ECE 358 Course Notes

Computer Networks

Michael Socha

4A Software Engineering
University of Waterloo
Spring 2018

Contents

1 Course Overview **1**
- 1.1 Logistics . . . 1
- 1.2 Overview of Topics . . . 1

2 Introduction - Computer Networks and the Internet **2**
- 2.1 What is the Internet? . . . 2
- 2.2 Network Edges . . . 2
  - 2.2.1 Access Networks and Physical Media . . . 2
  - 2.2.2 Digital Subscriber Line (DSL) . . . 2
  - 2.2.3 Cable Network . . . 3
  - 2.2.4 Ethernet . . . 3
  - 2.2.5 Wireless Access Networks . . . 3
  - 2.2.6 Data Packet Transmission . . . 3
  - 2.2.7 Physical Media . . . 3
  - 2.2.8 Coaxial Cable . . . 4
  - 2.2.9 Fiber Optic Cable . . . 4
  - 2.2.10 Radio . . . 4
- 2.3 Network Core . . . 4
  - 2.3.1 Packet Switching . . . 4
  - 2.3.2 Routing vs Forwarding . . . 4
  - 2.3.3 Circuit Switching . . . 4
  - 2.3.4 Packet Switching vs Circuit Switching . . . 5
  - 2.3.5 Internet Structure . . . 5
- 2.4 Delay, Loss and Throughput . . . 5
  - 2.4.1 Sources of Packet Delay . . . 5
  - 2.4.2 Queuing Delay . . . 5
  - 2.4.3 Measuring Delay and Loss . . . 6
  - 2.4.4 Throughput . . . 6
- 2.5 Protocol Layers and Service Models . . . 6
  - 2.5.1 Protocol Layers . . . 6
  - 2.5.2 Open System Interconnection (OSI) Model . . . 6
- 2.6 Security . . . 7
  - 2.6.1 Types of Transmission . . . 7
  - 2.6.2 Examples of Attack . . . 7

3 Application Layer **8**
- 3.1 Network Application Introduction . . . 8
- 3.2 Network Application Architectures . . . 8
  - 3.2.1 Client-Server Architecture . . . 8
  - 3.2.2 Peer-to-Peer (P2P) Architecture . . . 8
- 3.3 Process Communication . . . 8
  - 3.3.1 Client vs Server Processes . . . 8
- 3.4 Sockets . . . 9

3.5 Addressing Processes . . . 9
3.6 App-layer protocols . . . 9
3.7 Transport Services . . . 9
3.7.1 Data Integrity / Reliable Data Transfer . . . 9
3.7.2 Throughput . . . 9
3.7.3 Timing . . . 10
3.7.4 Security . . . 10
3.8 Transport Services Provided by the Internet . . . 10
3.8.1 TCP . . . 10
3.8.2 Securing TCP . . . 10
3.8.3 UDP . . . 10
3.9 HTTP Overview . . . 10
3.10 HTTP Connections . . . 11
3.10.1 Non-Persistent Connections . . . 11
3.10.2 Persistent Connections . . . 11
3.11 HTTP Messages . . . 11
3.11.1 Requests . . . 11
3.11.2 Responses . . . 12
3.12 Cookies . . . 12
3.13 Web Caching . . . 12
3.14 Conditional GET . . . 13
3.15 DNS: Domain Name System . . . 13
3.15.1 Server Classes . . . 13
3.15.2 DNS Records . . . 14
3.15.3 DNS Messages . . . 14
3.15.4 Inserting Records into DNS Database . . . 14
3.15.5 DNS Vulnerabilities . . . 15

4 Transport Layer **16**
4.1 Transport vs Network Layer . . . 16
4.2 Multiplexing and Demultiplexing . . . 16
4.2.1 Demultiplexing . . . 16
4.2.2 Multiplexing . . . 16
4.3 Connectionless Transport: UDP . . . 17
4.3.1 UDP Segment Header . . . 17
4.3.2 UDP Checksum . . . 17
4.4 Principles of Reliable Data Transfer . . . 17
4.4.1 RDT over Perfectly Reliable Channel . . . 17
4.4.2 RDT over Channel with Bit Errors . . . 18
4.4.3 RDT over Channel with Errors and Loss . . . 18
4.4.4 Pipelined RDT Protocols . . . 18
4.5 Connection-Oriented Transport: TCP . . . 19
4.5.1 Overview . . . 19
4.5.2 Segment Structure . . . 19
4.5.3 Timing Estimation . . . 20

