▼ HTTP Request

▼ General format of HTTP request message:



▼ eg:

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

▼ First line: Request line, has three fields:

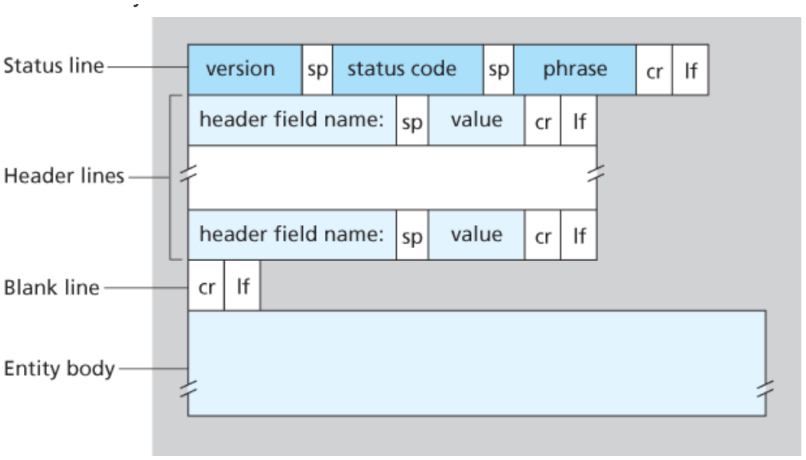
- method (GET)
  - can be GET, POST, HEAD, PUT, DELETE
  - GET is used when the browser request an object
- URL (/somedir/page.html)
  - location of requested object
- HTTP version
  - The version of HTTP that the browser use

▼ Subsequent line: Header line:

- Host
  - the host where the object resides — required by Web proxy caches
- Connection: close
  - browser tell the server it doesn't want persistent connection; it want the server to close connection after sending the requested object
- User-agent
  - the browser that makes the request — server can send different version of pages to different browser
- Accept-language
  - language that the user prefer

▼ HTTP Response

▼ General format of HTTP response message



- Common status code:
  - 200 OK: Request succeed, info is returned in the response
  - 301 Moved Permanently: Requested object has been moved; new URL is specified in *Location:* header of the response message. The client browser will automatically retrieve it
  - 400 Bad Request: Request can’t be understood by the server
  - 404 Not found: Requested docs does not exist on the server
  - 505 HTTP Version Not Supported: The requested HTTP protocol version is not supported by the server

▼ eg:

```
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 ...)
```

▼ First line: Status line

- Protocol version
- Status code
- Status message
  - 200 OK — server is found, sending the requested object

▼ Six header lines

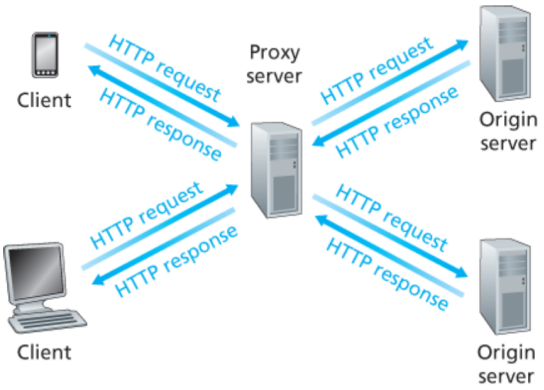
- Connection: close
- Date
  - time where HTTP response is created
- Server
- Last-Modified
  - Useful for object caching
- Content-Length
  - number of bytes in the object being sent
- Content-Type

▼ 2.2.5 Web Caching

▼ What is web cache?

- AKA proxy server, a network entity that keeps copies of recently requested objects of an origin Web server

▼ How does it work?



1. The browser'll establish TCP connection to Proxy server, sends HTTP request
2. The server'll check if it have the copy, if it have, it'll send HTTP response message to client
3. If not, it'll send HTTP request to the original server, get the object, send to client
4. It'll save a copy

- A web cache is typically purchased and installed by ISP

▼ Why is web cache important?

- It can reduce response time
- It can reduce traffic on the network

▼ What if the object in the cache is outdated?

- HTTP has a mechanism to allows cache to verify the object is up-to-date, called conditional GET
- A HTTP request is a conditional GET message if:
  - It uses GET method
  - It includes *If-Modified-Since* header line

▼ How does it work?

1. A browser send HTTP request to cache server

```
GET /fruit/kiwi.gif HTTP/1.1
Host: www.exotiquecuisine.com
```

2. Cache server respond to the browser

```
HTTP/1.1 200 OK
Date: Sat, 3 Oct 2015 15:39:29
Server: Apache/1.3.0 (Unix)
Last-Modified: Wed, 9 Sep 2015 09:23:24
Content-Type: image/gif

(data data data data data ...)
```

3. A few weeks later, when a browser request for the same data, the cache performs up-to-date check by issuing a conditional GET

```
GET /fruit/kiwi.gif HTTP/1.1
Host: www.exotiquecuisine.com
If-modified-since: Wed, 9 Sep 2015 09:23:24
```

   - It request the object only if the object had been modified since the last request

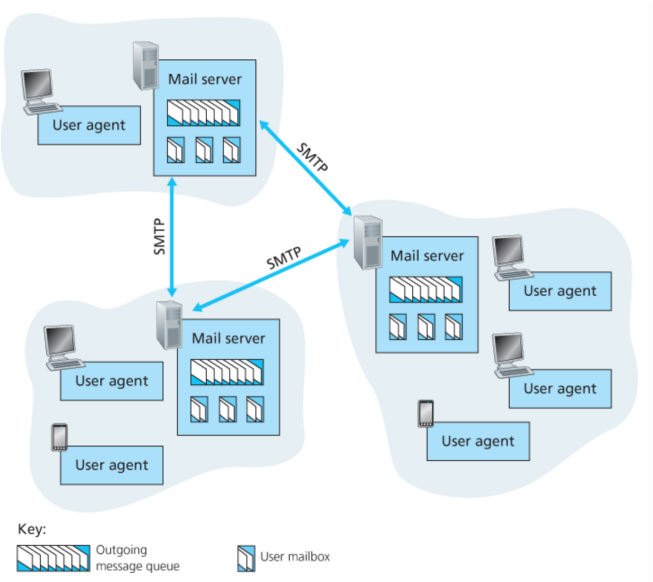
4. If not modified, the original server'll respond with code 304 NOT MODIFIED, without the copy of the object

```
HTTP/1.1 304 Not Modified
Date: Sat, 10 Oct 2015 15:39:29
Server: Apache/1.3.0 (Unix)

(empty entity body)
```

▼ 2.3 Electronic Mail in the Internet

▼ How does it work (overview)?

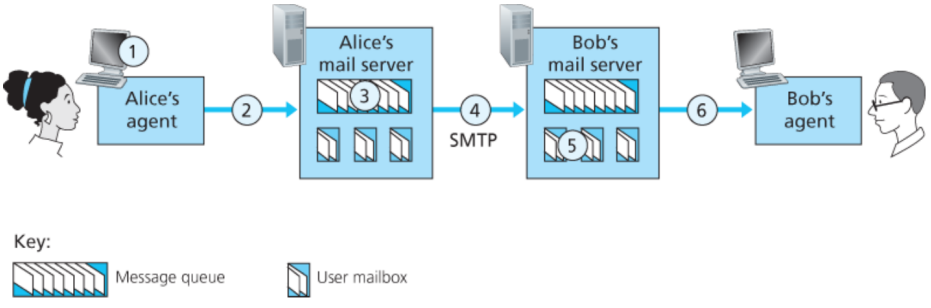


- There are 3 major components; user agents, mail servers, Simple Mail Transfer Protocol (SMTP)
- When A finished composing her email, her user agent (MS Outlook, Gmail etc) sends it to her mail server, where it is placed in the mail server's outgoing message queue.
- When B wants to read it, his user agent retrieves the email from his mailbox in his email server

▼ 2.3.1 What is Simple Mail Transfer Protocol SMTP?

- Protocol to transfer messages from sender's mail servers to the recipient's mail server

▼ How does it work?



1. Alice compose a message and provide Bob's e-mail address, instruct the user agent to send the message

2. Alice’s user agent sends the message to her mail server, it is placed in a message queue
3. The client side of SMTP on Alice’s mail server opens a TCP connection to SMTP server on Bob’s mail server
4. After SMTP handshakes, the SMTP client send Alice’s message into the TCP connection
5. At Bob’s mail server, the SMTP receives the message, stores it in Bob’s mailbox
6. Bob use his user agent to read the message form his mail server

- The TCP connection is direct, no intermediate mail server involved

▼ Example of messages exchanged between SMTP client (C) and SMTP server (S)

```
S:  220 hamburger.edu
C:  HELO crepes.fr
S:  250 Hello crepes.fr, pleased to meet you
```

```
C:  MAIL FROM: <alice@crepes.fr>
S:  250 alice@crepes.fr ... Sender ok
C:  RCPT TO: <bob@hamburger.edu>
S:  250 bob@hamburger.edu ... Recipient ok
C:  DATA
S:  354 Enter mail, end with "." on a line by itself
C:  Do you like ketchup?
C:  How about pickles?
C:  .
S:  250 Message accepted for delivery
C:  QUIT
S:  221 hamburger.edu closing connection
```

▼ 2.3.3 How are mail message formatted?

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

▼ How can a user access the email from the mail server?

- By mail access protocol:
  - ▼ POP3
    - Begins when client open TCP connection to the server on port 110
    - ▼ Then POP3 will go through three phases:
      - Authorisation - sends username, password

```
telnet mailServer 110
+OK POP3 server ready
user bob
+OK
pass hungry
+OK user successfully logged on
```

      - Transaction - retrieve message — user can mark message for deletion

```
C: list
S: 1 498
S: 2 912
```

```
S: .
C: retr 1
S: (blah blah ...
S: ..................
S: ..........blah)
S: .
C: dele 1
C: retr 2
S: (blah blah ...
S: ..................
S: ..........blah)
S: .
C: dele 2
C: quit
S: +OK POP3 server signing off
```

- Update - occurs when client issued *quit* command — the mark for deleted will be deleted

▼ There are two mode

- download-and-delete - after retrieved, it'll be deleted
- download-and-keep - the email stay on the server after download

▼ IMAP

- Associate message with a folder
- User can create folders
- Users can search folders for message matching specific criteria
- Users can obtain specific component of message — useful when there is low bandwidth

▼ By using Web-Based E-Mail

- When accessing email, the HTTP used instead of POP3/IMAP
- When sending email, the email is sent over to mail server by HTTP

▼ 2.4 Domain Name Server (DNS)

- Host can be identified via:- hostname, IP address

▼ What is DNS?

- To translate hostname into IP addresses
- Runs over UDP, port 53

▼ How does DNS work?

1. A user enter hostname in the browser, the browser passes the hostname to the client side of DNS app
2. The DNS client sends query containing the hostname to a DNS server
3. The DNS client receives a reply with the IP address, send to the browser
4. The browser can initiate TCP connection to the IP address on port 80 (HTTP)/ 443 (HTTPS)

▼ 2.4.1 What are the important services other than translating hostname?

▼ Host aliasing

- If a hostname have a complicated name, *relay1.west-coast.enterprise.com* — canonical hostname —, user can still refer to it when typing their aliases such as *enterprise.com.*

▼ Mail server aliasing

- Mail server hostname can be complicated, *@relay1.west-coast.com,* instead of *@yahoo.com,* DNS can be invoked by mail application to obtain the original hostname instead of the one written by the user

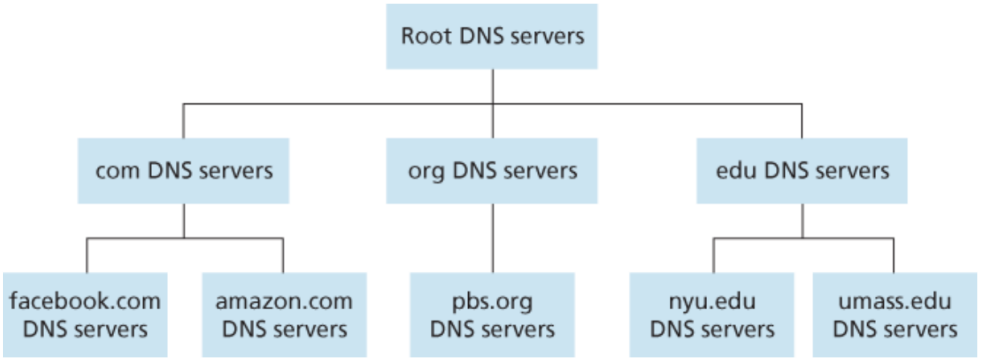
▼ Local distribution

- For a very busy sites such as *cnn.com,* it can be replicated over multiple server, with different IP addresses. When a browser

invoke DNS, it'll give the IP address of different server for different requests

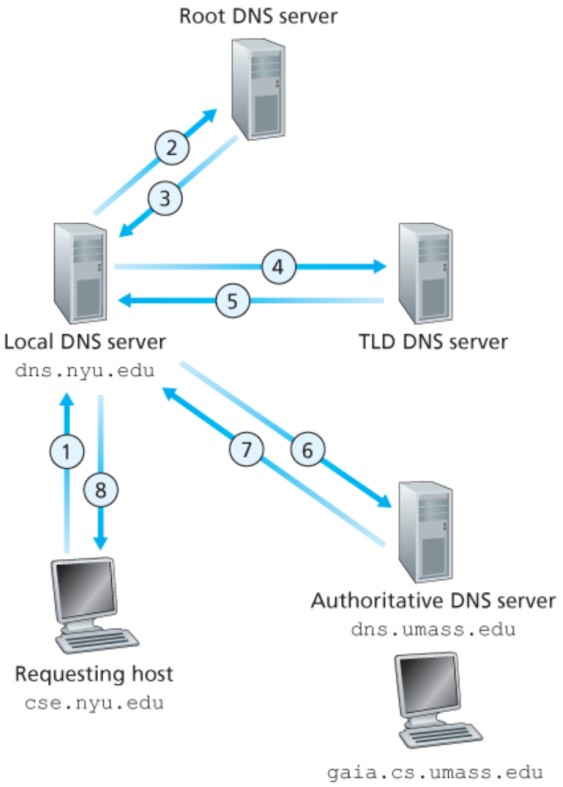
▼ 2.4.2 Overview of How DNS Works

- Since it is ineffective to have only one DNS server for the whole world, there is a concept of Hierarchical Database:



- Local DNS server
  - Typically run by each ISP
  - When host makes a DNS query, the local DNS will forward to above DNS
- Root DNS server - provides IP address for top level domain, TLD
- Top-Level Domain (TLD) server - provides IP address of authoritative DNS servers
  - Each TLD (.com, .edu, .gov) has its own server
- Authoritative DNS server
  - Each organisation with publicly accessible hosts must provide publicly accessible DNS records.

▼ Example



1. cse.nyu.edu desires IP address of gaia.cs.umass.edu, it query message to local dns server, dns.nyu.edu.
2. local dns server forwards the query to root DNS server
3. Root DNS server takes not of the *.edu* suffix, return to the local DNS server list of TLD server responsible for the .edu suffix.
4. Local DNs server resend the query to the TLD server, TLD server takes note of umass.edu, respond back the authoritative DNS server for University of Massachusetts. (the local DNS can cache the TLD IP address, so no need to request it to the Root DNS in the future)
5. Local DNS resend query to authoritative DNS, and get the IP for gaia.cs.umass.edu. (it can cache the IP address for further requests)

▼ 2.4.3 DNS Records and Messages

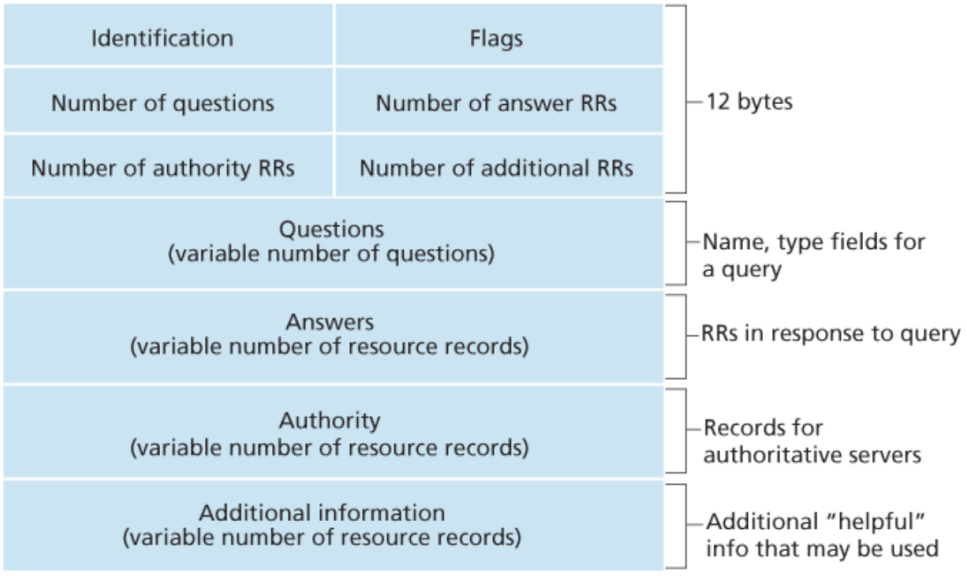
▼ How does DNS store the hostname-to-IP address files?

- By using Resource Records (RRS)

(Name, Value, Type, TTL)

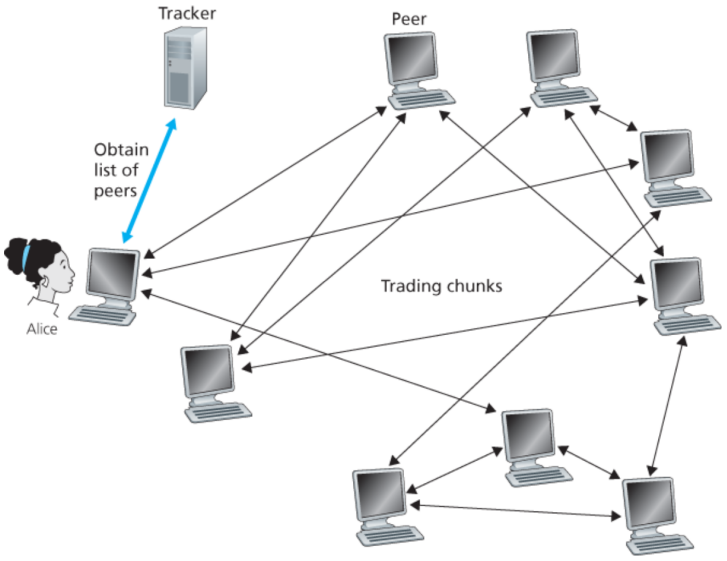
- Type:
  - A: Name = hostname, Value = hostname's IP address
    - If the DNS server is authoritative, or it has been cached
  - NS: Name = domain, Value = hostname of authoritative DNS server for the domain — it'll also have type A RRS for the authoritative DNS.
  - CNAME: Name = alias, Value = canonical name (original name)
  - MX: Name = alias, Value = canonical name (same with CNAME, but its for mail server)

▼ How is the form of DSN messages look like?



▼ 2.5 Peer-to-Peer File Distribution

▼ BitTorrent



- Peers that participate in the distribution is called torrent
- Peers in the torrent download equal-size chunks of the file from one another, a chunk typically is 256KBytes.
- When one peer has completed the download, it may leave the torrent or continue to upload chunks — or if they stop halfway and continue

▼ How does it work?

1. When Alice joins a torrent it'll register itself with the tracker, and periodically inform the tracker she's still in the torrent.
2. The tracker will gives list of IP addresses of peers to Alice, Alice attempts to establish TCP connection with the peers
3. Periodically, Alice will ask each peers (over TCP) list of chunks they have, if she has L neighbours, she'll get L lists.
4. Alice will use the *rarest first* — chunk that have the fewest repeated copies among the neighbours— technique to choose which chunk to request — it is to equalise the numbers of copies of each chunk in the torrent
5. She will upload chunk to the 4 neighbours that are supplying data at the highest rate — these peers are called unchoked—, every 10 seconds she'll recalculate the rate and update the peer accordingly

6. Every 30 seconds, she'll randomly pick one additional neighbour (Bob) to send chunks to — the peer is called optimistically choked—
7. Since Alice is sending chunks to Bob, she may become a top sender for Bob, hence he'll send chunks to Alice, and he then may be Alice top sender

- the mechanism is called tit-for-tat

▼ 2.6 Video Streaming and Content Distribution Networks

▼ What are the examples of multimedia services?

- 2.6.1 Internet Video

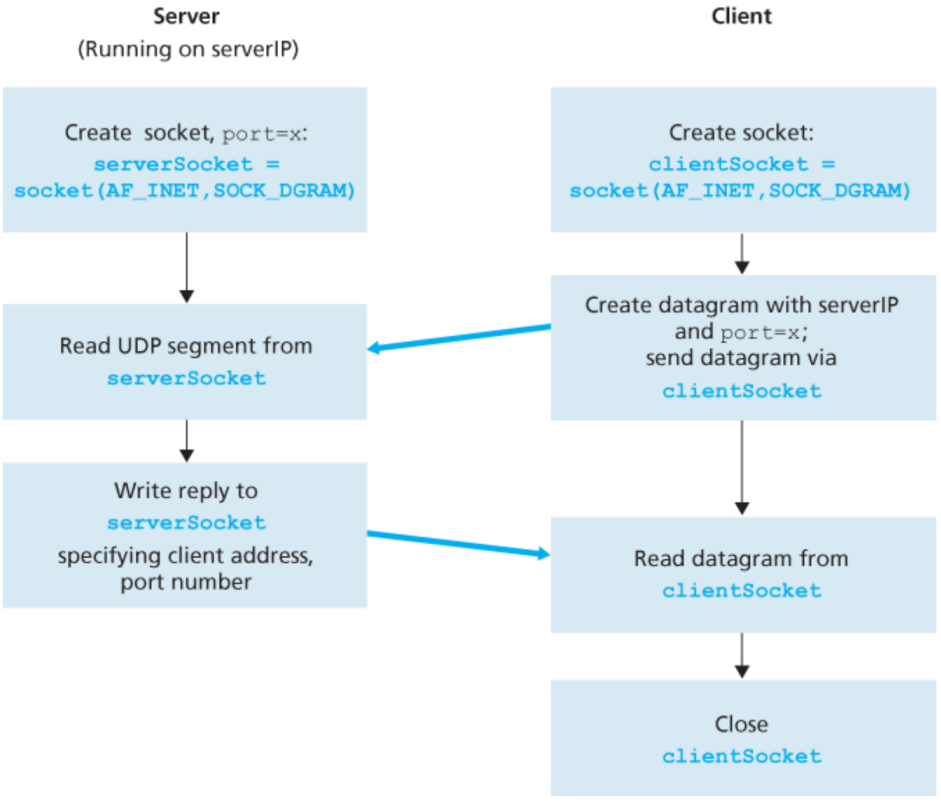
▼ 2.6.2 HTTP streaming and DASH

- Videos are stored in HTTP server
- One problem: all'll receive the same encoding, despite the different bandwidth of the clients
- Hence Dynamic Adaptive Streaming over HTTP, DASH is developed
  - The video is encoded into different version, each version with different quality
  - The client will dynamically requests chunks of video segment of a few seconds length, if the bandwidth is high, the client'll select higher rate version.
  - The selection is done with HTTP GET request

▼ 2.7 Socket Programming: Creating Network Applications

▼ 2.7.1 Socket programming with UDP

- The whole process:



- Socket programming on client side, UDPClient.py:

```
from socket import *
serverName = 'hostname'
serverPort = 12000
```

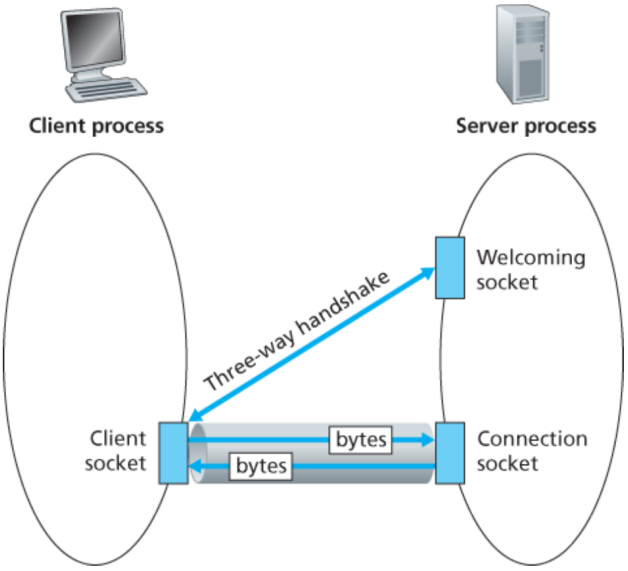
```
clientSocket = socket(AF_INET, SOCK_DGRAM)
message = raw_input('Input lowercase sentence:')
clientSocket.sendto(message.encode(),(serverName, serverPort))
modifiedMessage, serverAddress = clientSocket.recvfrom(2048)
print(modifiedMessage.decode())
clientSocket.close()
```

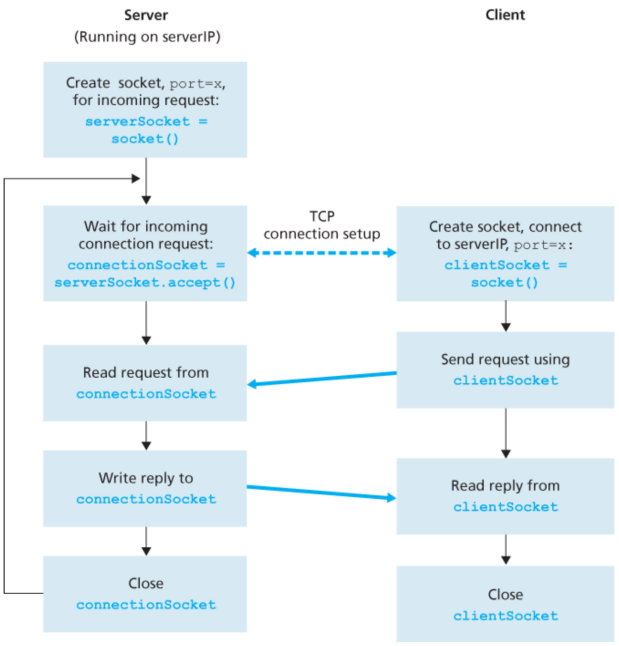
- Socket programming on server side, UDPServer.py:

```
from socket import *
serverPort = 12000
serverSocket = socket(AF_INET, SOCK_DGRAM)
serverSocket.bind(('', serverPort))
print("The server is ready to receive")
while True:
    message, clientAddress = serverSocket.recvfrom(2048)
    modifiedMessage = message.decode().upper()
    serverSocket.sendto(modifiedMessage.encode(), clientAddress)
```

▼ 2.7.2 Socket programming with TCP

- The TCP process





- Socket programming on client side TCPClient.py:

```
from socket import *
serverName = 'servername'
serverPort = 12000
clientSocket = socket(AF_INET, SOCK_STREAM)
clientSocket.connect((serverName, serverPort))
sentence = raw_input('Input lowercase sentence:')
clientSocket.send(sentence.encode())
modifiedSentence = clientSocket.recv(1024)
print('From Server: ', modifiedSentence.decode())
clientSocket.close()
```

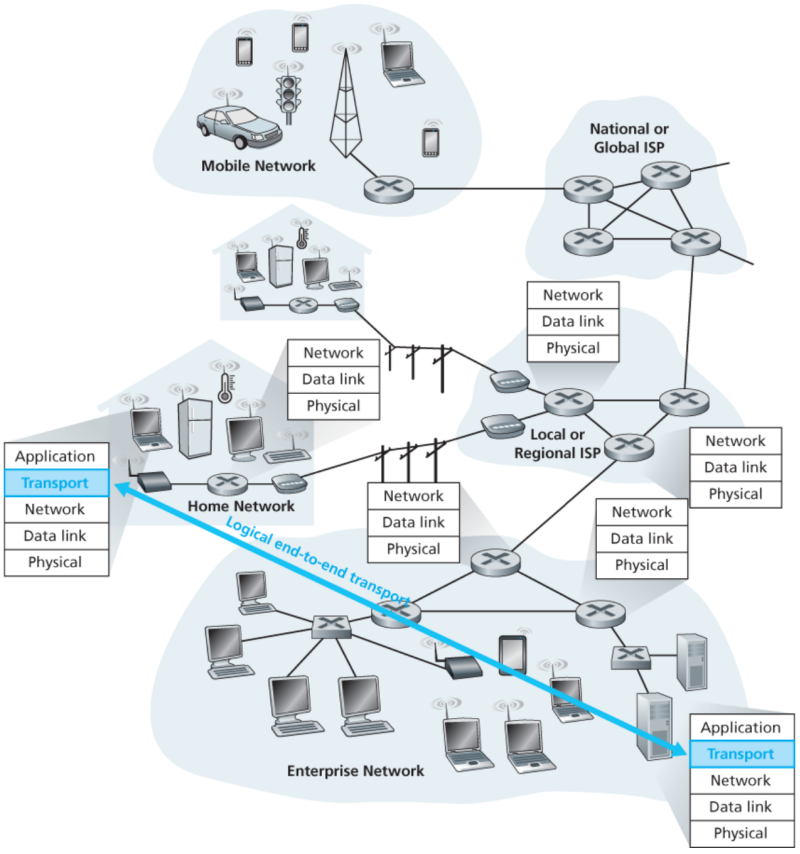
- Socket programming on server side, TCPServer.py:

```
from socket import *
serverPort = 12000
serverSocket = socket(AF_INET, SOCK_STREAM)
serverSocket.bind(('', serverPort))
serverSocket.listen(1)
print('The server is ready to receive')
while True:
    connectionSocket, addr = serverSocket.accept()
    sentence = connectionSocket.recv(1024).decode()
    capitalizedSentence = sentence.upper()
    connectionSocket.send(capitalizedSentence.encode())
    connectionSocket.close()
```

▼ Chapter 3: Transport Layer

▼ 3.1 Introduction and Transport-Layer Services

▼ 3.1.0 Intro



▼ What is transport layer?

- Provides logical communication between app on different hosts — logical communication is the perspective as if the two end hosts is connected directly when in reality there are several nodes in between

▼ How does transport layer works in general?

- On the sending side, the transport layer converts the app-layer messages received from the app layer process into transport-layer packet, aka segments:
  1. Transport layer breaks the messages into several chunks, add a transport-layer header to each chunk
  2. Transport layer passes the segment to the network layer, the segment is encapsulated within a network-layer packet aka datagram, then sent to destination — the network router only act on the network-layer fields of the datagram; they don’t give a damn about the transport-layer segment

3. On the receiving side, the network layer extracts the transport-layer segment from the datagram, send to the transport layer of the receiving host
4. The transport layer process the segment, send to appropriate socket

▼ 3.1.1 Relationship Between Transport and Network Layers

▼ What is the difference between network-layer and transport-layer?

- Transport layer provides logical communication between process running on different hosts
- Network layer provides logical communication between hosts

▼ Analogy of network-layer and transport layer

- In this analogy:
  - kids = application process
  - house = host
  - Bill and James = transport layer
  - Postal service = network layer
- There is 2 house, one at Johor, one at Kedah
- Each house houses multiple kids who like to exchange letters to each other
- Each day, one of the kids on each house, say Bill and James will collect the mails form the kids and send it to the postal service.
- And when the letters arrive at each house, Bill and James will distribute the letters to the respected kids.
- Bill and James does not involve in sending the letter to the each house physically, they only drop it at the postal service and let them do the job
- The postal service only involve in sending the letters to the house, but not involved in handing it to each kids

▼ 3.2 Multiplexing and Demultiplexing

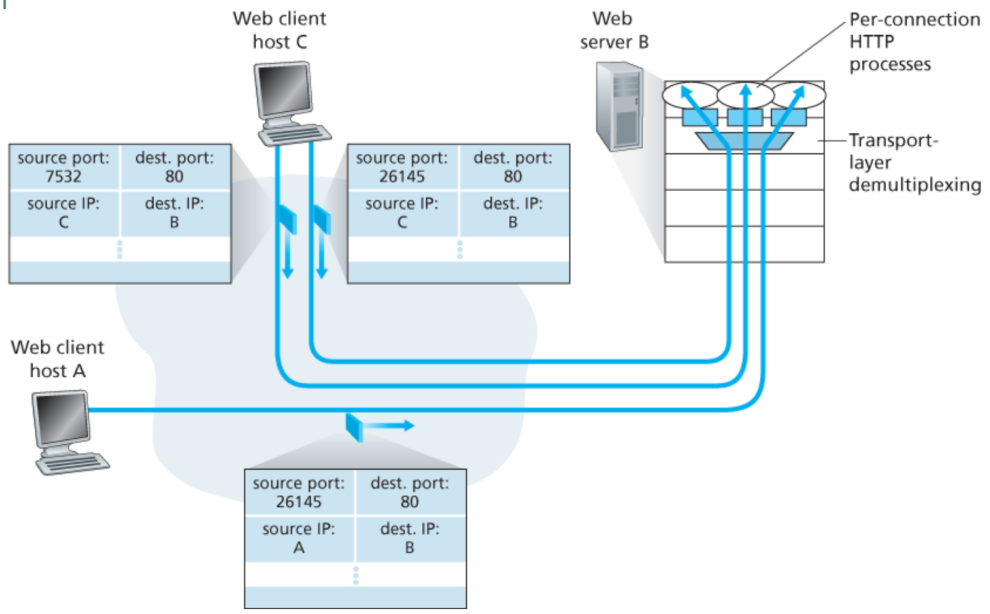


Figure 3.5 Two clients, using the same destination port number (80) to communicate with the same Web server application

▼ What is multiplexing and demultiplexing process?

- Extending the host-to-host delivery service by the network layer to a process-to-process delivery service for applications running on the hosts

▼ What is multiplexing?

- Process of gathering data chunks at the source host from different sockets, encapsulating each data chunk with header information to create segments, then passing it to the network layer
- From Bill and James analogy:
  - Bill and James gather the letters from all kids, pass to the postal service

▼ What is demultiplexing?

- Process of delivering data in a transport-layer segment to the correct socket
- Bill and James receives the mail from postal service, observing to whom each later is for, gives it to the respected kids

▼ How does the multiplexing and demultiplexing process work?

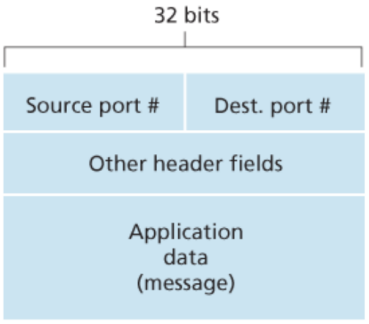


Figure 3.3 Source and destination port-number fields in a transport-layer segment

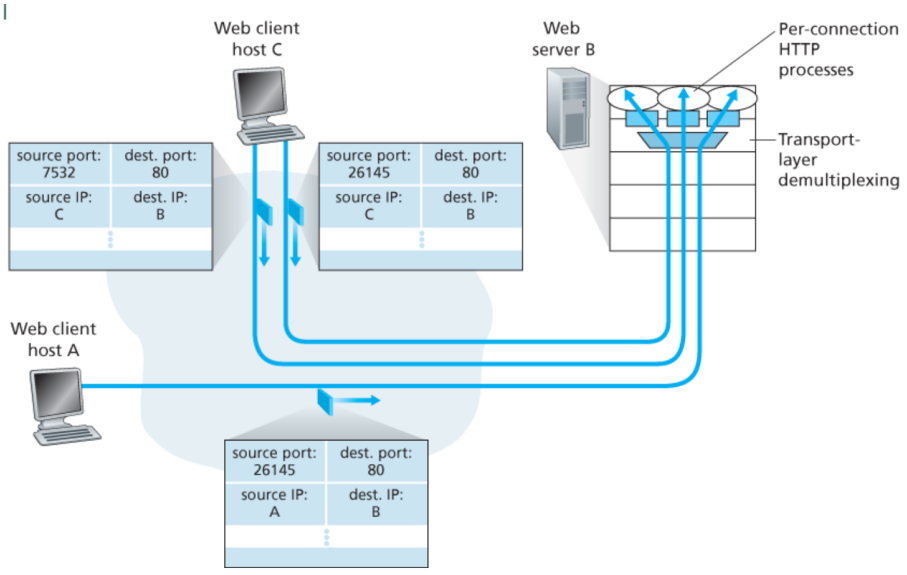
- Each socket have its identifier, called port number — 16-bit number, 0 - 1023 are called well-known port number, reserved by well-known app protocol
- Source port is there incase the destination host want to respond

▼ What are two types of multiplexing and demultiplexing?

▼ Connectionless — UDP

- When segment arrive, the transport layer examines the destination port number, directs the segment to the corresponding socket.

▼ Connection-oriented — TCP



- TCP server has a welcoming socket that waits dor connection-establishment (SYN request) on port 12000

- The TCP clients created a socket, send a connection request — a TCP segment with destination port 12000 — segment with the lines:

```
clientSocket = socket(AF_INET, SOCK_STREAM)
               clientSocket.connect((serverName,12000))
```

- When the server receives the connection-request segments, it locates the server process that's waiting to accept a connection on port 12000. The server then creates a new socket:

```
connectionSocket, addr = serverSocket.accept()
```

- The transport layer at the server notes the following values in the segment:
  - Source port number
  - Source IP address
  - Destination port
  - Destination IP address
- All segments matching these values'll be demultiplexed into the socket

▼ 3.3 Connectionless Transport: UDP

▼ Why use UDP over TCP?

Application	Application-Layer Protocol	Underlying Transport Protocol
Electronic mail	SMTP	TCP
Remote terminal access	Telnet	TCP
Web	HTTP	TCP
File transfer	FTP	TCP
Remote file server	NFS	Typically UDP
Streaming multimedia	typically proprietary	UDP or TCP
Internet telephony	typically proprietary	UDP or TCP
Network management	SNMP	Typically UDP
Name translation	DNS	Typically UDP

Figure 3.6 Popular Internet applications and their underlying transport protocols

▼ Finer application-level control over what data is sent, and when

- UDP segment immediately passed down to the network layer
- TCP segment has a congestion-control mechanism, will throttle when the link is congested
- TCP resend the segment until the receiver send an ACK response

▼ No connection establishment

- UDP will send the segment immediately without establishing connection with the receiver, no delay
- Chrome uses QUIC (Quick UDP Internet Connection), which uses UDP as its protocol and implements reliability in an application-layer protocol on top of UDP.

▼ No connection state

- TCP maintains connection state in the end system — receives, send buffers, congestion-control parameters, sequence and acknowledgement number parameters for reliable data service, congestion control mechanism.
- UDP doesn’t track any of those, hence UDP server can support much more active client than TCP

▼ Small packet header overhead

- TCP segment has 20 bytes header

- UDP segment has 8 bytes heder

▼ 3.3.1 UDP Segment Structure

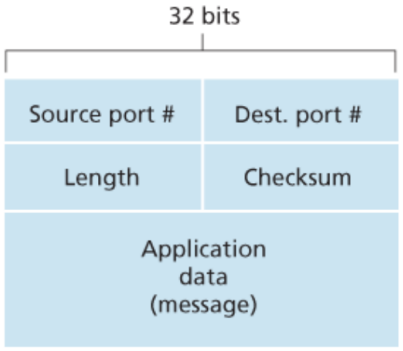


Figure 3.7 UDP segment structure

- Length - length of whole segment (header + payload)

▼ What is Checksum?

- Provides error detection — to determine if bits in the UDP segment (only the header, not the payload) have been altered (noise in the links or error while stored in a router)

▼ How to calculate checksum?

0110011001100000

0101010101010101

1000111100001100

1. Add the 16 bits from the header

$$\begin{array}{r} 1011101110110101 \\ +\ 1000111100001100 \\ \hline 0100101011000010 \end{array}$$

2. Make the 1s compliment (turn 1 → 0 , 0→1)

- 0100101011000010 → 1011010100111101 (this is the checksum)

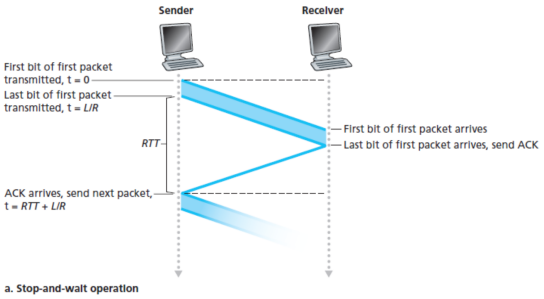
3. At the receiver end, all the four 16-bit words are added, if no errors, the sum will be 1111111111111111
    - If there is error, UDP will simply discard the damaged segment or send it to the application with a warning

▼ Why UDP has a checksum when many link-layer protocols provide error checking?

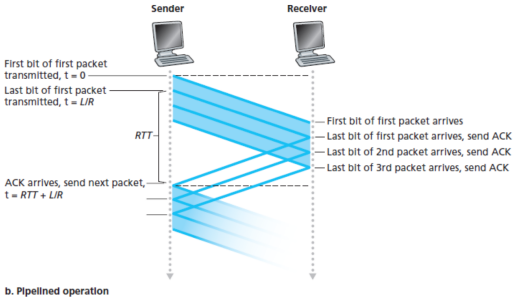
- Not all links between source and destination provide error checking.
- Even if segments are correctly transferred across a link, there may be bit errors when a segment is stored in a router's memory
- Applying the end-to-end design framework in computer networking — reliability and security features must reside in the communicating end nodes of the network

▼ 3.4 Principle of Reliable Data Transfer

▼ 3.4.2 Pipelined Reliable Data Transfer Protocol



a. Stop-and-wait operation



b. Pipelined operation

- In stop-and-wait, the ack from the receiver must be received before the sender send the next package

- In pipelining, the sender is allowed to send multiple packets without waiting for the acknowledgement
- Pipelining consequences:
  - The range of sequence number must be increased, since each in-transit packet (unacknowledged) must have unique sequence number
  - The sender and receiver may have to buffer more than one packet

▼ What are 2 approaches toward pipelined error recovery?

▼ Go-Back-N (GBN)

▼ The sender is allowed to transmit multiple packets without waiting for an acknowledgement, but no more than the max allowable number, N

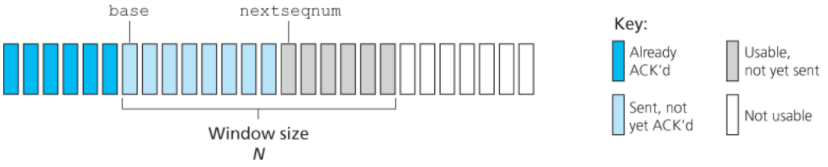


Figure 3.19 Sender's view of sequence numbers in Go-Back-N

- The number of packet transmitted at a time can't exceeds the window size
- The window size will move accordingly when the last *sent, not yet ACK'd* packet is ACK'd, hence it's also called as a *sliding-window protocol*

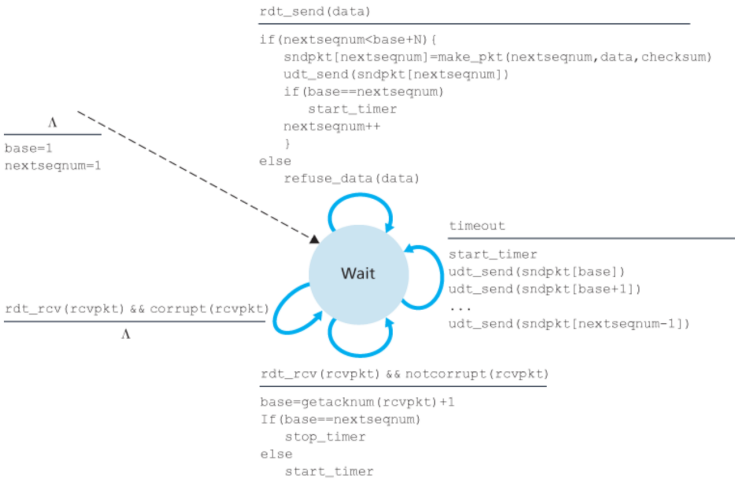


Figure 3.20 Extended FSM description of the GBN sender

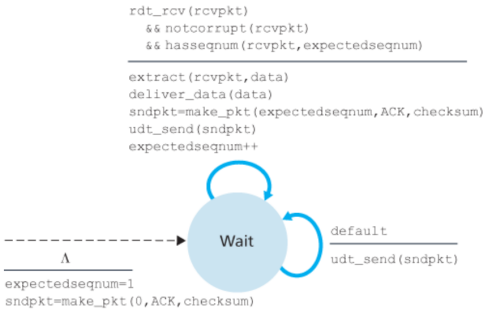


Figure 3.21 Extended FSM description of the GBN receiver

▼ What are 3 types of event a GBN sender must respond?