4.5.4 Reliable Data Transfer . . . 20
4.5.5 Flow Control . . . 21
4.5.6 Connection Management . . . 21
4.6 Principles of Congestion Control . . . 22
4.6.1 Causes of Congestion Control . . . 22
4.6.2 Approaches to Congestion Control . . . 22
4.7 TCP Congestion Control . . . 22
4.7.1 TCP Throughput . . . 23
4.7.2 TCP Fairness . . . 23

5 Network Layer **24**
5.1 Overview . . . 24
5.1.1 Forwarding and Routing . . . 24
5.1.2 Connection Setup . . . 24
5.1.3 Network Service Models . . . 24
5.2 Virtual Circuit and Datagram Networks . . . 25
5.2.1 Virtual Circuits (VCs) . . . 25
5.2.2 Datagram Networks . . . 25
5.2.3 VC vs Datagram Networks . . . 25
5.3 Router Architecture . . . 26
5.3.1 Input Processing . . . 26
5.3.2 Switching . . . 26
5.3.3 Output Processing . . . 26
5.4 IP: Internet Protocol . . . 27
5.4.1 IP Datagram Format . . . 27
5.4.2 IP Fragmentation . . . 27
5.4.3 IP Addressing . . . 28
5.4.4 Dynamic Host Configuration Protocol (DHCP) . . . 28
5.4.5 Network Address Translation (NAT) . . . 29
5.4.6 Internet Control Message Protocol (ICMP) . . . 29
5.4.7 IPv6 . . . 29
5.5 Routing Algorithms . . . 30
5.5.1 Routing Algorithm Classification . . . 30
5.5.2 Link-State Routing Algorithms . . . 30
5.5.3 Distance-Vector Routing Algorithm . . . 31
5.5.4 Comparison of Routing Algorithms . . . 31
5.5.5 Hierarchical Routing . . . 31
5.6 Routing in the Internet . . . 32
5.6.1 Routing Information Protocol (RIP) . . . 32
5.6.2 Open Shortest Path First (OSPF) . . . 32
5.6.3 Border Gateway Protocol (BGP) . . . 33
5.7 Broadcast and Multicast Routing . . . 33
5.7.1 N-Way-Unicast . . . 34
5.7.2 Uncontrolled Flooding . . . 34
5.7.3 Controlled Flooding . . . 34

6 Link Layer **35**
6.1 Services . . . 35
6.2 Error Detection and Correction . . . 35
6.2.1 Parity Checks . . . 36
6.2.2 Cyclic Redundancy Check . . . 36
6.3 Multiple Access Links and Protocols . . . 36
6.3.1 Channel Partitioning . . . 36
6.3.2 Random Access Protocols . . . 37
6.3.3 Taking-Turns Protocols . . . 38
6.4 Switched Local Area Networks (LANs) . . . 38
6.4.1 MAC Addresses and ARP . . . 38
6.4.2 Ethernet . . . 39
6.4.3 Link Layer Switches . . . 39
6.4.4 Virtual LANs . . . 40
6.5 Link Virtualization . . . 41
6.6 Data Center Networks . . . 41

1 Course Overview

1.1 Logistics

- **Professor:** Albert Wasef

1.2 Overview of Topics

This course focuses on the fundamentals of networking and thinking like a network engineer. Specific topics covered by this course include:

- LAN technologies and underlying protocols
- Transport protocols (TCP, retransmission)
- IP layer concepts (e.g. routing, addressing)
- Discrete-event simulation
- Network utilities

2 Introduction - Computer Networks and the Internet

2.1 What is the Internet?

The Internet is the world's largest computer network, connecting billions of devices. Devices connected to the Internet are known as hosts (end systems), and are running some kinds of network applications. Communication links are necessary for hosts to share information with each other, which can be done through a variety of means, including cables (e.g. fiber, copper), radio, or satellite. Packet switches (e.g. routers, switches) are responsible for forwarding chunks of data through these communication links.

Standardized protocols are necessary for communication between hosts. Protocols define the format and order of messages sent as well as actions taken upon message transmission and reception. Sample protocols include TCP, IP, and HTTP. These standards are maintained by the IEFT (Internet Engineering Task Force).

The Internet can also be viewed from a more service-oriented perspective, since it can be used to provide services such as the Web, VoIP, email, etc. to applications. The Internet also provides a programming interface to applications to interact with connected hosts.