- ▼ Invocation from above
  - When the upper layer (app) want to send packet, the sender will check the window, if full it'll return back to the upper layer.
- ▼ Receipt of an ACK
  - The acknowledgement for a pakcet with sequence number n'll be a cumulative acknowledgement, indicating all packet with a seq numbet up to and including n have been received correctly at the receiver
- ▼ A timeout event
  - When a packet is sent but no ACK response for a specific time, the packet'll be resended

▼ What are the actions of the receiver?

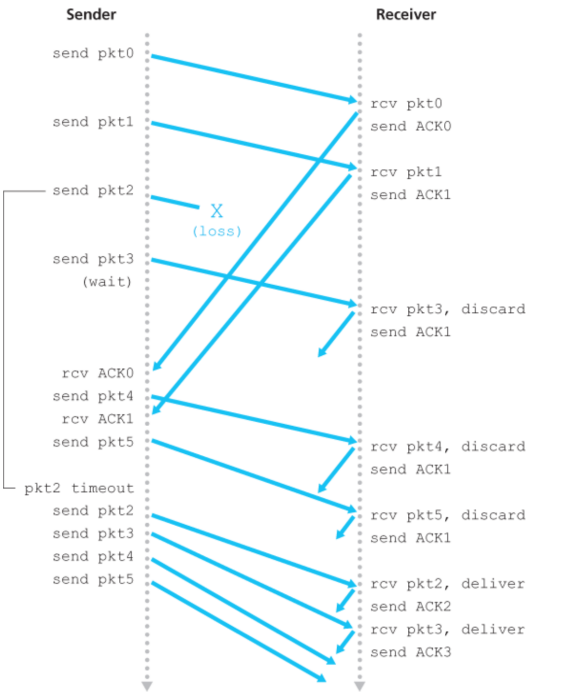


Figure 3.22 Go-Back-N in operation

- The receiver will only accept packets in order, if one packet arrived out of the order, it'll be discarded.
- Allow for a simple receiver buffering

▼ What are the disadvantages if GBN?

- A single packet error can cause retransmission of many packet

▼ Selective Repeat (SR)

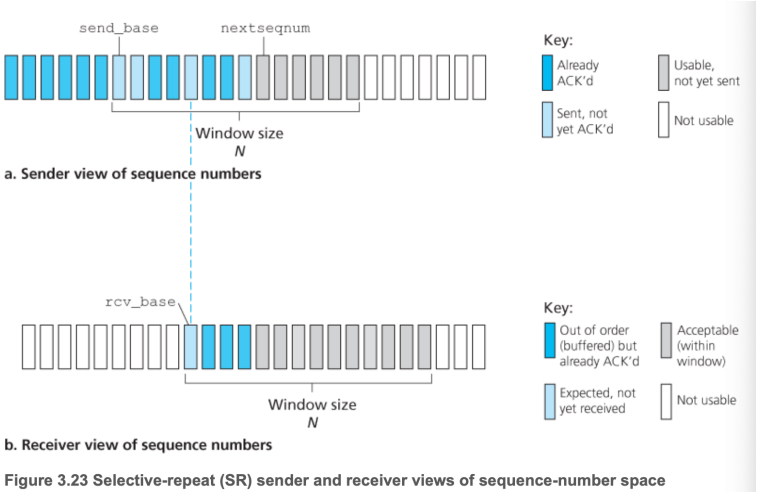


Figure 3.23 Selective-repeat (SR) sender and receiver views of sequence-number space

- Unlike GBN, out-of-order packets will buffer until its correct order

▼ What are the actions of the sender?

- Data received from above - If window size is not enough, packet will. be returned to upper layer or buffered — same as GBN
- Timeout. Each packet has its own logical timer
- ACK received. When ACK received the window base is moved, if there's unACKed packet in the window, it'll be retransmitted

▼ What are the actions of the receiver?

- *Packet with sequence number, rcv_base ≤ SN ≤ rcv_base+N-1 is correctly received* - selective ACK is returned to the sender
- *Packet with sequence number, rcv_base-N ≤ SN ≤ rcv_base-1 is correctly received*. ACK must be generated, to make sure no retransmission by the sender
- *Otherwise* - ignore the packet

▼ 3.5 Connection-Oriented Transport: TCP

▼ 3.5.1 What is TCP?

- The connection is not end-to-end, it's logical
- It's a full-duplex service — segments can flow from A to B and vice versa simultaneously
- It's point-to-point — single sender and receiver, cant multicasting (send data to multiple receivers in a single send operation)

▼ How does hosts initiate a TCP conncetion?

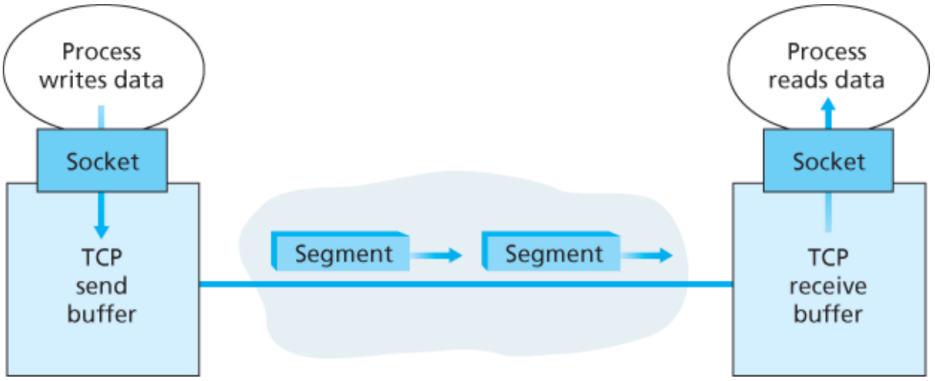


Figure 3.28 TCP send and receive buffers

- 3-way-handshake:
  - Sender send SYN request
  - Receiver send SYN ACK response
  - Sender send ACK response
- After the handshake, When the datagram passes through socket, TCP'll direct it to the connection's send buffer (it's set during the handshake)
- TCP will grab the chunks from buffer and pass it to network layer from time to time
- The max data (payload) in a segment is limited by *maximum segment size (MSS)* — MSS is set by determining the maximum transmission unit, MTU, typically 1500 bytes
- TCP will add TCP header to the data, then pass to the network layer

▼ 3.5.2 TCP Segment Structure

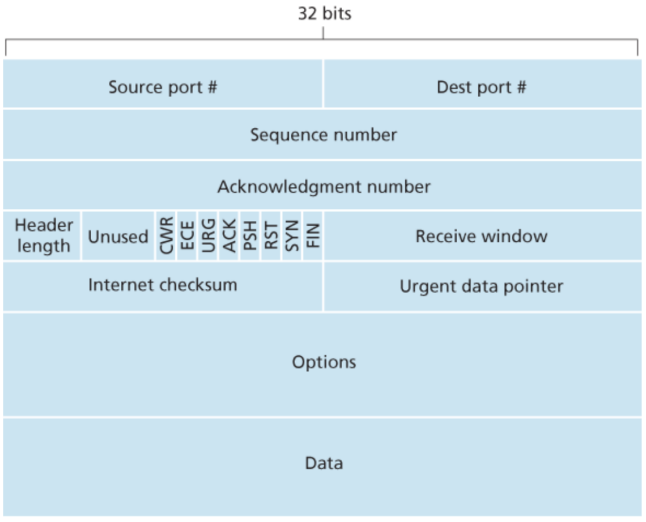


Figure 3.29 TCP segment structure

- Seq number (32 bits) & Ack number (32 bits): To implement reliable data transfer
- Receive window (16 bits): Flow protocol
- Header length (4 bits): Specify header length (all except payload), typically 20 bytes
- Options: Used when sender and receiver negotiates the *maximum segment size (MSS)*

▼ Flag field (6 bits)

- ACK: Indicate that a segment has been successfully received
- RST:
- SYN:
- FIN: To terminate TCP connection
- PSH: The receiver must pass the data to upper layer immediately
- URG: There’s data in the segment that the sending-side upper layer marked as urgent

▼ Seq number and Ack number

- seq number of a segment is the number of bytes it contain, eg 500

- Ack number is the number of bytes received + the next starting bytes the receiver expected, eg 501
- These is called cumulative acknowledgements.
- If the seq number is out of order, the RFC doesn't specify the procedure, it's up to the programmer to:
  - discard the segment
  - buffer the segment

▼ 3.5.3 Round-Trip-Time Estimation and Timeout

▼ How to estimate RTT?

- RTT is total time taken from a sender send a segment and it received ACK form the receiver
- sample RTT
  - Calculated for the latest segment (sent but not ACKED)
  - Since it'll fluctuate, to estimate RTT, the average of sampleRTT'll be used to estimate RTT = Estimated RTT
  - $EstimatedRTT = (0.875).EstimatedRTT + 0.125.SampleRTT$
- What is DevRTT?
  - Deviate RTT = how much typically SampleRTT deviates from EstimatedRTT
  - $DevRTT = 0.75.DevRTT\ .\ |SampleRTT - EstimatedRTT|$
  - smaller DevRTT = less deviate

▼ How to set TimeoutInterval (the time sender expect ACK response for a segment sent, if it take more time, the sender retransmit the segment)?

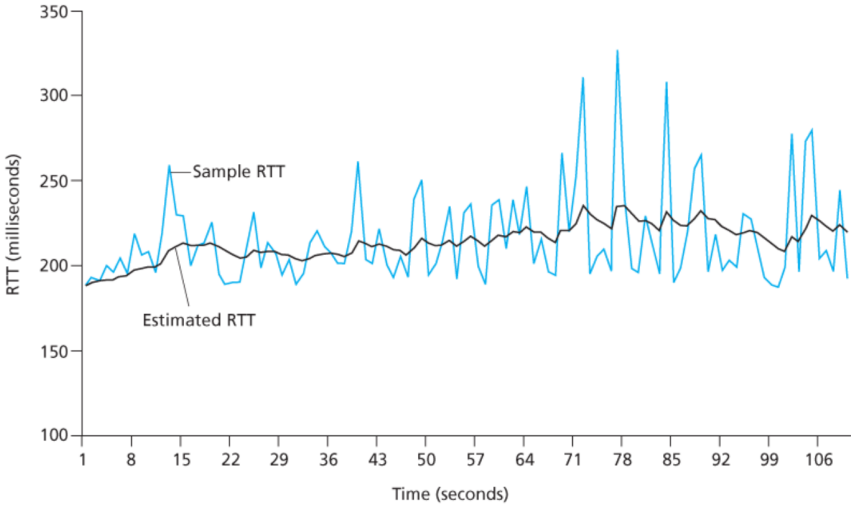


Figure 3.32 RTT samples and RTT estimates

- TimeoutInterval = EstimatedRTT + 4.DevRTT
  - An initial value of 1 second is recommended [RFC 6298]
  - When timeout happens, TimeoutInterval is doubled to avoid premature timeout (also important for congestion control—more retransmission, more congestion)
  - When there's new EstimatedRTT, TimeoutInterval is recalculated

▼ 3.5.4 Reliable Data Transfer

▼ What is Fast Retransmit?

Table 3.2 TCP ACK Generation Recommendation [RFC 5681]

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

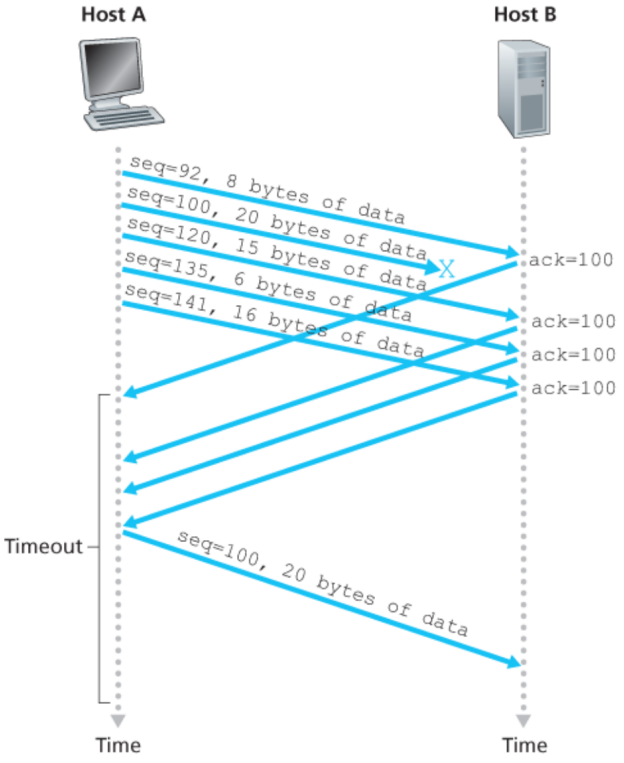


Figure 3.37 Fast retransmit: retransmitting the missing segment before the segment's timer expires

- A TCP sender can retransmit a packet even before the ACK exceeds timeoutInterval, by analysing the ACK response

- It’s because in the RFC, there are a few scenarios where the TCP receiver will send ACK, one is when it receives an out-of-order packets, where it’ll resend ACK for the previous packet.
- When the TCP sender receives 3 ACK for the same packet, it can conclude that the segment following the last ACKed segment is lost, so it’ll perform *fast retransmit*

▼ 3.5.5 Flow Control

- Flow control is to avoid the sender overflowing the receiver’s buffer

```
rwnd=RcvBuffer-[LastByteRcvd-LastByteRead]
```

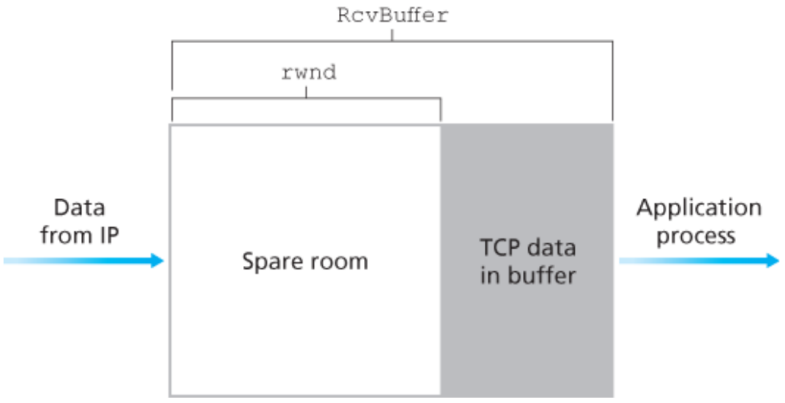


Figure 3.38 The receive window *(rwnd)* and the receive buffer *(RcvBuffer)*

- Flow control is achieved via receive window (give idea to sender how much free buffer space at the receiver)
- initially, rwnd (receive window) = RcvBuffer
- overtime, buffer space will add up due to unACK data (LastByteRcvd-LastByteRead), hence free buffer space is RcvBuffer - (LastByteRcvd-LastByteRead)
- to avoid overflow, host A makes sure that LastByteSent-LastByteAcked≤rwnd
- TCP specs requires host A to continue send segments with one data byte when the B’s rwnd is 0, so that B can respond with appropriate rwnd when it’s not congested (if not, B wont send any ACK that’ll tell

A that the buffer is available, and A wont send any segment since the last rwnd was 0, no transmission happen.

▼ 3.5.6 TCP Connection Management

▼ How does TCP connection established?

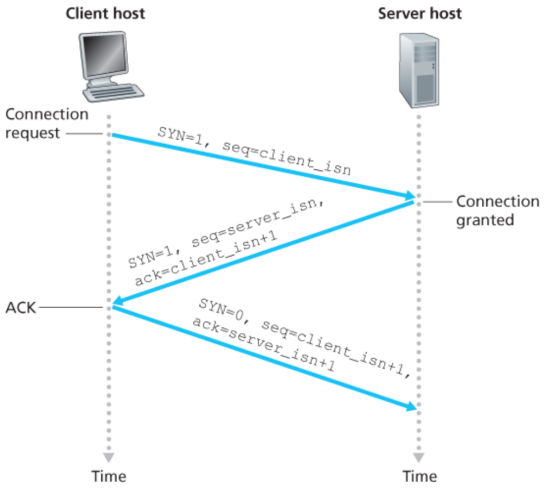
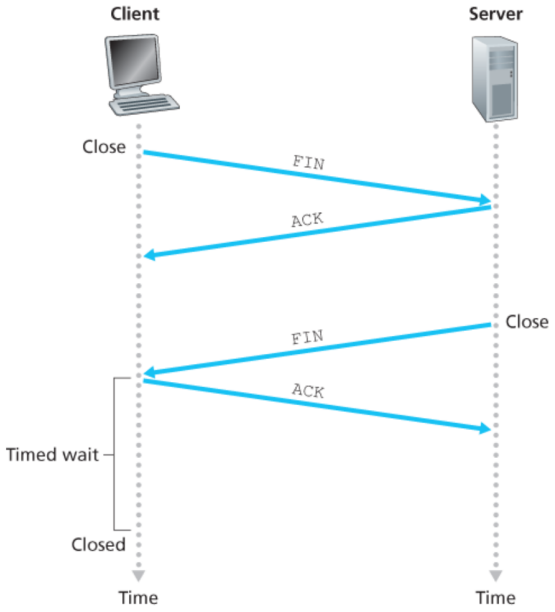


Figure 3.39 TCP three-way handshake: segment exchange

1. Client send SYN request, the initial seq number (client_isn) is randomly chosen to avoid security attack
2. Server allocate buffer space, respond with SYN ACK response, acknowledgement field of TCP segment set to *client_isn+1.* The server choose its initial seq number (server_isn), put it in the seq number field of the TXP segment header.
3. Client allocate buffers, send ACK segment (server_isn+1), may carry payload.

▼ How does TCP connection ended?



- Client send TCP segment with FIN flag set to 1
- Server send ACK to client
- Server send shutdown segment, FIN set to 1
- Client ACK the server's shutdown segment

▼ States of TCP protocol on client side

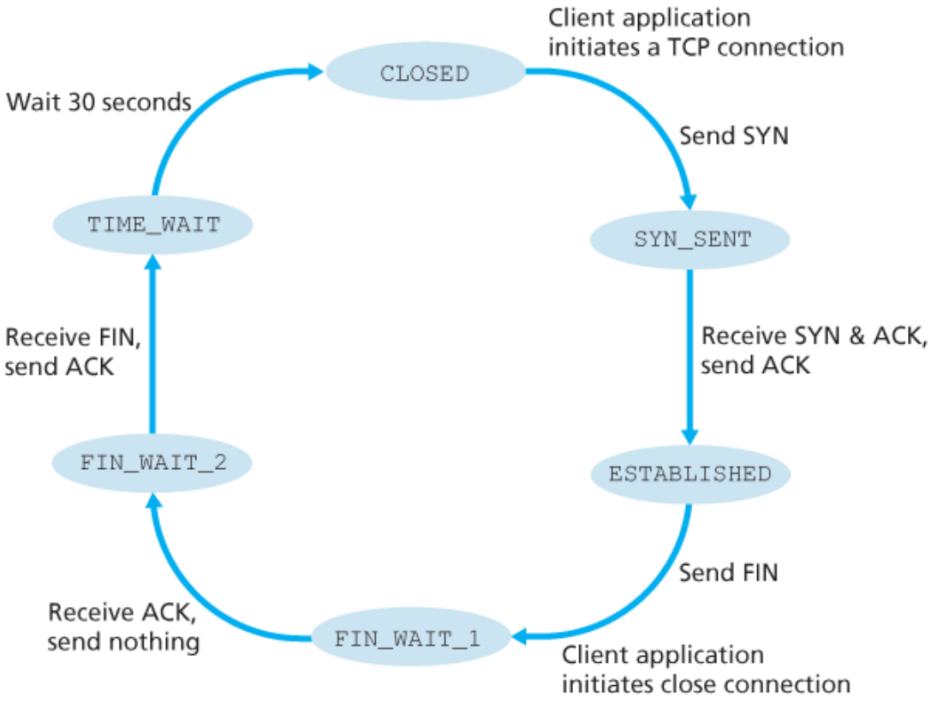


Figure 3.41 A typical sequence of TCP states visited by a client TCP

▼ States of TCP connection on server side

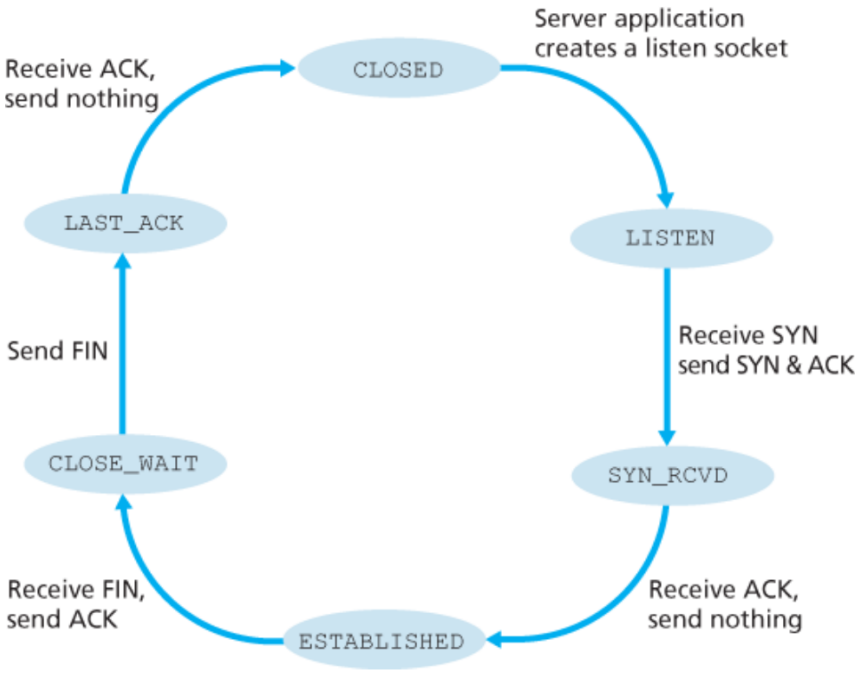


Figure 3.42 A typical sequence of TCP states visited by a server-side TCP

▼ 3.6 Principles of congestion control

▼ 3.6.2 Approach to Congestion Control

▼ End-to-end congestion control

- The network layer doesnt provide congestion-control, it’s done by the transport layer

▼ Network-assisted congestion control

- The router’ll provide feedback to the sender/ receiver regarding the congestion state of the network

▼ The feedback can be send in two ways:

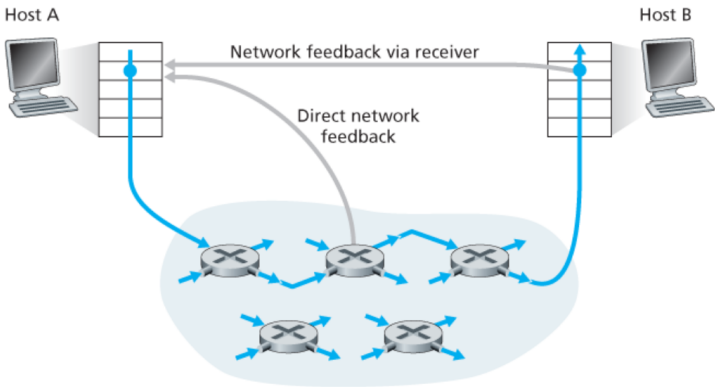


Figure 3.49 Two feedback pathways for network-indicated congestion information

- network router → sender
- router mark a field in packet indicating congestion (will take a full RTT)

▼ 3.7 TCP Congestion Control

▼ How can a TCP sender limit rate of send?

- The TCP congestion-control mechanism operating at the sender keeps track of congestion window (cwnd)
  - uncackedData≤ min {cwnd, rwnd}

▼ How a TCP sender know there is a congestion?

- Recall that if there is packet loss, receiver send 3 ACK.
- A packet loss can occur for multiple reasons, on is if there's congestion.

▼ How can TCP set its sending rate to make it no too slow that it underutilise the bandwidth and no too fast that it congest the network

- A lost segment implies congestion, hence, the TCP sender's rate should be decreased when a segment is lost.
- An acknowledged segment indicates that the network is delivering the sender's segments to the receiver, hence, the sender's rate can be increased when an ACK arrives for a previously unacknowledged segment
- Bandwidth probing - TCP sender'll increase the transmission rate until the network start to congest, then slow it a bit and the process is repeated.

▼ Components of TCP congestion-control algorithm:

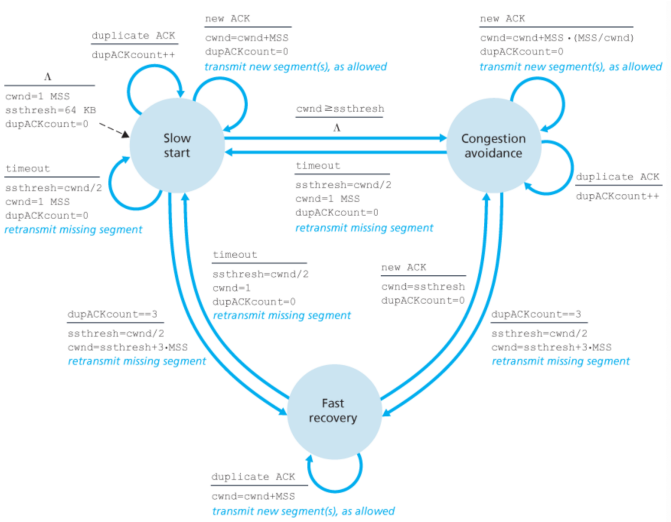


Figure 3.51 FSM description of TCP congestion control

▼ Slow Start

- When TCP connection begin, cwnd willl be set to 1 MSS, making initial rate = MSS/RTT
- MSS'll be increased by 1 every time ACK segment is received
- it'll exponentially grow until:
  - loss event occured (timeout) - cwnd set back to 1, a variable ssthresh (slow start threshold) set to half of the cwnd where the congestion occured
  - cwnd equal to ssthresh
- Then it'll move to congestion avoidance mode

▼ Congestion Avoidance

- when cwnd = ssthresh, MSS is increased by only a single MSS every RTT
- it'll stop when congestion occured, ssthresh set to new value, process is repeated

▼ Fast Recovery

- 2 ACK is received
- value of cwnd increased by 1 MSS for every duplicate ACK received for the missing segment

- when ACK arrives for the missing segment, TCP enter congestion avoidance mode

- *Fast recovery only incorporated in the newer version of TCP, TCP Reno. Old version, TCP Tahoe'll go to congestion avoidance either when timeout or 3 ACK.

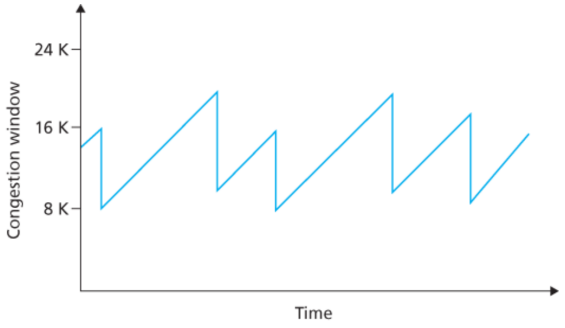


Figure 3.53 Additive-increase, multiplicative-decrease congestion control

- TCP congestion control is also referred as additive-increase, multiplicative-decrease (AIMD), since in slow start it increases the MSS (max segment size) by 1, then halving the MSS when congestion happen. It produces saw tooth behaviour as shown in figure 3.53.

▼ 3.7.2 Explicit Congestion Notification (ECN): Network-assisted Congestion Control

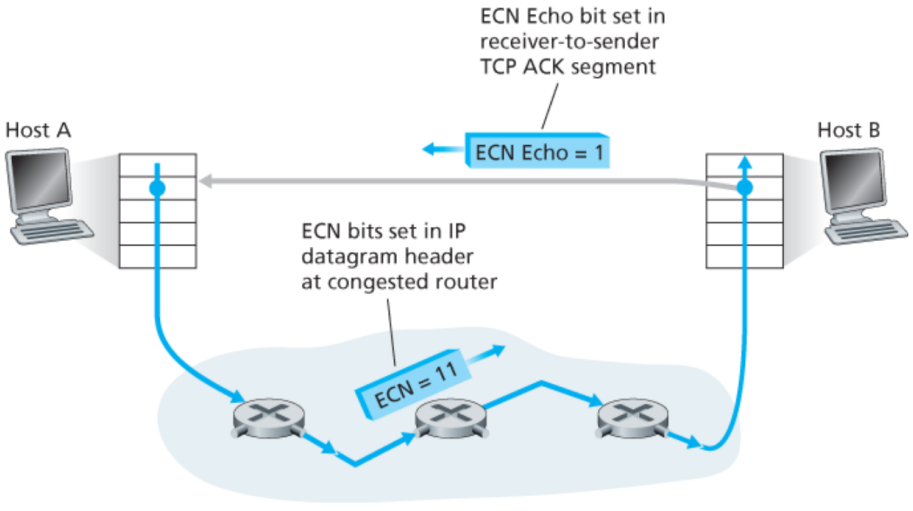


Figure 3.56 Explicit Congestion Notification: network-assisted congestion control

- Other than end-to-end congestion control mechanism by TCP, there is a network-based congestion control system called ECN

- 2 bits of ECN in IP header:
  - 1 used by router to indicate that it’s experiencing congestion - it’ll be carried to the destination host, which then inform the sending host
  - 1 is used by sender to inform router that the sender and receiver are ECN-capable, hence capable of taking action
- When the sender received ECN congestion indication, it’ll halve the congestion window

▼ Extensions to TCP

- Data Centre TCP (DCTCP) - used in data centre, uses ECCN
- Stream Transmission Protocol (SCTP) - reliable message-oriented protocol that allows different app-level streams to be multiplexed through a single SCTP connection
- TCP - Friendly Rate Control (TFRC) - goal is to smooth out the “saw-tooth” behaviour of TCP, suitable for multimedia app

▼ Chapter 4: The Network Layer: Data Plane

▼ 4.1 Overview of Network Layer

▼ What are 2 main parts of network layer?

- Data Plane - forward datagram from input link to output link
- Network Control Plane - control the per-router forwarding actions so that datagrams arrived at the designated receiver

▼ 4.1.1 Forwarding and Routing: The Data and Control Planes

▼ What are 2 main network-layer functions?

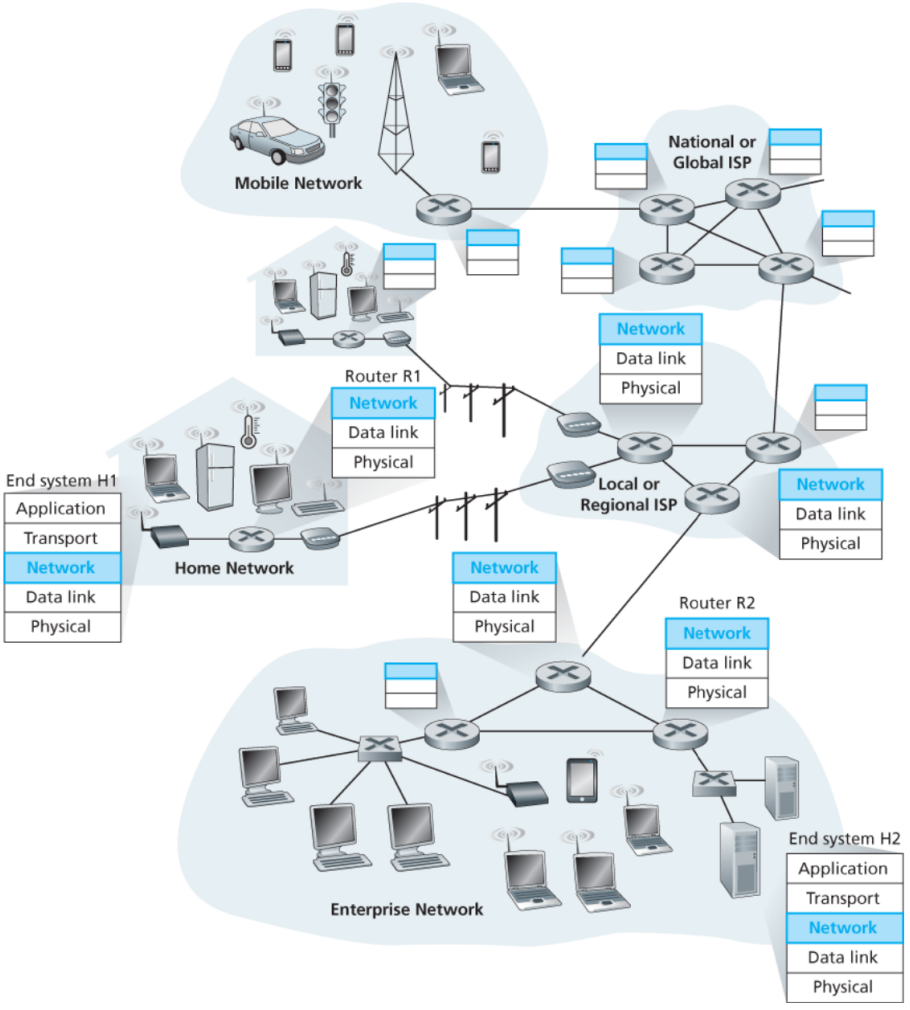


Figure 4.1 The network layer

- Forwarding -action of moves the packet to appropriate output link (port) in a router
- Routing - determines the end-to-end paths the packets will take
- analogy : drives from Nilai to johor,
  - forwarding: go through different turns, highway
  - routing: planning the trip, where to turn, where to stop etc

▼ What is forwarding table?

- a table that stores value of outgoing link interface to which the packet must be forwarded

▼ How is the router forwarding table configured?

▼ Traditional Approach

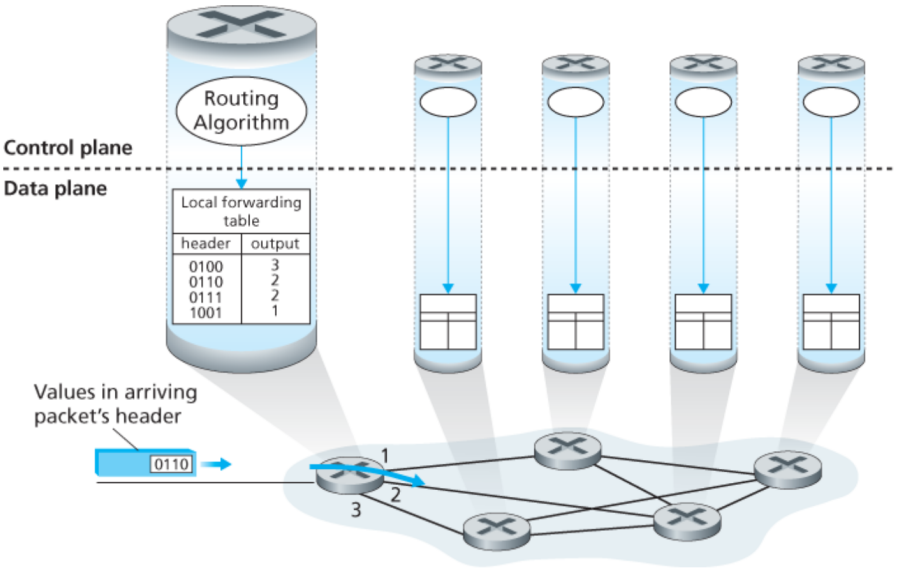


Figure 4.2 Routing algorithms determine values in forward tables

- A routing algorithm runs in every router, they communicate with each other to compute the values for its routing table

▼ (Software-defined networking) SDN Approach

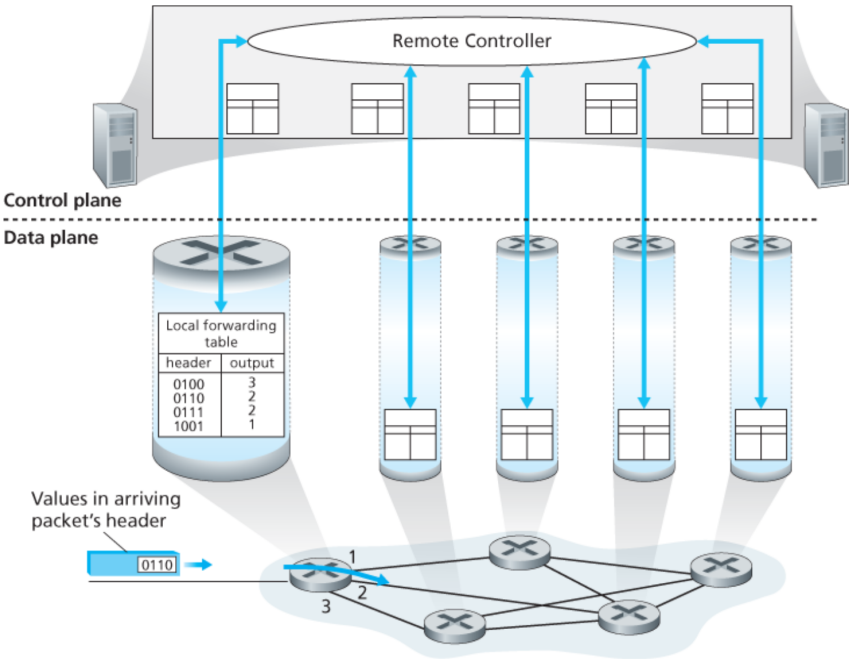


Figure 4.3 A remote controller determines and distributes values in forwarding tables

- The route is determined by the SDN on by ISP
- Router'll communicate with the SDN to. determine route

▼ 4.1.2 Network Service Model

▼ What are the service that network layer provide?

- Guarantee delivery - guarantees packet sent by a source'll arrive at destination host
- Guaranteed delivery with bounded delay - guarantee delivery with a specified delay
- In-order packet delivery - Guarantees packets arrive at destination in perfect order
- Guarantee minimal bandwidth - it sets a bit rate, as long as the sender transmits bits < bit rate, all packets'll arrive at the receiver
- Security - it encrypts all datagrams at the source, then decrypt at the destination

▼ 4.2 What's inside a Router?

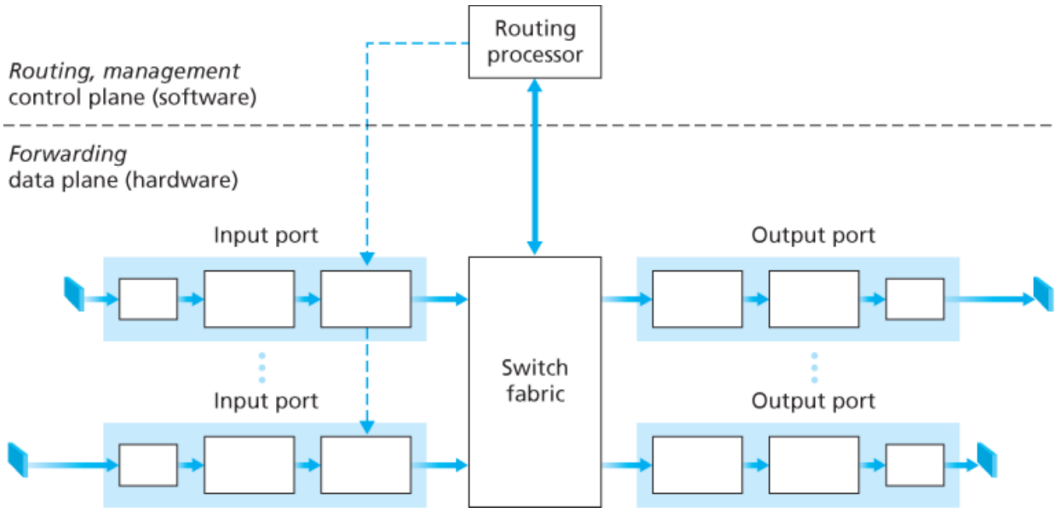


Figure 4.4 Router architecture

▼ Input ports

▼ The functions:

- leftmost box on input, rightmost box on output: act as the "door" for incoming, outgoing data
- Middle box: interoperate with other port

- Rightmost of input: lookup function, forwarding table is consulted to determine the router output port should the packet forwarded into

▼ How does it operate?

- input port lookup a destination address (”match”), the n send the packet into switching fabric to the specified output port(”action”) - destination-based forwarding

▼ Switching fabric

- connect router’s input ports to its output ports

▼ What are the types of switching?