2.2 Network Edges

Edge devices provide some sort of entry point to a network. Examples include computers, mobile devices, and servers (often in data centers). Communication between devices on a network can be wired or wireless.

2.2.1 Access Networks and Physical Media

End systems can connect to an edge router in various ways, including using residential access nets, institutional access networks and mobile access networks. Important considerations in such connections include a connection's bandwidth, latency, and whether it is shared/dedicated.

2.2.2 Digital Subscriber Line (DSL)

Network connections can be made through a digital subscriber line (DSL), which allows for the transmission of data over telephone lines. A digital subscriber line access multiplexer (DSLAM) can be used to connect multiple DSL lines to a digital communications channel. Downstream transmission rates (typically < 10 Mbps) tend to be much faster than upstream transmission rates (typically < 1 Mbps). Optimal transmission rates are rarely reached in practice. Each line connects directly to a central office.

2.2.3 Cable Network

Network connections can also be made through a cable network, which uses the same infrastructure as cable television. Differing frequencies are used to distinguish between different channels of communication. Hybrid Coaxial Cables (HFCs) are used to form the connection, which tend to have a downstream transmission around 30 Mbps and an upstream transmission around 2 Mbps. These connections attach to an ISP router, and multiple parties typically share access to a cable headend.

2.2.4 Ethernet

Most enterprise access networks use Ethernet connections, which tend to be much faster (available speeds include 10 Mbps, 100 Mbps, 1 Gbps, 10 Gbps). Nowadays, most end systems connect to an Ethernet switch.

2.2.5 Wireless Access Networks

Wireless access networks can connect end systems to routers without a cable connection. Wireless LANs (i.e. Wi-Fi) provide network access for a fairly small range, while wide-area access networks are provided by cellular operators and have a range of 10s of kilometers. Wireless LANs tend to have higher bandwidths than wide-area access networks.

2.2.6 Data Packet Transmission

A host sending function is responsible for:

- Taking an application message
- Breaking the message into chunks (packets) or length L bits
- Transmitting packets across a network at transmission rate R

packet transmission delay $= \frac{L}{R}$

2.2.7 Physical Media

The medium facilitating transmission between a transmitter and receiver is called a physical link. Physical links may be guided (i.e. solid cables, such as copper, fiber or coax), or unguided (i.e. signals may propagate freely, such as through radio).

2.2.8 Coaxial Cable

Coaxial cables are formed from two concentric copper conductors. Coaxial cables are bidirectional and they are are broadband, so they can support communication across multiple channels.

2.2.9 Fiber Optic Cable

Fiber optic cables feature a glass fiber carrying light pulses, where each pulse represents one bit. Fiber optics cables support high-speed point-to-point transmission, and have a low error rate.

2.2.10 Radio

Radio is a wireless bidirectional signal carried in the electromagnetic spectrum. The environment of propagation may cause signal reflection, obstruction (by objects in path) and interference. Radio link network types include terrestrial microwaves, LAN, wide-area and satellite.

2.3 Network Core

2.3.1 Packet Switching

Through store-and-forward packet-switching, an entire packet must arrive at a router before it can be transmitted on the next link. The resulting end-to-end delay is $\frac{2L}{R}$ (plus any propagation delay).

Should the arrival rate exceed a link's transmission rate, the resulting packets will queue up. If the memory in which the packets are stored fills up, packets can be dropped.

2.3.2 Routing vs Forwarding

Routing determines the source-destination route taken by packets, while forwarding moves packets to the appropriate output router.

2.3.3 Circuit Switching

Circuit switching is an alternative design for a network core. Instead of queuing up packets along shared lines, end-end resources between a transmission's source and destination are reserved (i.e. circuitry used only for that specific transmission). Such an approach is commonly used in telephone networks. Circuit switching can be implemented using FDM (frequency-division multiplexing) or TDM (time-division multiplexing).

2.3.4 Packet Switching vs Circuit Switching

Packet switching tends to be preferable to circuit switching for bursty data. Moreover, packet switching supports resource sharing, and the setup for calls is simpler than that of circuit switching. However, packet switching may have excessive congestion, resulting in packet delay and loss. Providing circuit-like behaviour (i.e. guaranteed bandwidth) to packet switching networks remains an unsolved problem.