▼ Switching via memory

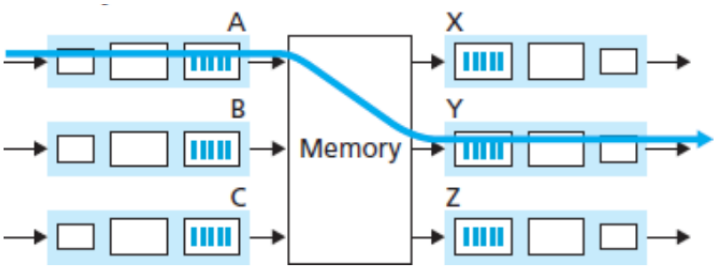
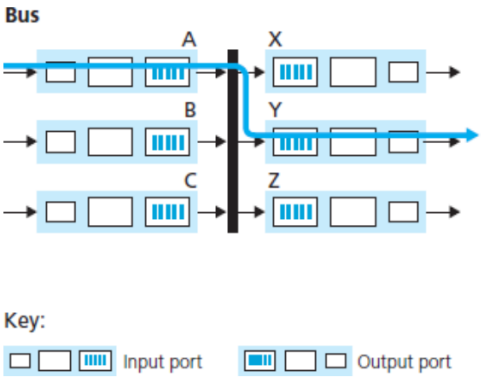


Figure 4.6 Three switching techniques

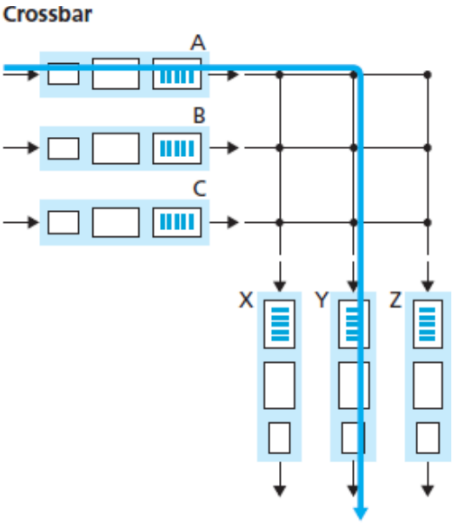
- done under direct control of routing processor
- packet from input port is copied into processor memory → processor extract destination address, lookup the appropriate output port → copy the packet to the output port buffer

▼ Switching via bus



- when packet arrive, a temporary header'll be add to assign which output port it should go
- the packet enter the bus one at a time, all output port receive the packet, only the designated port keep it

▼ Switching via interconnection network



-

▼ Output ports

- Transmit packets to appropriate outgoing link

▼ Routing Processor

- in traditional, it executes routing protocol
- in SDN, it communicates with the remote controller

> analogy: roundabout is the switch fabric, entry station is the input port, roundabout exit is the output port,

- Skipped - 366 - 374

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

▼ 4.3.1 IPv4 Datagram Format

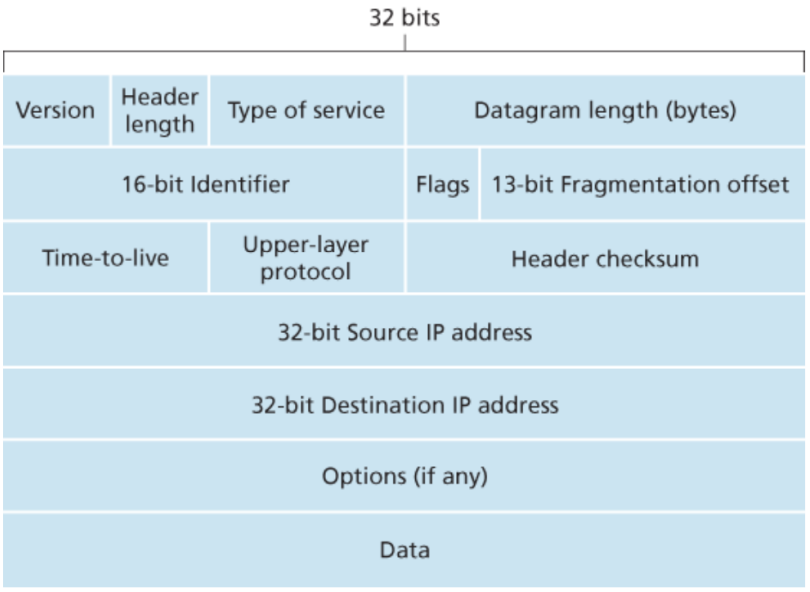


Figure 4.16 IPv4 datagram format

▼ Version number

- specify IP version, allow router to determine how to process the datagram

▼ Header length

- specify header length, so that can determine where the payload are (transport-layer segment and payloads)

▼ Types of service

- to differentiate types of IP datagrams (real-time such as used by IP telephony app and non-real-time such as FTP)

▼ Datagram length

- total length of the datagram (header+payload), usually size is ≤ 1500 bytes to allow it to fit in ethernet frame

- ▼ Identifier, flag, fragmentation offset
  - for Ip fragmentation
- ▼ Time-to-live
  - ensuring the datagrams do not circulate forever in the network. the field is decremented by one each time the datagram is processed by router, if TTL = 0, it’ll be dropped
- ▼ Protocol
  - to specify which trasnport-protocol should the data be passed, used when it reaches final destination
- ▼ Header checksum
  - for router to detect bit errors, if error it’ll be discarded (network layer won’t handle retransmission, transport-layer’ll handle it)
- ▼ Source and destination address
- ▼ Options
  - allow the IP header to be extended
  - removed in IPv6
- ▼ Data (payload)
  - transport-layer segment

▼ 4.3.2 What is IPv4 Datagram Fragmentation?

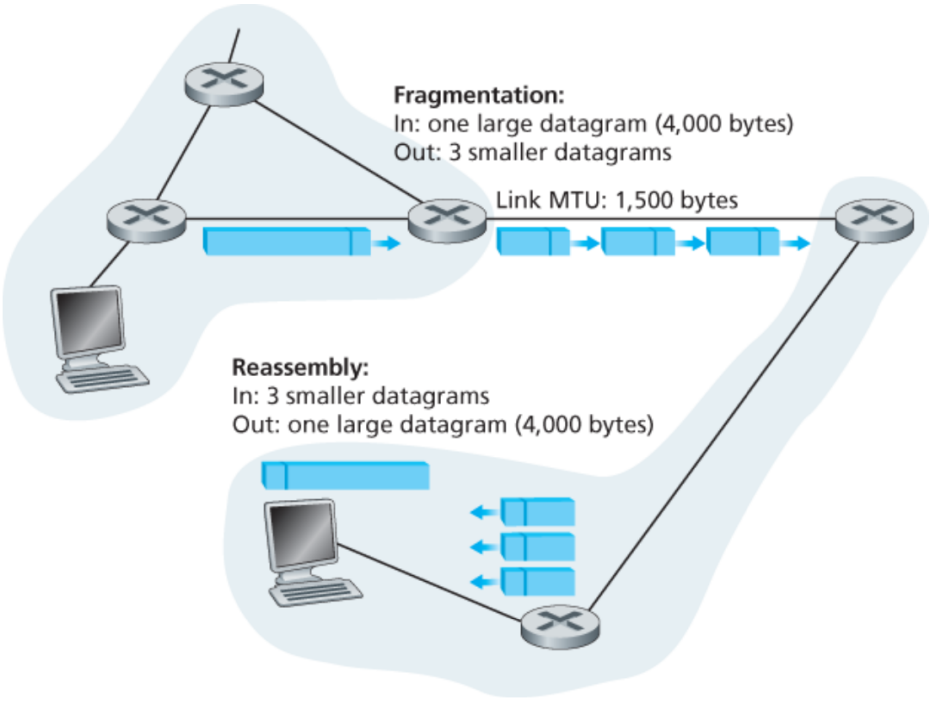


Figure 4.17 IP fragmentation and reassembly

- Different link-layer protocols can carry different size of network-layer datagram
- If the datagram size is larger than the link-layer protocol can carry, it is then divided into fragments
- When it arrives at the receiver, it will be combined referring to put *identification*, *flag* and *fragmentation offset fields* in the IP datagram header

▼ 4.3.3 IPv4 Addressing

- 255.255.255.255 or 00000000.00000000.00000000.00000000
- total possible IP address = 2^32 = 4 billions

▼ What is subnet?

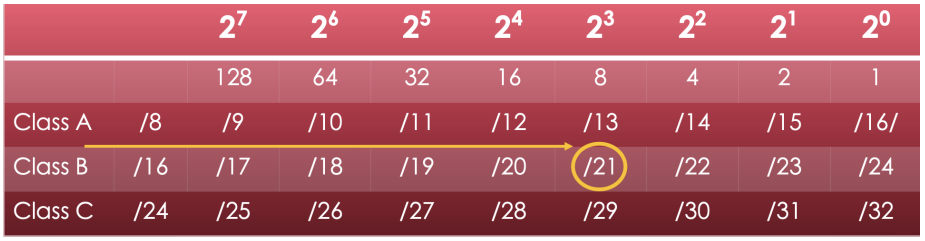
- eg: 223.1.1.0/24
- A network inside a network
- the first 24 bits represents the subnet

▼ How is Internet addressing handled?

▼ by using a strategy; Classless Interdomain Routing (CIDR)

- a.b.c.d/x
  - The first x bits are suffix, used when communicating with outside network
  - The rest are used within a network to distinguish between different host in the same network
  - No. of usable host = (2^n) - 2, since from the ip, the very first is the network address, the very last is the broadcast address

▼ old strategy; Classful Addressing



		2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
		128	64	32	16	8	4	2	1
Class A	/8	/9	/10	/11	/12	/13	/14	/15	/16/
Class B	/16	/17	/18	/19	/20	/21	/22	/23	/24
Class C	/24	/25	/26	/27	/28	/29	/30	/31	/32

- The prefix are only constrained to /8, /16, or /24, corresponds to class A, B, and C networks.
- Can be hard to choose, if /8 is too small (2^8-2 = 256 addresses), /16 is too big (2^16-2 = 65534).

- IP broadcast address, 255.255.255.255 is to refer to all the host in the subnet

▼ Who managed IP addresses?

- Internet Corporation for Assigned Names and Numbers (ICANN)

▼ How can a host obtain IP address?

▼ Dynamic Host Configuration Protocol (DHCP)

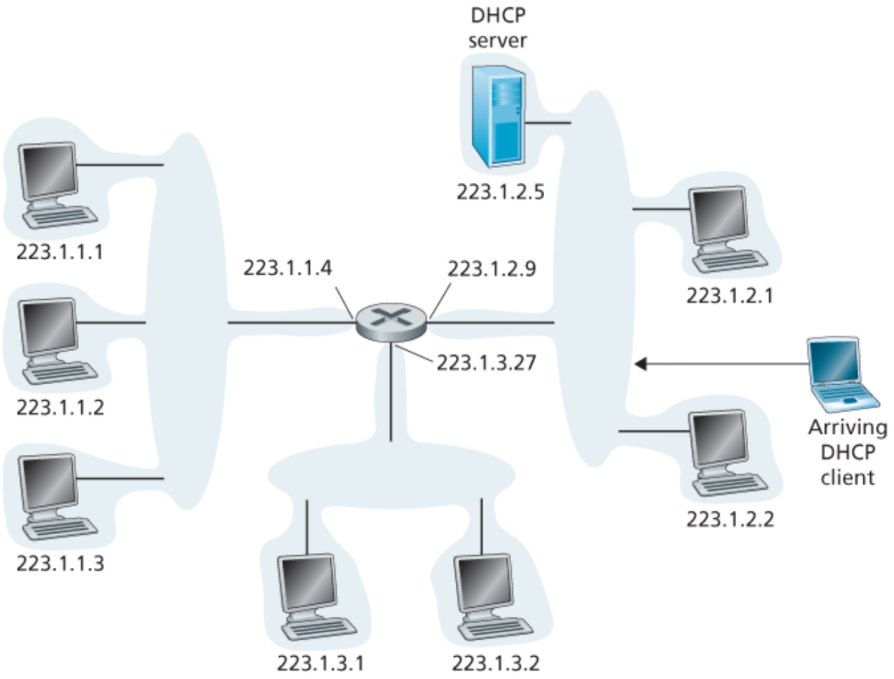


Figure 4.23 DHCP client and server

- plug-and-play, automatically allocate IP address to hosts
- a client-server protocol
- Each subnet will have a DHCP server

▼ When a host want an IP address, the process are:

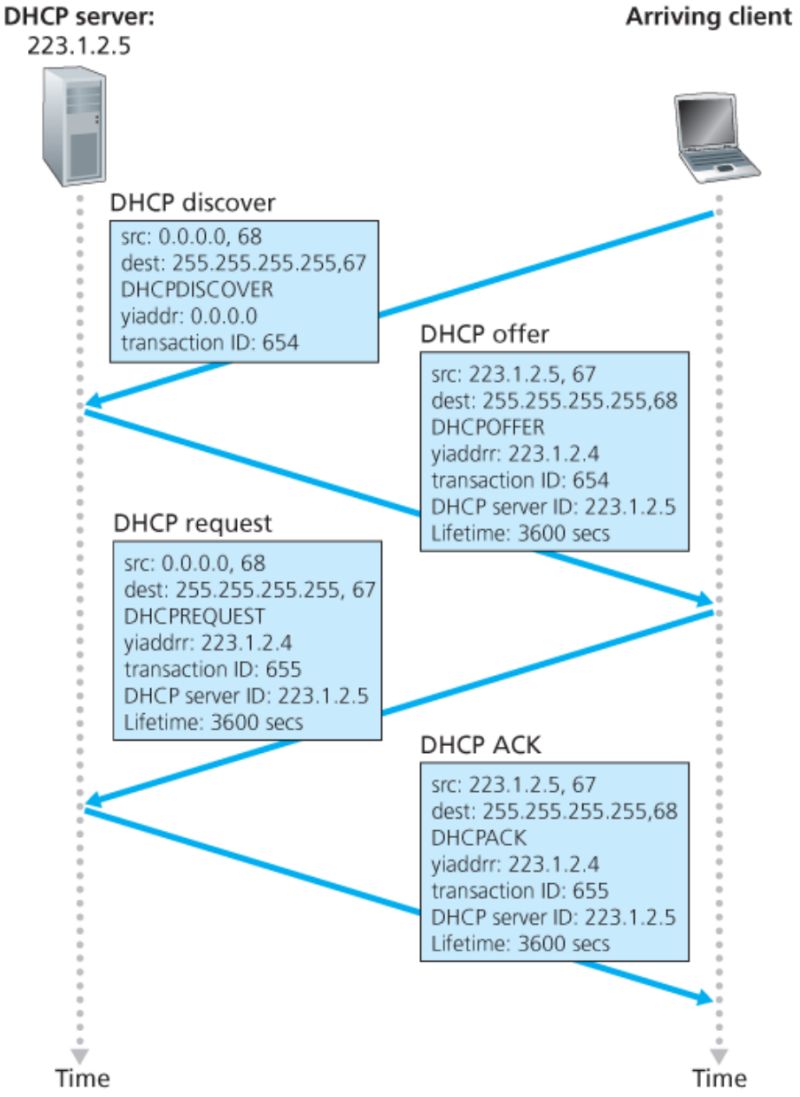


Figure 4.24 DHCP client-server interaction

- DHCP server discovery
  - client sent a DHCP discover message in a UDP packet to port 67, destination address is the broadcast address 255.255.255.255
- DHCP server offer(s)
  - DHCP responds with a DHCP offer message , sent to broadcast address (since a subnet may have multiple DHCP server, so client can choose which to choose)

        - contains: transaction ID, proposed IP address, network mask, IP address lease time (amount of time IP is valid)
    - DHCP request
        - Client'll choose one of the offerings, respond to the chosen with a DHCP request message
    - DHCP ACK
        - Server confirming the request

▼ How can a router handle many host within a subnet to connect to the Internet?

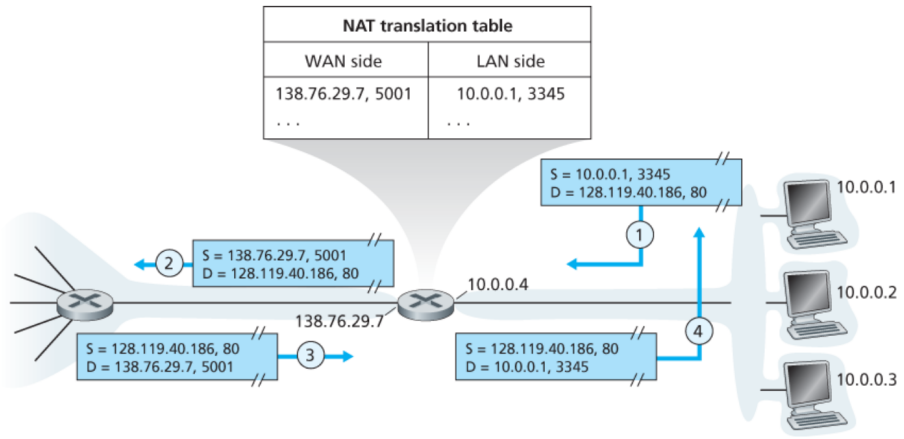


Figure 4.25 Network address translation

- By using Network Address Translation (NAT)
- If the router has an IP address of 128.76.29.7, all traffics in and out of the subnet'll have 128.76.29.7 as its address
    - eg: a host, IP = 10.0.0.1 request a web page on server 128.119.40.186
    - it send the datagram to the LAN.
    - The NAT router'll generate new source port number, replace the IP into its IP. (router can have up to 60 000 port numbers)

▼ 4.3.5 IPv6

▼ what are the difference with IPv4:

- ▼ Expanded addressing capabilities
  - 128 bits, every grain of sand can have IP address
  - new type of address, anycast address allows datagram to be delivered to a group of hosts. (a HTTP GET can be send to a number of mirror sites)
- ▼ 40-byte header
  - Faster processing of IP address header
- ▼ Flow labelling
  - packets can be labelled as flow, it'll have a special handling
- ▼ no fragmentation/ reassembly
  - if datagram is too big, router'll drop the packet, sends a "packet too big" ICMP error message to the sender
- ▼ no header checksum
  - since transport-layer and link-layer already have error checking, its redundant

▼ IPv6 Datagram Format

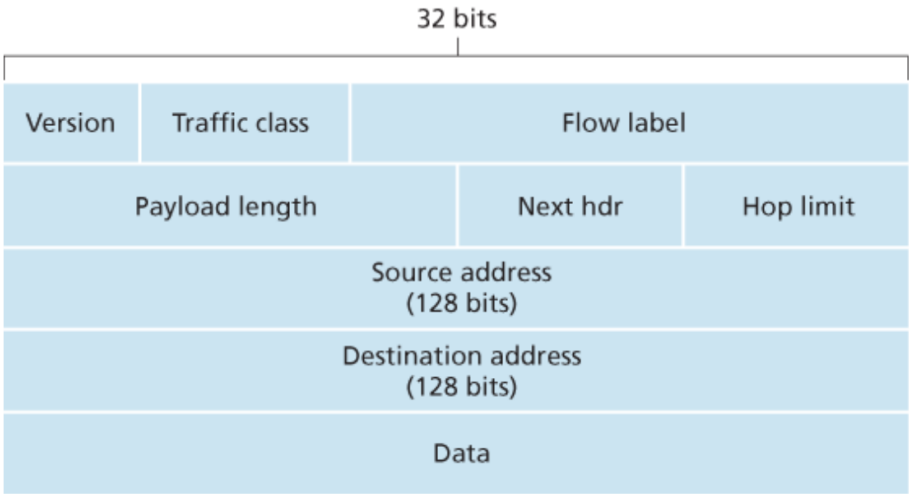


Figure 4.26 IPv6 datagram format

- Hop limit - decremented by one for every forwarding of the datagram, if 0, its discarded

▼ How to transition from IPv4 to IPv6?

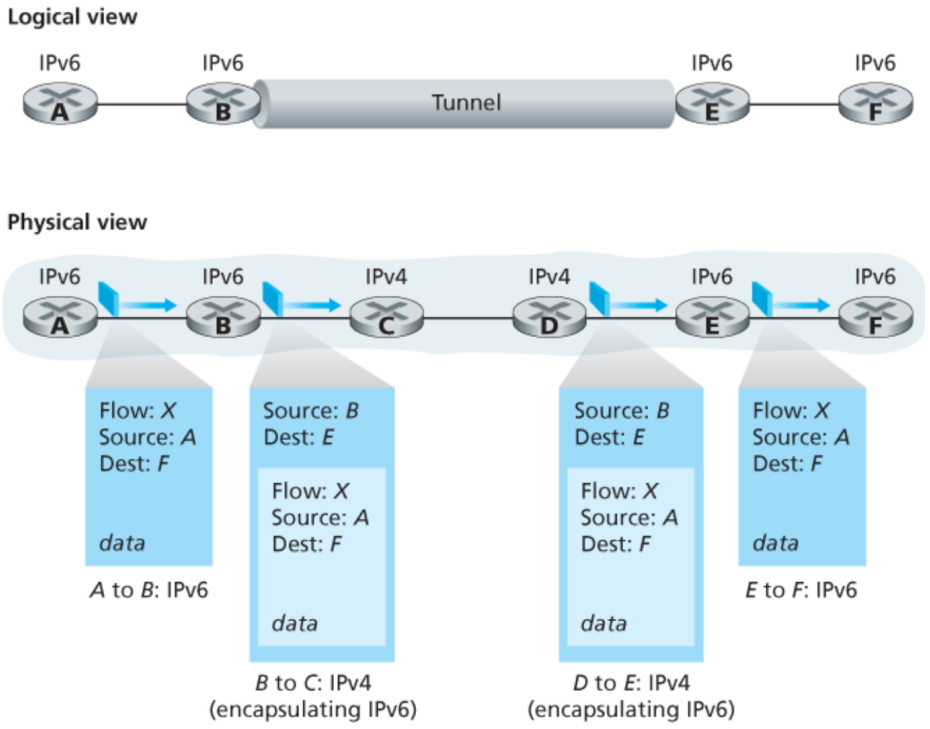


Figure 4.27 Tunneling

- By using tunnelling
- A and F wants to communicate, both uses IPv6, but the intermediates uses IPv4
- When IPv6 datagram arrives at B, B'll put the whole IPv6 datagram into the payload of IPv4
- When arrive at E, it'll extract the IPv6 datagram then route it to the appropriate receiever.

▼ 4.4 Generalised forwarding and SDN

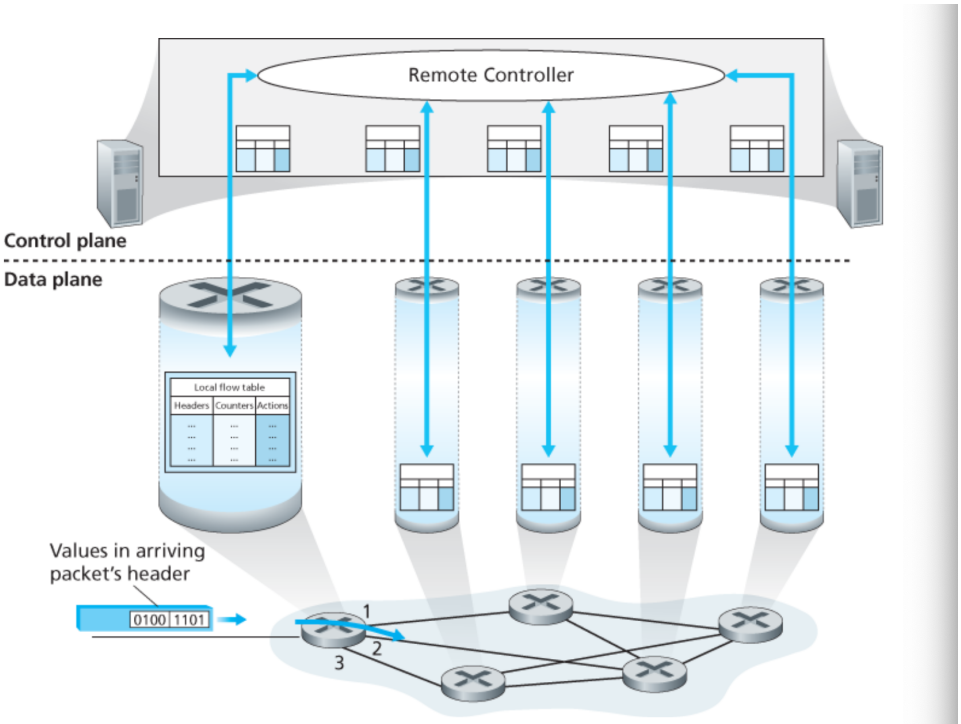


Figure 4.28 Generalized forwarding: Each packet switch contains a match-plus-action table that is computed and distributed by a remote controller

- there is a match-plus-action table in each packet switch, it includes (based on OpenFlow):
  - header field values: incoming packet will be matched
  - counters: count numbers of packets matched to table entry
  - action: actions to be taken when the packet matches a floe table entry

▼ 4.41 What can be matched in OpenFlow match-plus-action rule?

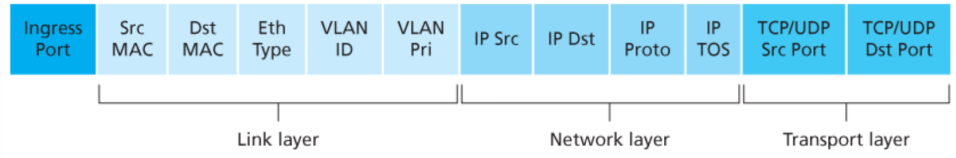


Figure 4.29 Packet matching fields, OpenFlow 1.0 flow table

▼ 4.4.2 What are the action can be made?

- Forwarding - packet may be forwarded
- Dropping - a flow table entry without action indicate that a matched packet should be dropped

- Modify-field - the values of header filed in fig4.29 may be modified before forwarding to appropriate output port

▼ Chapter 5: The Network Layer: Control Plane

▼ 5.2 Routing Algorithms

- The goals of a routing algorithm is to find the least-cost route, that is the shortest route.

▼ What are the classifications of routing algorithm?

▼ 1. Centralised or Decentralised

- Centralised - algorithm has complete information about connectivity and link costs
- Decentralised - the calculation is carried out iteratively, it changes information with its neighbouring nodes, gradually calculates the leats-cost path to the destination

▼ 2.Static or Dynamic

- Static - routes change very slowly, often as a result of human intervention
- Dynamic - changes the routing path as the network load or topology changes

▼ 3. Load-sensitive or Load-insensitive

- Load-sensitive - if a link is congested, it'll find another way

▼ What are the examples of Routing Algorithm?

▼ 5.2.1 Link-State (LS) Routing Algorithm - Dijkstra's algorithm

- A dynamic routing protocol, each router in the network maintains a detailed map or "link state" of the entire network. It includes the status and cost of each link in the network

▼ How does it works?

1. **Topology Discovery**: Each router broadcast information about its connected neighbours and their link cost to all other routers
2. **Link State Database (LSDB)**: Each router build their own LSDB which contains the complete map of the network

3. **Shortest Path Calculation** :Using LSDB, each router calculate the shortest path to every routers using Dijkstra's algorithm
4. **Routing Table Construction**: Based on the shortest path calculation, each router constructs its routing table
5. **Packet Forwarding** : Router'll forward the packet based on the routing table

▼ 5.2.2 The Distance-Vector (DV) Routing Algorithm

- A dynamic routing algorithm
- Routers only exchange informations with its directly connected neighbours

▼ How does DV works?

1. **Initial Routing Table**: Each router starts with its initial routing table, contains info about its directly connected neighbours and their costs
2. **Distance Vector Exchange**: Periodically, each router shares its routing table with its neighbours
3. **Distance Vector Update**: Each router updates their routing table (find shortest path) based on their neighbours's using Bellman-Ford algorithm
4. **Distance Table Updates**: If a router's routing table is updated, it sends the updated version to its neighbours
5. **Convergence**: Exchanging and updating continues until all routers have consistent information aka convergence
6. **Packet Forwarding**: Once convergence is reached, each router use its routing table to determine the best path

▼ 5.3 Intra-AS Routing in the Internet: Open Shortest Path First (OSPF)

▼ With millions of routers that operated by different ISPs, how to establish a connection between them?

- By organising the routers into autonomous systems (ASs), each AS consists of group of routers that are under the same administrative control (ISP)

- Each AS is identified by its globally unique autonomous system number (ASN) assigned by ICANN
- Routers within the same AS run the same routing algorithm(intra-autonomous system routing protocol) and have info about each other

▼ What is OSPF?

- Open Shortest Path First is an Intra-AS routing which is a link-state protocol that uses Dijkstra's algorithm
- Each router constructs a complete topological map of the entire autonomous system
- Each router runs Dijkstra's algorithm to determine shortest part to all subnets

▼ OSPF advanced properties

▼ Security

- Exchange between OSPF router can be authenticated, hence only trusted routers can participate in the OSPF protocol within an AS, preventing malicious intruders injecting incorrect info in to the routing table
- OSPF packet uses MD5

▼ Multiple same-cost paths

- If multiple paths have the same cost, it all can be used so that the traffic can be reduced

▼ Integrated support for unicast and multicast routing

- Multicast OSPF (MOSPF) uses the existing OSPF link database and adds a new type of link-state advertisement to the existing OSPF link-state broadcast mechanism.

▼ Support hierarchy within a single AS

- Each area can have its own OSPF routing algorithm
- Each area'll have one or more router responsible for routing packets outside the area.
- One OSPF act as a backbone

▼ 5.4 Routing Among the ISPs: Border Gateway Protocol (BGP)

- All ASs in the Internet run BGP as their inter-AS routing protocol
- It connects router to the outside of their AS
- It’s decentralised and asynchronous protocol, adopt distance-vector routing algorithm

▼ 5.4.1 BGP allows router to

1. Allows each subnet to advertise its existence to the rest of the Internet
2. Determine the best route to the prefixes

▼ 5.4.2 How does BGP advertises route information?

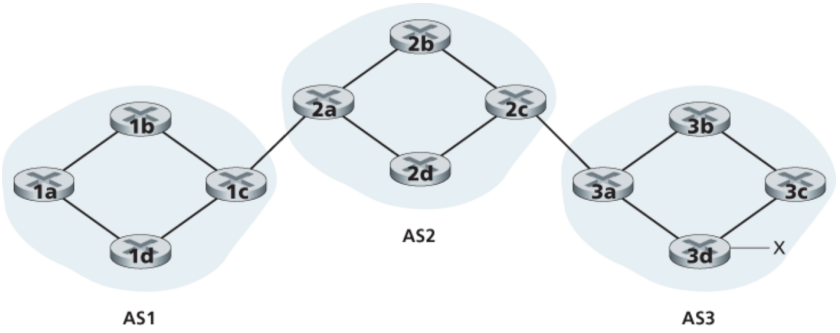


Figure 5.8 Network with three autonomous systems. AS3 includes a subnet with prefix x

- eg: subnet x (at router 3d) is to be advertised to all AS (AS1, AS2)
- router 1c, 2a, 2c, 3a act as gateway router (connect AS to other AS), connection between them are called external-BGP (eBGP)
- connection within an AS is called internal-BGP (iBGP)
- using eBGP, router 3a sends A*S3 x* (meaning - there is subnet x in AS3) message to router 2c
- router 2c uses iBGP to propagate message *AS3 x* to other routers in AS2
- router 2a uses eBGP to send message *AS2 AS3 x* to router 1c.
- using eBGP, router 1c sends *AS2* A*S3 x* (meaning - there is subnet x in AS3) message to router 2c

▼ 5.4.3 How does BGP determines the best routes?

- When advertising subnet using BGP, routers exchange BGP messages containing 2 attributes:
  - AS-PATH - contains list of ASs the route has passed through
  - NEXT-HOP - IP address that begins the AS-PATH (from previous example, NEXT-HOP for *AS2 AS3 x* is IP address of 2a

▼ BGP routing algorithms:

▼ Hot potato routing

- the goal is to get packets to the destination as quickly as possible, like having a hot potato on your hand
- considered as a selfish algorithm, reducing the cost of its own AS while ignoring other components outside of its AS

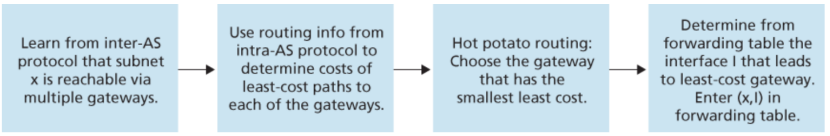


Figure 5.11 Steps in adding outside-AS destination in a router's forwarding table

▼ Route-Selection Algorithm

- Standard inter-AS routing protocol used on the Internet
- considers set of routes to the destination that have been accepted by the router
- If there are multiple routes, the following elimination rules are sequentially applied:
  1. Routes are assigned a local preference value, which can be set by the router or learned from another router in the same AS. Routes with the highest local preference values are selected.
  2. Among routes with the same highest local preference value, the route with the shortest AS-PATH is selected. This prioritises routes with fewer AS hops.
  3. If multiple routes remain with the same highest local preference value and AS-PATH length, hot potato

routing is used. The route with the closest NEXT-HOP router is selected.

4. If more than one route still remains, BGP identifiers are used to select the route.

▼ 5.4.4 What is IP-Anycast?

- a technique to replicate content on multiple servers located in different geographical locations and direct users to the server closest to them.
- BGP (Border Gateway Protocol) is often used to implement IP-anycast in applications such as Content Delivery Networks (CDNs) and the DNS system.
- In the IP-anycast configuration stage, the same IP address is assigned to each server in the network, and BGP is used to advertise this IP address from each server.
- BGP routers treat these advertisements as different paths to the same physical location, although they actually represent different paths to different physical locations.
- Each BGP router uses the BGP route-selection algorithm to determine the "best" route to the IP address, typically based on factors like AS-hop counts or proximity.
- When a client requests content from the IP address, the CDN or DNS system returns the common IP address regardless of the client's location.
- Internet routers, using the BGP route-selection algorithm, forward the client's request to the server that is determined to be the "closest" based on the chosen route.
- IP-anycast is extensively used by the DNS system to direct DNS queries to the nearest root DNS server, improving query response times.
- While IP-anycast can be effective in directing users to the closest server, CDNs often choose not to use it due to potential issues with routing changes leading to packets of the same TCP connection arriving at different server instances.

▼ 5.5 The Software Defined Network (SDN) Control Plane

▼ 5.5.1 SDN is an architectural approach to networking that have these characteristics:

▼ Flow-based forwarding

SDN-controlled switches can make packet forwarding decisions based on various header field values, including transport-layer, network-layer, or link-layer headers. This is in contrast to traditional router-based forwarding, which typically relies solely on the destination IP address. SDN switches use flow tables to specify packet forwarding rules, and it is the job of the SDN control plane to compute, manage, and install these flow table entries in the switches.

▼ Separation of data plane and control plane

In SDN, the data plane consists of the network switches, which execute the "match plus action" rules in their flow tables. The control plane, on the other hand, consists of servers and software that determine and manage the flow tables of the switches. This separation allows for centralised control and management of the network.

▼ Network control functions external to data-plane switches

The SDN control plane is implemented in software and executes on servers that are distinct and remote from the network switches. The control plane consists of an SDN controller (or network operating system) and a set of network-control applications. The controller maintains network state information and provides it to the network-control applications. It serves as a means for these applications to monitor, program, and control the underlying network devices. While the controller may be shown as a single central server, in practice, it is typically implemented on multiple servers for performance and availability.

▼ Programmable network

The network in SDN is programmable through the network-control applications that run in the control plane. These applications use APIs provided by the SDN controller to specify and control the behaviour of the network devices in the data plane. For example, a

routing application can determine end-to-end paths between sources and destinations using algorithms like Dijkstra's algorithm. Other applications can perform tasks such as access control or server load balancing. The network-control applications form the "brains" of the SDN control plane, enabling flexible and programmable network behaviour

▼ 5.5.2 OpenFLow Protocol

- it operates between SDN controller and SDN-controlled switch
- It uses TCP as the underlying transport protocol, typically using port number 6653.

▼ important message from SDN controller → switch

1. Configuration: This message allows the controller to query and set the configuration parameters of a switch. It enables the controller to manage the switch's settings.
2. Modify-State: This message is used by the controller to add, delete, or modify entries in the switch's flow table. It also allows the controller to set properties for switch ports.
3. Read-State: This message enables the controller to collect statistics and counter values from the switch's flow table and ports. It provides the controller with information about the network's state.
4. Send-Packet: This message allows the controller to send a specific packet out of a designated port at the controlled switch. The message carries the packet payload to be sent.

▼ Messages flowing from the SDN-controlled switch → controller

1. Flow-Removed: This message informs the controller when a flow table entry has been removed from the switch. This can occur due to reasons such as a timeout or as a result of a received modify-state message.
2. Port-Status: This message is sent by the switch to inform the controller about changes in the status of a switch port. It allows the controller to monitor and react to port status changes.
3. Packet-In: When a packet arrives at a switch port and doesn't match any flow table entry, it is sent to the controller for further

processing. Matched packets may also be sent to the controller as an action to be taken on a match. The packet-in message is used to send such packets to the controller for additional handling.

▼ 5.5.3 eg of how data and control plane interract:

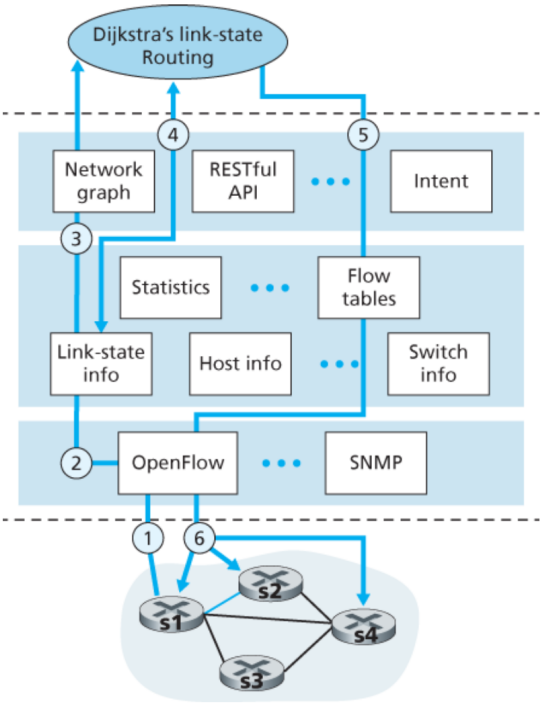


Figure 5.16 SDN controller scenario: Link-state change

1. Switch s1, experiencing a link failure between itself and s2, notifies the SDN controller of the link-state change using the OpenFlow *port-status* message.
2. The SDN controller receives the OpenFlow message indicating the link-state change, and notifies the link-state manager, which updates a link-state database.
3. The network-control application that implements Dijkstra's link-state routing has previously registered to be notified when link state changes. That application receives the notification of the link-state change.
4. The link-state routing application interacts with the link-state manager to get updated link state; it might also consult other components in the state-management layer. It then computes the new least-cost paths.

5. The link-state routing application then interacts with the flow table manager, which determines the flow tables to be updated.
6. The flow table manager then uses the OpenFlow protocol to update flow table entries at affected switches—s1 (which will now route packets destined to s2 via s4), s2 (which will now begin receiving packets from s1 via intermediate switch s4), and s4 (which must now forward packets from s1 destined to s2).

▼ 5.6 ICMP: The Internet Control Message Protocol

ICMP Type	Code	Description
0	0	echo reply (to ping)
3	0	destination network unreachable
3	1	destination host unreachable
3	2	destination protocol unreachable
3	3	destination port unreachable
3	6	destination network unknown
3	7	destination host unknown
4	0	source quench (congestion control)
8	0	echo request
9	0	router advertisement
10	0	router discovery
11	0	TTL expired
12	0	IP header bad

Figure 5.19 ICMP message types

- used by hosts and routers to communicate network-layer information to each other
- it’s carried as IP payload similar like TCP and UDP

▼ 5.7 Network Management and SNMP

▼ What is network management?

Network management includes the deployment, integration, and coordination of the hardware, software, and human elements to monitor, test, poll, configure, analyze, evaluate, and control the network and element resources to meet the real-time, operational performance, and Quality of Service requirements at a reasonable cost.

▼ 5.7.1 The Network Management Framework

▼ What are the key components of a network management

- **Managing Server**: The managing server is an application typically located in a centralized network management station, such as the network operations center (NOC). It serves as the central hub for network management activities, including collecting, processing, analyzing, and displaying network management information. The managing server allows network administrators to interact with the network's devices and initiate actions to control network behavior.
- **Managed Device**: A managed device refers to a piece of network equipment, including its software, that exists on the managed network. Examples of managed devices include hosts, routers, switches, modems, and other network-connected devices. A managed device can contain multiple managed objects, which represent specific hardware and software components within the device. For instance, a network interface card within a router is considered a managed object.
- **Management Information Base (MIB)**: Each managed object within a managed device has associated information that is collected into a Management Information Base (MIB). The MIB contains various types of information, such as counters, descriptive information, status information, and protocol-specific details. MIB objects are defined using a data description language called Structure of Management Information (SMI), ensuring well-defined and unambiguous syntax and semantics. MIB objects are organised into MIB modules, and there are both standard (defined by RFCs) and vendor-specific MIB modules.
- **Network Management Agent**: The network management agent is a process running on each managed device. It communicates with the managing server and carries out local actions on the managed device based on commands received from the managing server. The agent is responsible for executing management tasks and providing status updates or exceptional event notifications to the managing server

- **Network Management Protocol**: The network management protocol is the communication protocol used between the managing server and the managed devices. It enables the managing server to query the status of managed devices and issue commands to the agents running on those devices. The network management protocol also allows agents to notify the managing server about exceptional events or issues in the network. An example of a widely used network management protocol is SNMP (Simple Network Management Protocol).

▼ 5.7.2 Simple Network Management Protocol (SNMP)

- app-layer protocol to convey network-management control and information messages between a managing server and an agent

▼ What are the common usage of SNMP?

▼ request-response mode

- server sends a request [to query(retrieve) or modify (set) MIB object values] to SNMP agent
- SNMP agent performs action, sends reply

- agent to send unsolicited message, known as trap message to a managing server
  - Trap messages are used to notify a managing server of an exceptional situation that has resulted in changes to MIB values.

▼ Types of SNMP messages

Table 5.2 SNMPv2 PDU types

SNMPv2 PDU Type	Sender-receiver	Description
GetRequest	manager-to-agent	get value of one or more MIB object instances
GetNextRequest	manager-to-agent	get value of next MIB object instance in list or table
GetBulkRequest	manager-to-agent	get values in large block of data, for example, values in a large table
InformRequest	manager-to-manager	inform remote managing entity of MIB values remote to its access
SetRequest	manager-to-agent	set value of one or more MIB object instances
Response	agent-to-manager or	generated in response to
	manager-to-manager	*GetRequest*,
		GetNextRequest,
		GetBulkRequest,
		SetRequest PDU, or
		InformRequest
SNMPv2-Trap	agent-to-manager	inform manager of an exceptional event #

▼ Chapter 6: Link Layer and LANs

▼ 6.1 Introduction

▼ analogy

- To move from Nilai to Pavillion:
  - home → KTM Nilai by bike
  - KTM Nilai → MRT Kajang by train
  - MRT Kajang → MRT Bukit Bintang by mrt
  - MRT Bukit Bintang → Pavillion by taxi
- Same as packet, to move to a destination, it'll use different kind of link medium, hence different link layer protocols.

▼ 6.1.1 What are the services provided by the link layer?

- **Framing** - network layer is encapsulated into a link-layer frame
- **Link access** - medium access control (MAC) protocol responsible for coordinating the transmission of data frames on a shared communication link
- **Reliable delivery** - achieved with acknowledgements and retransmissions. the goal is to correct the error locally rather than forcing end-to-end retransmission of the data. It is unnecessary for low bit-errors links, such as fibre and coax
- Error detection and correction - bit errors can happen due to electromagnetic noise.

▼ 6.2 Error-Detection and Correction Techniques

▼ What are the three techniques used for detecting errors in the transmitted data?