2.3.5 Internet Structure

End systems typically connect to the Internet through access Internet Service Providers (ISPs). Since competing ISPs exist, they must be connected to one another as well as to their end hosts to effectively exchange packets. These connections are implemented using Internet Exchange Points (IXPs). Moreover, some content providers may setup their own networks (content provider networks) to connect their data centers to the Internet, often bypassing regional ISPs. The resulting Internet network structure is quite complex, with its evolution having been driven by a combination of business and politics.

2.4 Delay, Loss and Throughput

2.4.1 Sources of Packet Delay

$$d_{nodal} = d_{proc} + d_{queue} + d_{trans} + d_{prop}$$

- **Nodal processing** includes the time to check bit errors and determine the output link. This time is usually 10-1000us.
- **Queuing delay** is the time spent waiting at the output link for transmission.
- **Transmission delay** is $\frac{L}{R}$
- **Propagation delay** is d/s, where d is the length of the physical link and s is the propagation speed.

2.4.2 Queuing Delay

Let R be the link bandwidth, L be the packet length, and a be the average package arrival rate. The average queuing delay can be measured using $\frac{La}{R}$. If this value is close to 0, the queuing delay is small. Once the value exceeds 1, more packets are arriving than can be serviced, and the queuing delay may be infinite. Should packets be dropped, they may be re-transmitted by a previous node in the network, but this is not guaranteed.

2.4.3 Measuring Delay and Loss

The traceroute program can be used to measure the delay and loss on a network. It works by sending some test packets to a router which then returns the packets towards the sender.

2.4.4 Throughput

Throughput is the rate (bits/sec) at which bits are transferred between a sender and a receiver. Throughput can be measured as instantaneous (rate at a single point in time) or average (rate over a period of time). A bottleneck link is a link on an end-to-end path that constrains end-to-end throughput. For example, if one link on a network has a throughput of 20Mbps and another has a throughput of 50Mpbs, then the bottleneck link is the 20Mpbs link.

2.5 Protocol Layers and Service Models

2.5.1 Protocol Layers

Each layer implements some sort of service, and may rely on services provided by the layer below. Layering helps simplify dealing with complex systems. In particular, a layering system helps in identification of a system's components and developing a model of how different components interact. Layering also allows for a modular design, which eases maintenance and updating processes.

2.5.2 Open System Interconnection (OSI) Model

OSI is a conceptual model describing the layers of telecommunication or computing systems. The OSI layers are:

- **Application:** Supports network applications
- **Presentation:** Allows applications to interpret meanings of data (e.g. for encryption, compression)
- **Session:** Synchronization, checkpointing and recovery of data (i.e. controls connection between computers)
- **Transport:** Provides process-process data transfer
- **Network:** Handles routing from source to destination
- **Link:** Provides data transfer between directly connected nodes
- **Physical:** Deals with physical specs of connection

The Internet stack consists of the above layers except the presentation and session layer.

2.6 Security

The field of network security deals with how computer networks can be attacked, defending against such attacks and architectures that minimize the risk of attacks. Internet technologies were not designed at first with much security in mind, though security considerations have since been added across all networking layers.

2.6.1 Types of Transmission

- **Virus:** Self-replicating infection started by opening/executing object
- **Worm:** Self-replicating infection started by passively receiving object

2.6.2 Examples of Attack

- **Spyware:** Records user action (e.g. keystrokes, websites visited).
- **Botnet:** A collection of infected hosts running bots that can be used for spam, distributed denial of service (DDoS attacks), etc.
- **Denial of Service (DoS):** An attack where network resources (e.g. bandwidth) become unavailable to legitimate traffic due to the introduction of large amounts of bogus traffic. A distributed denial of service (DDos attack) performs a DoS attack using a collection of hosts.
- **Packet Sniffing** Promiscuous network connections having their data read by a third party as it passes by.
- **IP Address Spoofing** Packets being sent from a false source IP address, which can be used to hide one's identity or impersonate another host.

3 Application Layer

3.1 Network Application Introduction

Networks apps are designed to run on end systems, and use the network to communicate with other hosts. An example is a Web application, where the involved network apps are the browser running on the user host and the Web server program running in the Web server host. These network apps are only designed for end systems; network-core devices function on layers below the application layer.

3.2 Network Application Architectures

3.2.1 Client-Server Architecture

A client-server architecture features an always-on host, called the server, which services requests from many user hosts, known as clients. Servers respond to client requests by returning requested data to them. Clients only communicate with the web server, and not between each other. A server has a permanent IP addresses, while clients may have dynamic IP addresses.