▼ 6.2.1 Parity Checks

- types:
  - even parity scheme - one parity bit is added so that total 1s is even
  - odd parity scheme - parity value is chosen to there is odd number of 1s
- Receiver’ll count the number of 1, if it’s odd in an even parity scheme or vice-versa, then the data is corrupted

▼ 6.2.2 Checksumming Methods

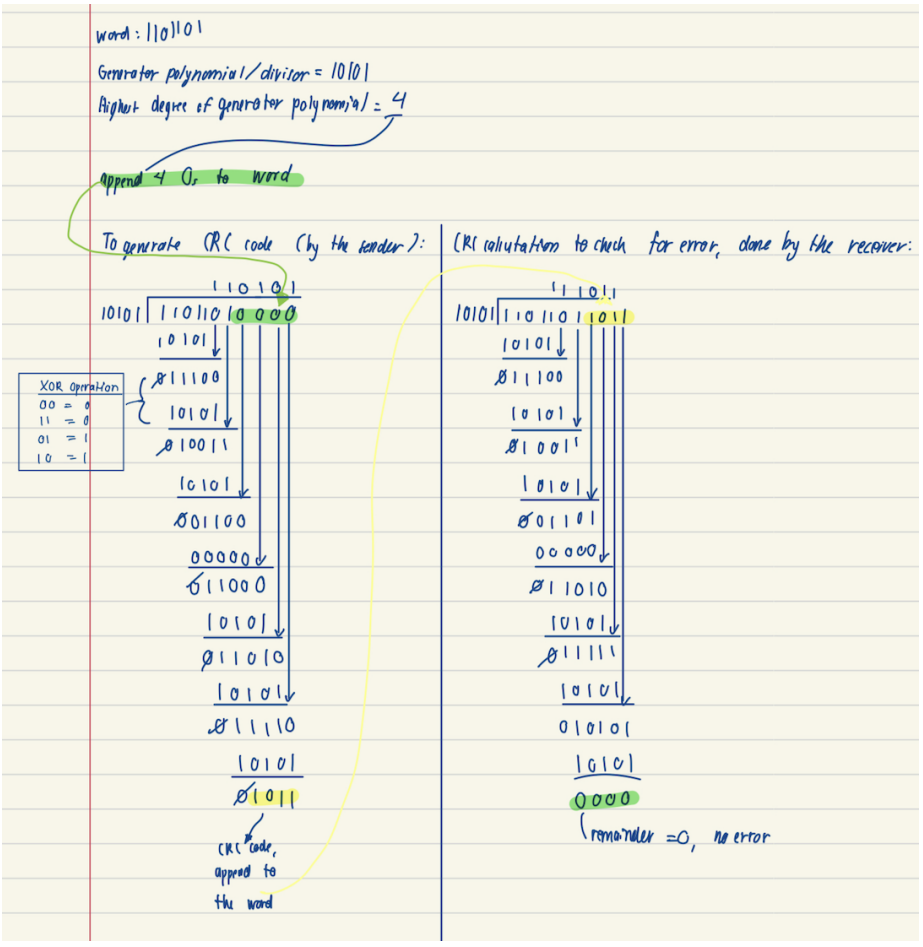
- the data bits are treated as sequence of k-bit integers
- the integers are summed, the sum is used as the error-detection bits
- provide weak protection against errors
- typically used at the transport layer, implemented in software

▼ 6.2.3 Cyclic Redundancy Check (CRC)

- The bit string to be sent is viewed as a polynomial with coefficients of 0 and 1

- The sender and receiver must agree on generator polynomial (divisor)
- The sender'll append CRC checksum/ CRC code, R to the original data
- When receiver receive the data, it'll perform CRC calculation

▼ example



▼ 6.3 Multiple Access Links and Protocols

- Multiple nodes on a shared broadcast in computer network can be a problem; how to coordinate the access of multiple sending and receiving nodes to a shared broadcast channel?
- the problem is called the multiple access problem.

▼ Analogy

The analogy of a classroom can be used to understand the multiple access problem. Imagine a classroom where students want to talk to different students at the same time. If they all speak at once without

any coordination, their messages will collide and become jumbled, making it impossible to understand. To prevent this, the students follow certain rules or protocols, such as raising their hands to speak or taking turns, ensuring that only one student speaks at a time and everyone gets a chance to be heard.

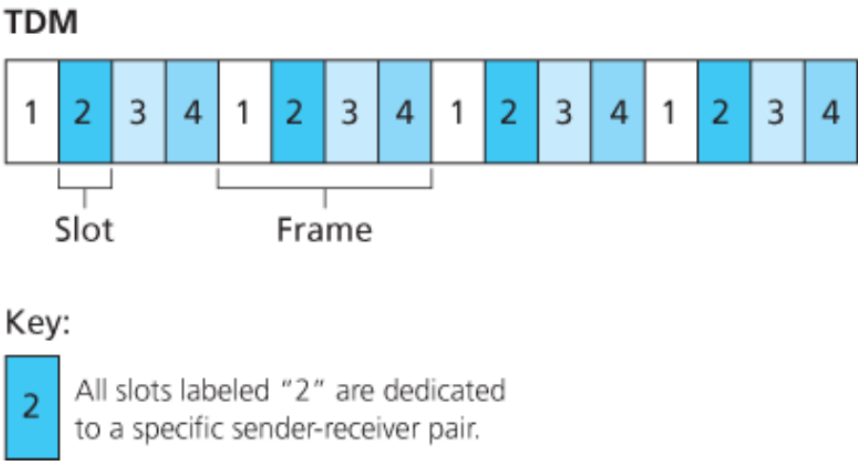
▼ what are the ideal multiple access protocols for a broadcast channel of rate R bits per second have these characteristics?

1. When only one node has data to send, that node has a throughput of *R* bps.
2. When *M* nodes have data to send, each of these nodes has a throughput of *R*/*M* bps. This need not necessarily imply that each of the *M* nodes always has an instantaneous rate of *R*/*M*, but rather that each node should have an average transmission rate of *R*/*M* over some suitably defined interval of time.
3. The protocol is decentralised; that is, there is no master node that represents a single point of failure for the network.
4. The protocol is simple, so that it is inexpensive to implement.

▼ What are the categories of multiple access protocols

▼ 6.3.1 Channel Partitioning Protocols

▼ Time-division multiplexing (TDM)

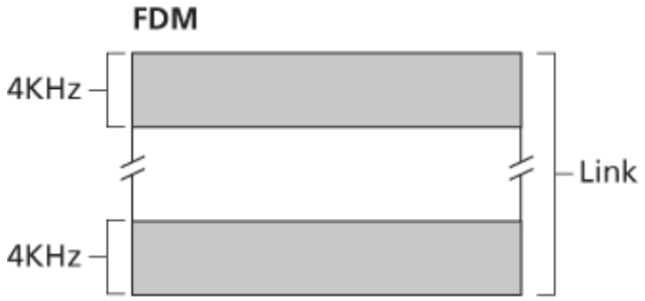


- Divides time into frames, frames into time slots. Each node is assigned a specific time slot, and when a node has data

to send, it transmits during its assigned time slot. TDM ensures that each node gets a fair share of the channel

- Drawback: Nodes are limited to an average rate even when they are the only ones with data to send, and they have to wait for their turn to transmit
- Analogy: Everyone in the classroom are assigned time to speak, one at a time during their allocated time-slot.

▼ Frequency-division multiplexing (FDM)



- FDM divides the channel's bandwidth into different frequencies and assigns each frequency to a node. It creates multiple smaller channels within the larger channel.
- drawback: nodes have limited bandwidth even when they’re the only one sending data

▼ Code Division Multiple Access (CDMA)

- CDMA assigns a unique code to each node, and each node uses its code to encode the data it sends. CDMA allows different nodes to transmit simultaneously while their receivers correctly receive the data despite interference. CDMA is widely used in cellular telephony and has anti-jamming properties.

▼ 6.3.2 Random Access Protocols

- Nodes transmit at the full channel rate, but collisions may occur. To mitigate collisions, nodes introduce random delays before retransmitting, increasing the probability of successful transmission by reducing the chances of repeated collisions.

▼ Slotted ALOHA

- each transmission node operates at the full speed of the channel, R bps
- When there is collision, each node wait for a random amount of time then retransmit again

▼ Two main problems

- Some slots'll be wasted due to collision: When there is a large number of nodes, the efficiency is only37%
- Some slot remain empty because nodes refrain from retransmitting for a random amount of time at the same time

▼ ALOHA

- precursor to slotted ALOHA
- When a frame arrives at a node, the node immediately transmits the entire frame into the broadcast channel without waiting for any time slots. If the transmitted frame collides with other transmissions, the node will retransmit the frame with a probability p. Otherwise, it will wait for a frame transmission time and either transmit with probability p or remain idle with probability 1 - p
- efficiency is half the slotted ALOHA or 18%!!

▼ Carrier Sense Multiple Access (CSMA), Carrier Sense Multiple Access with Collision Detection (CSMA/CD)

▼ Two important rules for human conversation

- Listen before speak = carrier sensing
  - a node listens to the channel before transmitting, if there is a frame being transmitted, it'll wait until it detects not transmission then begins transmission
- If someone begins talking at the same time, stop talking = collision detection
  - a node listens to the channel while transmitting, if it detects another node is transmitting, it'll stop transmitting, wait for a random time then retransmit

▼ How can collision still occur after carrier sensing?

It can happen due to the propagation delay of signals along the broadcast medium. Even if a node senses the channel as idle, there might be ongoing transmissions that haven't reached the sensing node yet. This delay in sensing leads to collisions. Hence, CSMA/CD was introduced

▼ How does it operates?

1. The node obtains a frame from the network layer and prepares it for transmission.
2. If the channel is sensed as idle, the node starts transmitting the frame. If the channel is sensed as busy, the node waits until it becomes idle.
3. While transmitting, the node continuously listens for any signal energy from other nodes on the channel.
4. If the node detects signal energy from other nodes during transmission, it aborts the transmission.
5. After aborting, the node waits for a random amount of time before returning to step 2.

▼ How to determine the random time before retransmission?

- Using the **binary exponential backoff** algorithm

▼ How does it work?

1. When a frame encounters n collisions, the node chooses a random value, K, from the set {0, 1, 2, ..., 2^n-1}.
2. The waiting time is calculated as K multiplied by 512 bit times (for Ethernet). This waiting time corresponds to the time it takes to transmit 512 bits on the Ethernet network.
3. The maximum value of n is typically limited to 10 collisions.

- For example, if a node transmits a frame for the first time and experiences a collision, it randomly selects K=0 with a 0.5 probability or K=1 with a 0.5

probability. If K=0, the node immediately starts sensing the channel again. If K=1, the node waits for 512 bit times (e.g., 5.12 microseconds for a 100 Mbps Ethernet) before initiating the sense-and-transmit-when-idle cycle. After each subsequent collision, K is chosen randomly from an increasing set of values.

▼ How efficient is it?

- formula, *CSMA/CD efficiency = 1/(1 + 5 * (dprop/dtrans))*
- ddrop = maximum time it takes for signal energy to propagate between any two adapters
  - as the propagation delay (dprop) approaches zero, the efficiency approaches 1 — with minimal propagation delay, colliding nodes can quickly detect collisions and abort their transmissions, avoiding wasted channel time
- dtrans = the time required to transmit a maximum-size frame (approximately 1.2 milliseconds for a 10 Mbps Ethernet)
  - as the transmission time (dtrans) becomes very large, the efficiency also approaches 1 — when a node successfully accesses the channel, it can utilize it for an extended period, resulting in productive utilization of the channel

▼ 6.3.3 Taking-Turns Protocols

▼ Polling protocol

- It requires a designated master node to sequentially poll each node in a round-robin fashion. The master node sends a message to each node, allowing them to transmit a maximum number of frames. After a node has transmitted its frames, the master node moves on to the next node. This eliminates collisions and empty slots, resulting in higher efficiency

- Drawbacks: polling delay and relies on the master node, which can reduce the throughput if only one node is active and can lead to channel inoperability if the master node fails.

▼ Token-passing protocol

- A special frame called a token is passed among the nodes in a fixed order. When a node receives the token, it holds onto it if it has frames to transmit, forwarding it otherwise. If a node has frames, it sends them up to max no of frames and then passes the token to the next node. Token passing is decentralised and highly efficient
- Drawback: potential for a single node failure to disrupt the entire channel or the need for recovery procedures if a node fails to release the token.
- New token-passing protocols : Fibre Distributed Data Interface (FDDI) and IEEE 802.5 token ring protocol

▼ 6.3.4 Data-Over-Cable Service Interface Specification (DOCSIS): Link-Layer Protocol for Cable Internet Access

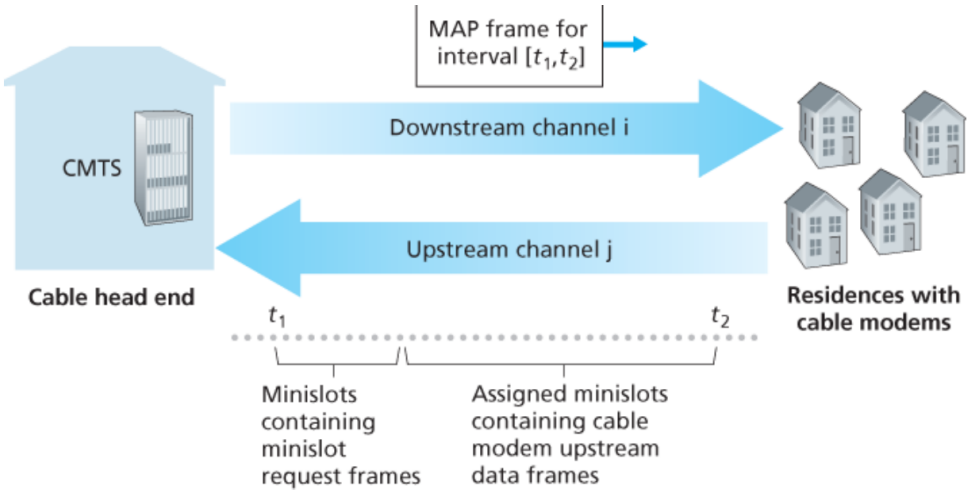


Figure 6.14 Upstream and downstream channels between CMTS and cable modems

▼ What is DOCSIS?

- A standard that defines the protocols used in cable networks for cable Internet access

- It uses Frequency Division Multiplexing (FDM) to divide the network into downstream(Cable Modem Termination System (CMTS) → modem) and upstream channels (modem → CMTS)
- Each upstream is shared by multiple modems, hence can lead to collisions
  - Therefore CMTS uses TDM-like approach, the upstream channel is divided into intervals of time, each containing mini-slots
  - CMTS grant permission to individual cable modems to transmit during specific mini-slots by sending a control message called MAP on the downstream channel

  ▼ How can a modem inform CMTS that it want to send data?

    - They send mini-slot-request frames to the CMTS during dedicated interval mini-slots
    - These frames may collide with other mini-slot-request (modem infer that it collides if it didn't receive any response from CMTS)
    - If collision happen, modems use binary exponential backoff to delay retransmission of the mini-slot-request frame

▼ 6.4 Switched Local Area Network

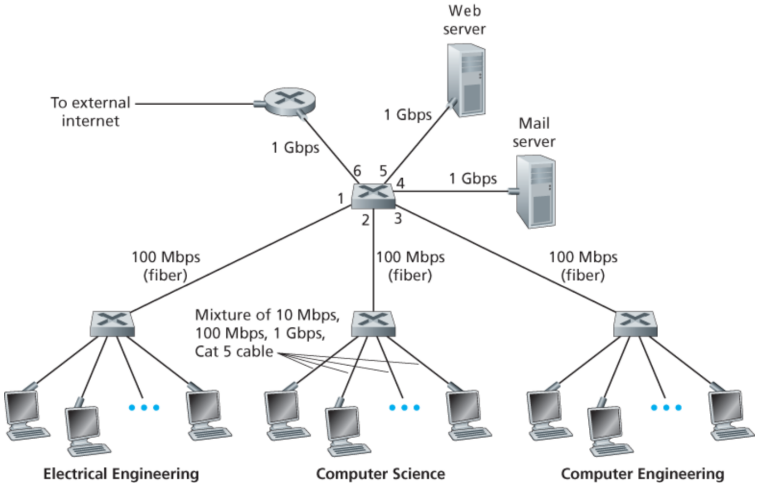


Figure 6.15 An institutional network connected together by four switches

- figure shows local network, the switches operate at the link-layer, hence they uses link-layer frames instead of network-layer datagram, don't

recognise network-layer addresses (IP address) and don’t use routing algorithm like RIP or OSPF for routing

- instead, they use Link-Layer address

▼ 6.4.1 Link-Layer Addressing and ARP

▼ Media Access Control (MAC) Addresses

- A device network interface have both IP and MAC address where MAC is permanent while IP can change
- MAC is primarily used in a local network
- MAC address is 6 bytes long and expressed in hexadecimal notation
- Every MAC address is unique, manufacturers purchase first 24 bits of MAC address from the IEEE then it’s up to the manufacturer to set the rest bits
- When sending a link-layer frame, the sender includes destination’s MAC address.
- To broadcast a frame to all hosts on the LAN, destination MAC address is set to FF-FF-FF-FF-FF-FF (48 bits of 1s)

▼ How can an IP address be translated into MAC address?

- by using the Address Resolution Protocol (ARP)
- each device maintain an ARP table that map IP Address to MAC Address.

▼ How does it work?

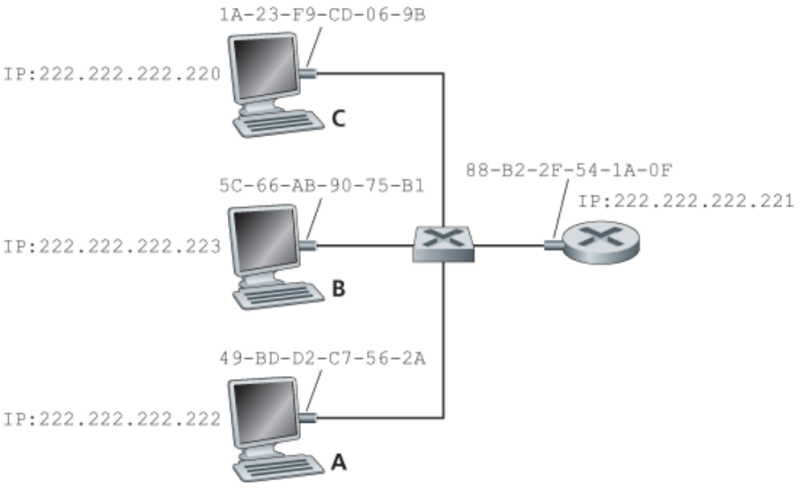


Figure 6.17 Each interface on a LAN has an IP address and a MAC address

IP Address	MAC Address	TTL
222.222.222.221	88-B2-2F-54-1A-0F	13:45:00
222.222.222.223	5C-66-AB-90-75-B1	13:52:00

Figure 6.18 A possible ARP table in 222.222.222.220

- Host A with IP address 192.168.1.10 and Host B with IP address 192.168.1.20.
- Host A wants to send an IP datagram to Host B.
- However, Host A needs to know the MAC address of Host B to construct the link-layer frame for transmission.
- Host A checks its ARP table to see if it has the MAC address of Host B.
- If the ARP table has an entry for Host B's IP address, Host A can use the corresponding MAC address.
- If the ARP table doesn't have an entry for Host B's IP address, Host A needs to perform an ARP resolution.
- Host A broadcasts an ARP query packet on the network, asking for the MAC address associated with IP address 192.168.1.20.
- The query packet is sent to the broadcast MAC address(FF-FF-FF-FF-FF-FF), which ensures all devices on the subnet receive it.

- Host B receives the ARP query packet, recognises its IP address, and sends an ARP response packet to Host A with its MAC address.
- Host A updates its ARP table with the IP-to-MAC mapping for Host B.
- Now, with the MAC address of Host B, Host A can construct the link-layer frame and send the IP datagram to Host B.

▼ Analogy

- Abu want to know the IC number of someone who lives at room DA2
- Hence he'll shout in a room full of people "What's the IC number of the person living in room DA2?"

▼ How can a host send datagram off its subnet?

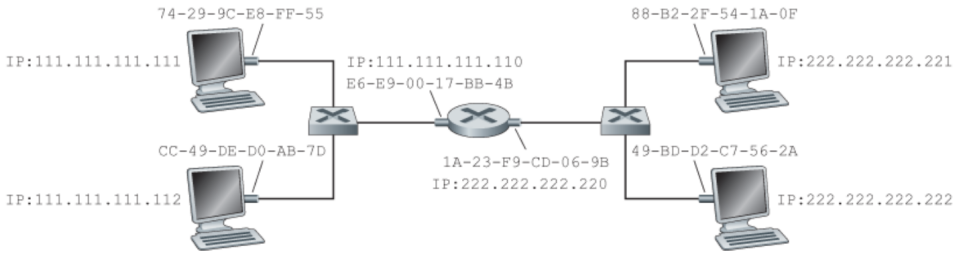


Figure 6.19 Two subnets interconnected by a router

- Each host has one IP address and one adapter, while a router has an IP address for each of its interfaces.
- Subnet 1 has the network address 111.111.111/24, and Subnet 2 has the network address 222.222.222/24.
- Suppose host 111.111.111.111 wants to send an IP datagram to host 222.222.222.222.
- The sending host needs to determine the appropriate destination MAC address for the link-layer frame.
- It cannot use the MAC address of host 222.222.222.222 because it won't match any adapter on Subnet 1.
- Instead, it needs to use the MAC address of the router interface 111.111.111.110, which is the first-hop router on the path to the

destination.

- The sending host uses ARP to obtain the MAC address of 111.111.111.110.
- Once it has the MAC address, it creates a frame containing the datagram addressed to 222.222.222.222 and sends it to Subnet 1.
- The router adapter on Subnet 1 receives the frame addressed to it and passes it to the network layer of the router.
- The router consults its forwarding table and determines that the datagram should be forwarded via router interface 222.222.222.220.
- The router obtains the destination MAC address through ARP.
- The router encapsulates the datagram in a new frame with the correct MAC address and sends it to Subnet 2.
- The frame reaches the destination host, and the datagram is successfully delivered.

▼ 6.4.2 Ethernet

- Ethernet has become the dominant wired LAN technology due to several factors. It was the first high-speed LAN to be widely deployed, making network administrators familiar with its workings. Other LAN technologies like token ring, FDDI, and ATM were more complex and expensive, discouraging switches. Ethernet continually evolved to match or exceed the data rates of competing technologies. The introduction of switched Ethernet in the 1990s further improved data rates. Ethernet's popularity made hardware cheap and widely available. Initially using a bus topology, Ethernet transitioned to a hub-based star topology and eventually to switch-based installations. Ethernet operates at the link layer, using CSMA/CD for multiple access and broadcast communication.