3.2.2 Peer-to-Peer (P2P) Architecture

P2P networks do not feature an always-on server, but rather a collection of end systems that may communicate directly with one another; peers can request services from and provide services to other peers. These systems are self-scalable, with new peers being able to bring both new service capacity and new service demands. This type of architecture poses many challenges related to management and security.

3.3 Process Communication

A process is a program running within a host. If multiple processes run within the same host, they can communicate through inter-process communication procedures defined by the underlying OS. To communicate between hosts, processes need to exchange messages with one another.

3.3.1 Client vs Server Processes

Client processes run on client hosts, where they initiate communication. Server processes run on server hosts, where they wait to be contacted to serve client requests. In a P2P architecture, hosts may need to run both client and server processes.

3.4 Sockets

A host sends and receives messages on a network through a software interface called a socket, serving as the interface between the application and transport layer. Sockets are sometimes referred to as the API between an application and network.

3.5 Addressing Processes

In order to send a message across a network to a destination host, the address of that host and an identifier that specifies the receiving process (socket) must be provided. A host is identified by its IP address, while its socket is identified by a port number. Popular applications are linked to specific port numbers (e.g. Web server has port number 80).

3.6 App-layer protocols

An application-layer protocol is responsible for defining the following:

- Types of messages exchanges (e.g. request, response)
- Message syntax
- Message semantics
- Rules for when and how processes send and response to messages

App-layer protocols may be open (e.g. HTTP, SMTP) or may be proprietary (e.g. Skype)

3.7 Transport Services

Candidate transport services can be evaluated along four main dimensions:

3.7.1 Data Integrity / Reliable Data Transfer

A protocol that guarantees that data sent between applications is delivered correctly and completely is said to provide reliable data transfer. Loss-tolerant applications (e.g. multimedia) do not require perfect data integrity.

3.7.2 Throughput

Applications that require a certain level of throughput to function correctly (e.g. multimedia) are described as bandwidth-sensitive. Applications that do not have strict throughput requirements are described as elastic.

3.7.3 Timing

Some applications (e.g. real-time chat, multiplayer games) may not function well if the time to communicate between source and destination applications exceeds a certain amount of time.

3.7.4 Security

Examples of security-related features a transport service can provide include encryption, enforcing authentication or ensuring data integrity.

3.8 Transport Services Provided by the Internet

The Internet makes two transport protocols available, namely TCP and UDP

3.8.1 TCP

TCP is a connection-oriented protocol, meaning that the client and server exchange transport-layer control info before application messages are exchanged. When the application finishes sending messages, it must tear down this connection. TCP also provides reliable data transfer. Also, for the welfare of the Internet in general rather than specific applications, TCP provides a congestion-control mechanism, which throttles sending processes when the network between the client and server is congested. Flow control is also provided.

3.8.2 Securing TCP

Neither TCP or UDP provide any encryption. To remedy this issue, an enhancement for TCP, known as Secure Sockets Layer (SSL), can be used to provide process-to-process security services, including encryption, data integrity and end-point authentication.

3.8.3 UDP

UDP is a lightweight transport protocol. Unlike TCP, UDP is not connection-oriented, and does not provide reliable data transfer. Also, UDP does not provide a congestion-control mechanism.

3.9 HTTP Overview

HyperText Transfer Protocol (HTTP) is the Web's main application-layer protocol. HTTP can be used to load web pages, which consist of objects than can include HTML files, image

files, Java applets, etc.. Most Web pages consist of a base HTML file that references other objects. Each object is addressable by a Uniform Resource Locator (URL), which includes a host name and a path name.

HTTP uses TCP as its underlying transport protocol. HTTP clients (e.g. Web browsers) are responsible for initiating a TCP connection with a server, after which the two hosts can exchange messages through their socket interface. HTTP is considered to be a stateless protocol, meaning that HTTP servers are not required to maintain information about past client requests.

3.10 HTTP Connections

3.10.1 Non-Persistent Connections

In non-persistent HTTP connections, at most one object is sent over each connection, after which the connection is closed and a new one must be established. These types of connections have significant overhead, with TCP connection variables having to be stored on both the client and webserver, and each object suffering from a delivery delay of 2 Round-Trip Times (RTTs).

3.10.2 Persistent Connections

Persistent HTTP connections allow for a multiple objects from the same host to be sent over a single connection. Therefore, it is possible to have as few as one RTT of overhead for all objects on a server. This connection remains open until a configurable timeout interval, after which it is closed. The default mode of HTTP makes use of persistent connections.

3.11 HTTP Messages

3.11.1 Requests

Common request types:

- **GET:** Used to request an object, with the requested object identified in the URL (most common type of request)
- **POST:** Used to submit a request with an entity body (e.g. to submit data from a filled out form)
- **HEAD:** Similar to get, but leaves out the requested object
- **PUT:** Used to upload an object to a specific path
- **DELETE:** Used to delete an object on a specific path

3.11.2 Responses

Common response codes:

- **200 OK:** Request succeeded, requested object later in this message.
- **301 Moved Permanently:** Resource has been moved, new URL in Location: header of this message.
- **400 Bad Request:** Request message not understood by server.
- **404 Not Found:** Requested object not found on server.
- **505 HTTP Version Not Supported:** HTTP protocol version not supported by server.

3.12 Cookies

Although HTTP servers are stateless, it is often useful to be able to identify users, which can be done using cookies. Cookies technology has 4 components, namely:

1. Cookie header line in HTTP response message
2. Cookie header line in HTTP request message
3. Cookie stored on user end system
4. Backend database on website indexed by cookies

The key idea behind cookies it that they can be assigned to users when they first visit a website, after which the user can be identified because they include the cookie in their HTTP requests. Sample uses of cookies include authorization and saving user state and settings.

3.13 Web Caching

The goal of web caches (also known as proxy servers) is to satisfy a client request without involving the origin server. Instead, a client sends messages to a proxy that either already cached the results, in which case it responds to the request, or does not have the result, in which it sends a request to the origin server. Note that the proxy is acting as both a server and a client.

The main benefits of web caching are reducing response time (effective when client has a high-speed connection to the cache) and reducing traffic. Caches are typically installed by an ISP (e.g. a university might install a cache and configure all clients to point to it).

3.14 Conditional GET

A conditional GET can be used by web caches to ensure they are not returning state data. Conditional GETs use GET requests but include an If-modified-since: header. If a resource has not been modified since the specified time, the response code is 304 instead of 200.

3.15 DNS: Domain Name System

DNS serves to provide a mapping between host names and their IP addresses. DNS implements this through a distributed database implemented in a hierarchy of servers known as name servers. DNS is used by the application layer, where hosts communicate with a DNS server to resolve names, and can then proceed with the request.

DNS services include:

- Hostname to IP translation
- Host aliasing
- Mail server aliasing
- Load distribution (i.e. cycling through replicated web servers to avoid overloading a single one)

Although a single centralized DNS may sound like an appealing idea, it would face many problems, including:

- Single point of failure - DNS server going down would make the Internet unusable
- Huge traffic volume to a single server
- Server would be physically distant from most users
- Maintenance concerns - the DNS database would be huge and would require very frequent updates

3.15.1 Server Classes

- **Root name servers** are contacted by a local name server if a name cannot be resolved. Root name servers may contact servers lower on the hierarchy in resolving the name.
- **Top-level domain (TLD) servers** are responsible for top-level domains such as com, org, net, and country top-level domains.
- **Authoritative DNS Servers** are an organization's own DNS servers, providing authoritative hostname to IP mappings for their named hosts. These servers can be maintained by an organization itself or by a service provider.

- **Local DNS Name Servers** (also known as default name servers) serve as local proxies to DNS requests provided by an ISP. Local DNS Name Servers store a cache of name-to-address translation pairs, and in the case of a cache miss, are able to forward a request into the DNS hierarchy. Once forwarded into the DNS hierarchy, querying can be iterative (each server returns the next server to query to the local DNS server), or recursive (the root DNS server initiates a series of calls down the server hierarchy and ultimately returns the resolved name).

3.15.2 DNS Records

DNS servers store resource records (RRs), including RRs that map hostnames to IPs. An RR contains the following fields: (Name, Value, Type, TTL (time-to-live)). The record types are as follows:

- **Type=A** Name is hostname and Value is IP address.
- **Type=NS** Name is domain and Value is host-name of authoritative DNS server that can return IP address for hosts in the domain.
- **Type=CNAME** Name is alias and Value is the corresponding canonical hostname.
- **Type=MX** Name is alias and Value is the corresponding mail server. "MX" stands for "mail exchange".