▼ Ethernet Frame Structure

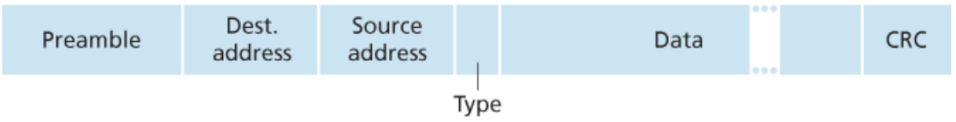


Figure 6.20 Ethernet frame structure

▼ Data field (46 - 1,500 bytes)

- carries IP datagram, if IP datagram > 1500, it'll be fragmented, if < 46 bytes, it'll be stuffed
- Network layer'll remove the stuffing based on the length field in the datagram header

▼ Destination address (6 bytes)

- MAC address of the destination adapter BB-BB-BB-BB-BB-BB

▼ Source address (6 bytes)

- MAC address of network adapter that transmit the frame to LAN

▼ Type field (2 bytes)

- Ethernet can carry multiple network-layer protocol, hence type is used to distinguish each type

▼ Cyclic Redundancy Check (CRC) (4 bytes)

- For error detection (if there is error, it'll simply discard the frame, the retransmission protocol is handled by the transport-layer protocol if there's any (TCP is reliable, UDP not)
- Hence, Ethernet is an unreliable service

▼ Preamble (8 bytes)

- to prepare the receiving adapters and synchronise their clock with the sender's clock
- 7 bytes of preamble is filled with *10101010,* it act as a wake-up call to the receiver that data will come
- the last 2 bits of the 8th preamble is 1011, signalling that the the important data of the frame is about to follow

- The purpose of the preamble is to establish a common timing reference between the sender and receiver. By analysing the bit pattern in the preamble, the receiver can detect the clock frequency and adjust its own clock accordingly. This synchronisation allows the receiver to anticipate the arrival of the actual data and receive it at the correct speed.

▼ 6.4.3 Link-Layer Switches

▼ How does switch work?

switch table:

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

Figure 6.22 Portion of a switch table for the uppermost switch in Figure 6.15

▼ Based on the switch table, switch can :

- **Filtering** - determines whether a frame should be forwarded to some interface or dropped
- **Forwarding** - determines the interfaces to which frame should be directed, then moves it to those interfaces

▼ Possible scenarios (suppose a frame with destination *DD-DD-DD-DD-DD-DD* arrives on interface *x*):

- There is no entry for *DD-DD-DD-DD-DD-DD* , the switch forward the packet to all interface except for x (switch broadcast the frame)
- *DD-DD-DD-DD-DD-DD* associated with interface x. Since the frame is already in the LAN segment that contains *DD-DD-DD-DD-DD-DD* (it comes from interface x) then there's no need to forward the frame, the frame is discarded
- *DD-DD-DD-DD-DD-DD* is associated with interface y, the switch forward it to the interface

▼ Self-Learning

switches are plug-and-play devices, it doesn’t need any set-ups:

1. A switch starts with an empty table.
2. When a frame is received on an interface, the switch records the MAC address, interface, and current time in its table, associating the sender's address with the interface it arrived on.
3. The switch updates its table as more frames are received, ensuring that every host on the LAN is eventually recorded.
4. If no frames are received from a specific MAC address within a certain time period (aging time), the switch removes that address from its table.
5. This automatic and dynamic process allows the switch to keep track of the devices connected to each interface without requiring manual configuration.

▼ What are the features of switches?

Switches offer several advantages over broadcast links, such as buses or hub-based star topologies:

1. Elimination of collisions: Switches buffer frames and only transmit one frame at a time, eliminating collisions and maximising bandwidth utilisation. This improves performance compared to LANs with broadcast links.
2. Heterogeneous links: Switches can connect different links operating at different speeds and using different media. This allows for the integration of legacy and new equipment, making switches ideal for mixed environments.
3. Management: Switches provide enhanced security and ease network management. They can detect and isolate malfunctioning adapters, preventing network disruptions. Switches also gather statistics on bandwidth usage, collisions, and traffic types, helping network administrators debug and plan network improvements.

▼ What is switch poisoning?

- attacker fload switch with tons of frames with bogus MAC address to fill up the switch table with bogus entries, leaving no room for MAC address of legitimate hosts
- hence switch'll broadcast most frames, allowing the attacker to sniff the frames

▼ Router vs switch

Table 6.1 Comparison of the typical features of popular interconnection devices

	Hubs	Routers	Switches
Traffic isolation	No	Yes	Yes
Plug and play	Yes	No	Yes
Optimal routing	No	Yes	No

▼ Pros and cons of switches:

- Plug-and-play functionality.
- High filtering and forwarding rates.
- Simplified network management.
- Suitable for LAN environments.
- Restricts network topology to a spanning tree.
- Requires large ARP tables.
- Susceptible to broadcast storms that can cause network collapse.

▼ Pros and cons of routers:

- Provides flexibility in routing paths.
- Supports hierarchical addressing.
- Offers firewall protection against broadcast storms.
- Suitable for larger networks and interconnecting different subnets.
- Allows for more diverse network topologies.
- Requires manual configuration of IP addresses.

        - Higher per-packet processing time due to layer-3 processing.
        - Not as easily deployable as switches.
    - For large networks, routers are generally a better choice due to their advanced traffic management capabilities, routing functionality, and ability to handle complex network topologies. Routers can effectively manage and route traffic between different network segments, provide security features like firewall protection, and support hierarchical addressing schemes.
    - On the other hand, for small networks with fewer hosts and simpler requirements, switches are often sufficient. Switches provide efficient local traffic handling, increased bandwidth availability, and simplified network management. They are cost-effective, easy to deploy, and offer high-performance connectivity within a local network segment.

▼ 6.4.4 Virtual Local Area Networks (VLANs)

▼ Using multiple switches such in figure 6.15 have drawbacks:

1. **Traffic isolation**: VLANs solve the problem of broadcast traffic affecting the entire network by creating separate virtual networks within a single physical network. Each VLAN has its own broadcast domain, limiting broadcast traffic to VLAN members only.
2. **Efficient use of switches**: VLANs allow different groups or departments to share the same physical switch, reducing the need for multiple switches and optimising network resources while maintaining traffic isolation.
3. **User management**: VLANs make it easier to manage user movements between groups or departments. Instead of physically rewiring connections, network operators can reconfigure VLAN assignments using software, simplifying user management tasks.

- Ports on a switch can be grouped into VLANs, creating separate broadcast domains, and frames are only delivered between ports within the same VLAN.

▼ How can different VLAN communicate?

- VLAN switch port can be connected to external router, and configure the router's port to belong to both VLAN — in logical view, it is as if there are 2 switches connected to a router (there are device that contains both VLAN switch and router, so no external router is needed)

▼ How to connect multiple VLAN switches?

- Via VLAN Trunking
- one port on each switch can be set to Trunk
- Trunk port'll carry frames from all VLAN

▼ How does a switch know to which VLAN that a *frame arrives on trunk port* belong to?

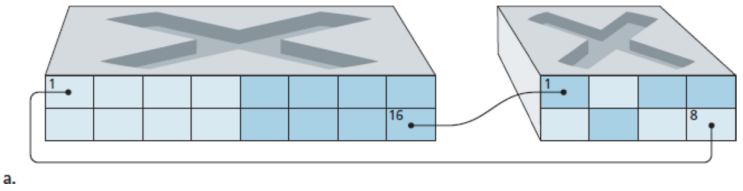
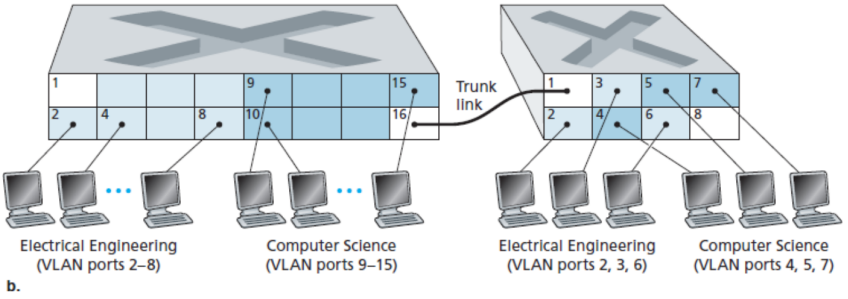


Figure 6.26 Connecting two VLAN switches with two VLANs: (a) two cables (b) trunked



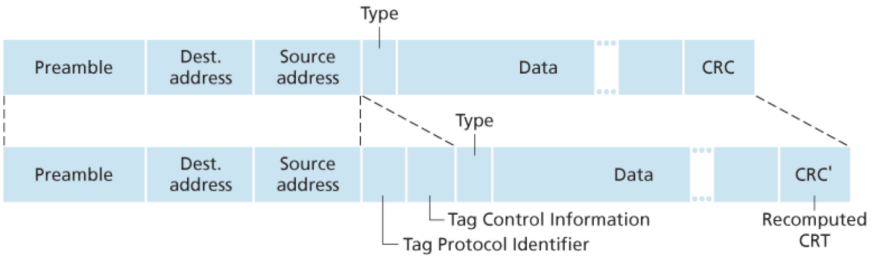


Figure 6.27 Original Ethernet frame (top), 802.1Q-tagged Ethernet VLAN frame (below)

- IEEE has defined extended the Ethernet frame format, 802.1Q for frames crossing a VLAN trunk.
- Four-bytes VLAN tag (Tag Protocol Identifier & Tag Control Information) is added to the header by the sending switch, then removed by the receiving switch

▼ What are the VLAN tags?

- 2-bytes Tag Protocol Identifier (TPID)
- 2-bytes Control Information - 12-bit VLAN Identifier
- 3-bit priority field, same as TOS field in IP datagram

▼ 6.5 Link Virtualisation: A network as a Link Layer

▼ 6.5.1 Multiprotocol Label Switching (MPLS)

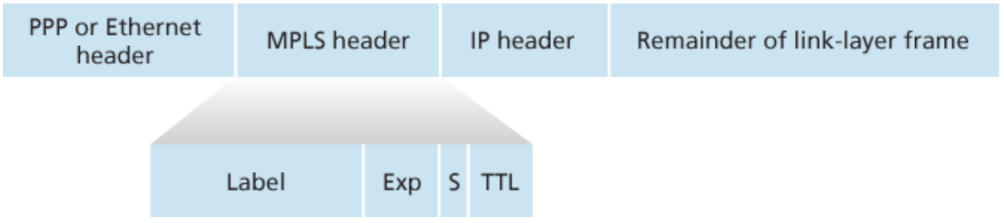


Figure 6.28 MPLS header: Located between link- and network-layer headers

- it’s a protocol used in computer network to enhance the packets forwarding
- Instead of forwarding based on the IP address, MPLS allow for a predetermined route based on the label on the datagram
- MPLS headers are added to link-layer frames between MPLS-capable devices (aka label-switched router)
- Allow for VPN - each customer's network is logically isolated from others, creating a private network environment. This isolation is achieved by assigning unique MPLS labels to the customer's traffic, effectively separating it from other traffic within the provider's network. The customer's IP addresses are used within their own private network, but they are not exposed or visible to the service provider's network or other VPN customers. The MPLS infrastructure uses labels to forward the customer's traffic instead of relying on the customer's IP addresses for routing

▼ 6.7 Full Scenario of a Web Page Request

▼ Short Version

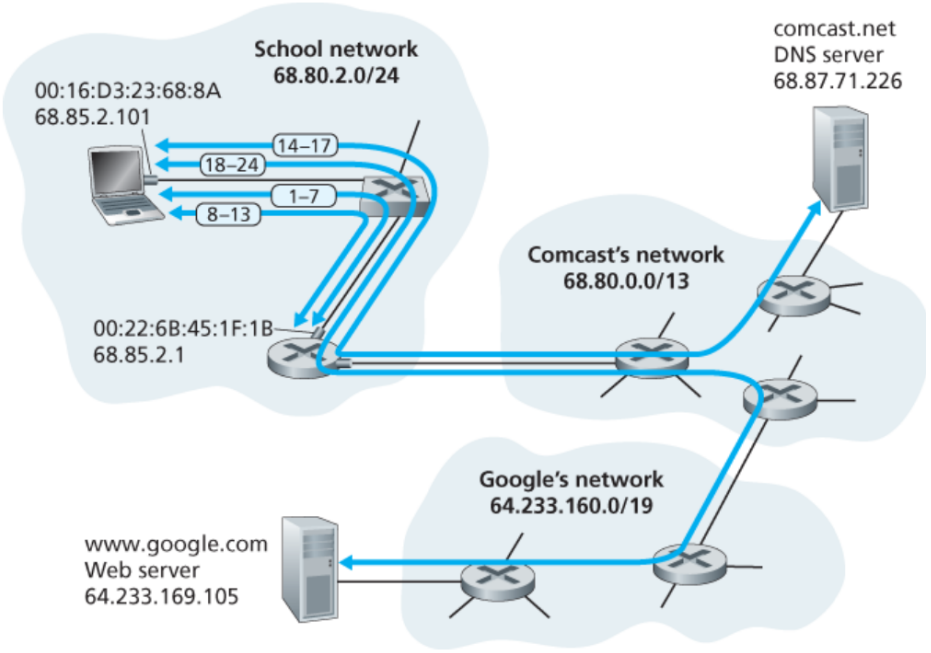


Figure 6.32 A day in the life of a Web page request: Network setting and actions

1. Bob's laptop connects to the school's Ethernet switch via an Ethernet cable.
2. Bob's laptop uses DHCP to obtain an IP address from the school's router (MAC: 00:22:6B:45:1F:1B, IP: 68.85.2.1).
3. Bob's laptop sends a DHCP request message (UDP) to the DHCP server (MAC: FF:FF:FF:FF:FF:FF) to obtain an IP address.
4. The DHCP server allocates an IP address (e.g., 68.85.2.101) to Bob's laptop and sends a DHCP ACK message (UDP) with network information, including DNS server (68.87.71.226), default gateway (68.85.2.1), and subnet block (68.85.2.0/24).
5. Bob's laptop wants to access www.google.com.
6. Bob's laptop creates a DNS query message (UDP) for www.google.com and sends it to the DNS server (68.87.71.226) via the gateway router (MAC: 00:22:6B:45:1F:1B, IP: 68.85.2.1).
7. Bob's laptop sends an ARP query message (Ethernet) to obtain the MAC address of the gateway router.

8. The gateway router replies with an ARP reply message (Ethernet) containing its MAC address (00:22:6B:45:1F:1B).
9. Bob's laptop receives the ARP reply and addresses the Ethernet frame containing the DNS query to the gateway router (MAC: 00:22:6B:45:1F:1B, IP: 68.85.2.1).
10. The gateway router forwards the DNS query to the DNS server.
11. The DNS server responds with a DNS reply message (UDP) containing the IP address of www.google.com (e.g., 64.233.169.105).
12. Bob's laptop wants to retrieve the web page from www.google.com.
13. Bob's laptop creates a TCP socket and performs a three-way handshake with www.google.com.
14. The TCP SYN, SYNACK, and ACK segments are exchanged between Bob's laptop (MAC: 00:16:D3:23:68:8A, IP: 68.85.2.101) and www.google.com (MAC: unknown, IP: 64.233.169.105).
15. Bob's laptop creates an HTTP GET message and sends it to www.google.com through the TCP connection.
16. The HTTP server at www.google.com processes the request and sends back an HTTP response message containing the web page content.
17. Bob's laptop receives the HTTP response message and displays the web page.

Route taken: Bob's laptop -> School's Ethernet switch -> School's router (MAC: 00:22:6B:45:1F:1B, IP: 68.85.2.1) -> Gateway router (MAC: 00:22:6B:45:1F:1B, IP: 68.85.2.1) -> DNS server (IP: 68.87.71.226) -> www.google.com (IP: 64.233.169.105)

▼ Original Version

6.7.1 Getting Started: DHCP, UDP, IP, and Ethernet

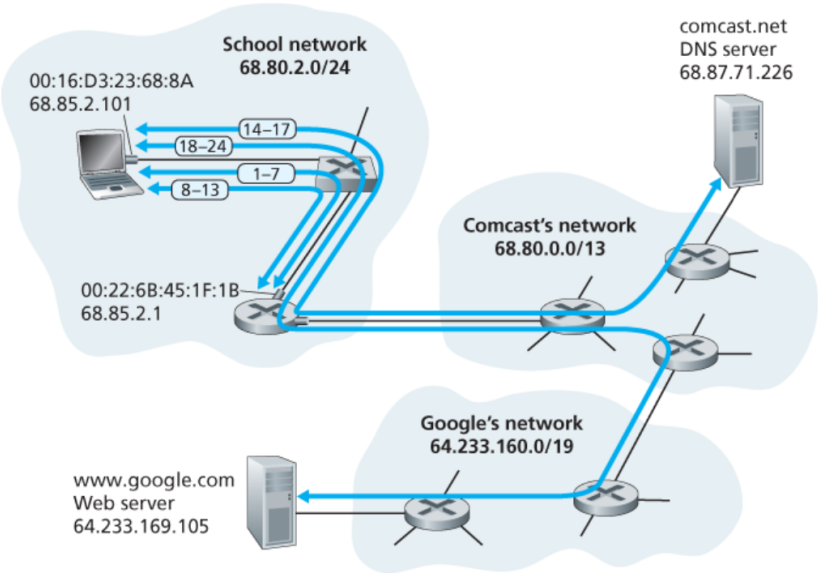


Figure 6.32 A day in the life of a Web page request: Network setting and actions

Let's imagine that Bob boots up his laptop and connects it to an Ethernet cable connected to the school's Ethernet switch, which is connected to the school's router. The school's router is connected to an ISP (comcast.net), which provides the DNS service. The DHCP server is running within the router.

1. Bob's laptop needs an IP address to connect to the network, so it sends a DHCP request message to the DHCP server.
2. The DHCP request is sent as a UDP segment within an IP datagram. The source IP address is 0.0.0.0, and the destination IP address is a broadcast address (255.255.255.255).
3. The IP datagram containing the DHCP request is placed within an Ethernet frame. The frame is broadcast to all devices connected to the switch, with the destination MAC address set to FF:FF:FF:FF:FF:FF.
4. The router receives the broadcast Ethernet frame and extracts the IP datagram.
5. The DHCP server within the router allocates an IP address (e.g., 68.85.2.101) to Bob's laptop and sends a DHCP ACK message as a UDP segment.
6. The DHCP ACK is sent in a unicast Ethernet frame from the router to Bob's laptop, using the laptop's MAC address.

7. Bob's laptop receives the Ethernet frame containing the DHCP ACK and records its IP address, DNS server IP address (e.g., 68.87.71.226), default gateway IP address (e.g., 68.85.2.1), and subnet block (68.85.2.0/24).
8. Bob's laptop needs to find the MAC address of the gateway router, so it sends an ARP query message with the target IP address of the gateway router (68.85.2.1).
9. The gateway router receives the ARP query and replies with an ARP reply message, providing its MAC address (00:22:6B:45:1F:1B).
10. Bob's laptop receives the ARP reply and extracts the MAC address of the gateway router.
11. Bob's laptop can now address Ethernet frames to the gateway router's MAC address. It sends a frame containing a DNS query to the gateway router, with an IP destination address of the DNS server (68.87.71.226).
12. The IP datagram within the frame is forwarded to the DNS server through the network.
13. The DNS server receives the DNS query, looks up the requested domain (www.google.com), and forms a DNS reply message with the corresponding IP address (e.g., 64.233.169.105). The DNS reply message is sent back to Bob's laptop using the laptop's IP address.
14. The DNS reply is placed in an IP datagram, encapsulated in an Ethernet frame, and forwarded back through the network to Bob's laptop.
15. Bob's laptop extracts the IP address of the server (www.google.com) from the DNS reply.

6.7.2 Still Getting Started: DNS and ARP

1. Bob's laptop creates a TCP socket to establish a connection with the HTTP server at www.google.com.
2. To create the socket, Bob's laptop needs the IP address of www.google.com. It sends a DNS query message to the DNS server (68.87.71.226), requesting the IP address of www.google.com.
3. The DNS query is encapsulated in a UDP segment, which is then placed in an IP datagram. The destination IP address is that of the

DNS server, and the source IP address is Bob's laptop's IP address (e.g., 68.85.2.101).

4. The IP datagram containing the DNS query is placed within an Ethernet frame and sent to the gateway router.
5. Bob's laptop doesn't know the MAC address of the gateway router, so it sends an ARP query to obtain it.
6. The gateway router receives the ARP query and replies with an ARP reply containing its MAC address.
7. Bob's laptop receives the ARP reply and extracts the MAC address of the gateway router.
8. Bob's laptop can now send an Ethernet frame containing the DNS query to the gateway router's MAC address. The IP datagram in the frame has the DNS server's IP address as the destination, and the frame's destination address is the MAC address of the gateway router.
9. The gateway router receives the frame, extracts the IP datagram, and forwards it to the DNS server.
10. The DNS server receives the IP datagram containing the DNS query, looks up the IP address for www.google.com, and sends a DNS reply to Bob's laptop.
11. The DNS reply is encapsulated in an IP datagram, placed within an Ethernet frame, and forwarded back to Bob's laptop.

6.7.3 Still Getting Started: Intra-Domain Routing to the DNS Server

1. The gateway router receives the frame, extracts the IP datagram containing the DNS reply, and determines that the datagram should be sent to the leftmost router in the Comcast network.
2. The IP datagram is placed in a link-layer frame appropriate for the link between the school's router and the leftmost Comcast router. The frame is sent over this link.
3. The leftmost router in the Comcast network receives the frame, extracts the IP datagram, and determines the outgoing interface to forward the datagram toward the DNS server.

4. The IP datagram eventually arrives at the DNS server, which processes the DNS reply and extracts the IP address for www.google.com.

6.7.4 Web Client-Server Interaction: TCP and HTTP

1. Bob's laptop now has the IP address of www.google.com and creates a TCP socket to establish a connection with the HTTP server.
2. Bob's laptop performs a three-way handshake with the TCP in www.google.com to establish the TCP connection.
3. Once the connection is established, Bob's laptop creates an HTTP GET message with the URL for the web page he wants to fetch.
4. The HTTP GET message is written into the TCP socket, becoming the payload of a TCP segment. The TCP segment is placed in an IP datagram and forwarded to www.google.com.
5. The HTTP server at www.google.com receives the TCP segment, processes the HTTP GET message, and generates an HTTP response message with the requested web page content.
6. The HTTP response message is sent back to Bob's laptop using the established TCP connection.
7. The HTTP response is received by Bob's laptop, and his web browser program extracts the HTML content from the response.
8. Finally, Bob's web browser displays the web page retrieved from www.google.com.