3.15.3 DNS Messages

Query and reply are the only kinds of DNS messages. The parts of a message are described below:

- **Header:** Contains message identifier, which is copied from a query into its reply. The header also includes a number of flags, including one for query or reply, whether recursion is desired, whether recursion is available, and whether the reply is authoritative.
- **Questions:** Includes name and type field for a query.
- **Answers:** Contains records that were queried for.
- **Authority:** Contains records of other authoritative servers.
- **Additional:** Additional info (e.g. for a Type MX reply, this contains the Type A record for the canonical hostname of the mail server).

3.15.4 Inserting Records into DNS Database

Domain names can be registered at commercial entities called registrars. Once the domain is confirmed to be available, a Type A DNS record and a Type MX DNS record are inserted.

3.15.5 DNS Vulnerabilities

DDoS attacks against root servers are generally ineffective, since root servers have packet filters, and even if they do go down, most local DNS servers store the IPs of TLD servers.

A potentially more dangerous line of attack would be to DDoS TLD servers. However, even if TLD servers go down, the impact of the outage would be mitigated by caching in local DNS servers.

Other potential lines of attack include man-in-the-middle attacks, in which DNS queries from hosts are intercepted and bogus replies returned, and DNS poisoning, where bogus replies are sent to a DNS server to fill its cache with incorrect records. DNS infrastructure can also be used to launch a DDoS attack by sending DNS queries with a spoofed source of the attack target to many authoritative servers, with the queries designed so that their response is larger than the original query (i.e. the attacker's DoS efforts get amplified).

4 Transport Layer

The transport layer allows for logical communication between applications running on different hosts. Logical communication means that the applications involved can interact with one another as though they were directly connected. Transport-layer protocols are implemented in end systems, with the sender side breaking up application messages into transport-layer packets (or segments), and the receiving side reassembling them into messages sent to the application layer.

4.1 Transport vs Network Layer

The transport layer is just above the network layer. The network layer provides logical communication between hosts, while the transport layer provides logical communication between processes on these hosts.

4.2 Multiplexing and Demultiplexing

Multiplexing and demultiplexing involves extending the network host-to-host delivery service to a transport-layer process-to-process delivery service.

4.2.1 Demultiplexing

Demultiplexing involves directing incoming transport segments to a particular socket. Each segment received by the host is packaged in an IP datagram, which contains header information that includes the destination and source port numbers. IP addresses and port numbers are used to direct the segment to the appropriate socket.

In connectionless demultiplexing, only the destination port number is involved in determining which socket should receive a segment; segments with the same dest port but different source IP addresses are sent to the same socket.

In connection-oriented demultiplexing, the source and destination IP addresses and ports are all considered, with the receiver using all four values to direct the segment to the appropriate socket.

4.2.2 Multiplexing

Multiplexing involves gathering data chunks from a host's sockets, encapsulating each chunk with header information that is later used in demultiplexing, and passing the segments to the network layer.

4.3 Connectionless Transport: UDP

UDP is a "bare bones" transport protocol that does just about as little as the transport layer can possibly do. The only major functionality added on top of IP is multiplexing/demultiplexing and some basic error checking. No "handshake" occurs between the sending and receiving processes before a segments are exchanged, making UDP a connectionless transport protocol.

A few potential benefits of UDP include:

- No connection establishment, lowering overall delay. DNS makes use of this property.
- Simpler to implement.
- Small header size.
- No congestion control.

4.3.1 UDP Segment Header

A UDP segment header consists of source port, dest port, segment length, and checksum fields. Each of these fields consists of 2 bytes, forming an 8 byte header.

4.3.2 UDP Checksum

The goal of a UDP checksum is to detect errors in the transmitted segment. The checksum is formed by treating the UDP segment as a sequence of 2 byte words, summing them up (with overflow bits wrapping around and being added to the front), and taking the 1s complement. The receiver re-computes the checksum, and if it differs from the sent one, it can detect that an error occurred.

4.4 Principles of Reliable Data Transfer

Reliable data transfer (rdt) is an extremely important problem encountered in networking not only at the transport layer, but also at the link layer and application layer. Reliable data transfer is provided by a reliable data transfer protocol, and must work even if the layers below it are unreliable. This section covers unidirectional rdt; bidirectional rdt is conceptually similar but more tedious to explain.

4.4.1 RDT over Perfectly Reliable Channel

This is a trivial case where no error checking is necessary, since errors do not occur in the first place.

4.4.2 RDT over Channel with Bit Errors

Three new mechanisms can be added to help overcome bit errors:

1. **Error detection**, which can be done through computing a checksum.
2. **Receiver feedback**. Acknowledgments (ACKs) can be sent to the sender to notify them a packet was received ok, and negative acknowledgments (NAKs) can be sent to the sender to notify them of an error.
3. **Retransmission**. Upon receiving a NAK, the sender can re-send the packet.

This described protocol is known an example of a stop-and-wait protocol, since a sender will not send a new message until it receives acknowledgment that the receiver has successfully received the previous one.

A major limitation of the system above is that it assumes the ACK and NAK signals are not corrupted. This can be solved by a sender re-transmitting a packet if a corrupted ACK or NAK is received. A sequence number must also be added so that duplicates can be discarded by the receiver.

Moreover, if sequence numbers are used, NAKs do not need to be sent, since messages that were not acknowledged successfully are the ones without an ACK corresponding to their sequence number.

4.4.3 RDT over Channel with Errors and Loss

The possibility of packet loss over a channel adds to problems of packet loss detection and handling. This can be addressed by the sender using a countdown timer mechanism for each packet, which will trigger the packet to be re-sent if an ACK signal is not received on time. Since it is possible that a packet is not lost but merely delayed, this opens up the possibility of duplicate packets being sent, but this is already handled through packet sequence numbers.

4.4.4 Pipelined RDT Protocols

Although the protocol described above correctly handles packet error and loss, it has incredibly poor performance due to its stop-and-wait behaviour. Pipelined RDT protocols can be applied to overcome this stop-and-wait limitation by allowing multiple "in-flight", yet to be acknowledged packets. As a general rule, these protocols require that a larger range of sequence numbers be used, and that buffering be supported at the sender or receiver.

Go Back N (GBN) protocols allow for a sender to have up to N unacknowledged packets in transport at a given time. The receiver sends cumulative acknowledgments, meaning that packets are only acknowledged if they arrive in order. Should a sender timer for receiving an

ACK for a packet expire, the next N packets will be retransmitted. GBN is often referred to as a sliding window protocol, with N referred to as the window size.

Selective Repeat protocols differ from GBN protocols in that they allow the receiver to acknowledge received packets individually. Like in GBN, a window size of N is used to limit the number of unacknowledged packets sent. However, unlike in GBN, the sender may have already received AKCs for some of the packets in this window, since the receiver acknowledges received packets regardless of whether they are in order. Out-of-order packets are buffered in the receiver until the preceding packets are received, after which they can be delivered to the upper layer. A timeout mechanism is also used by the sender for every packet.

4.5 Connection-Oriented Transport: TCP

4.5.1 Overview

TCP is a connection-oriented transport protocol because before a sender and receiver can exchange information, some sort of "handshake" between them must first occur. The TCP protocol runs only on the end systems, and its state is not stored in the intermediary network elements. TCP supports full duplex (i.e. bi-directional) point-to-point (i.e. one sender, one receiver) transport.

4.5.2 Segment Structure

When TCP messages are sent, they are often broken up into chunks of the maximum segment size (MSS), or potentially into smaller chunks for interactive applications. The header of a TCP segment includes:

- Source and dest port numbers
- Sequence number and acknowledgment number, which are used in implementing a reliable data transfer service
- Header length, which specifies length of TCP header in words (length can vary due to options field)
- Flag fields (ACK bit for acknowledgments, RST, SYN and FIN bits for connection setup and teardown, PSH bit indicating receiver should push data to upper layer immediately, and URG used to indicate whether there is urgent data)
- Receive window, which is used for flow control
- Checksum field, as in UDP
- Urgent data pointer indicating the last word of urgent data

The sequence number is the byte-stream number of the first byte in a segment. The acknowledgment number is the number of the next byte expected from the sender. TCP provides cumulative acknowledgments with respect to the bytes in a stream, since it only acknowledges bytes up to the first missing one. Upon receiving an out-of-order segment, the TCP protocol allows for either discarding the segment (rarely used in practice) or buffering it (more common).

4.5.3 Timing Estimation

TCP makes use of a timing mechanism to recover from potentially lost segments. If the timeout value is too short, many unnecessary retransmissions are likely, while if it is too long, the reaction to segment loss will be too slow.

Estimating the RTT can help in determining a good timeout value. Since RTT can fluctuate significantly for specific samples, it is typically estimated with a moving average of previous RTT values:

$$RTT_{Estimated} = (1 - \alpha) \cdot RTT_{Estimated} + \alpha \cdot RTT_{Sample}$$

A typical value for α is 0.125.

The level of variability in the RTT can be measured as follows:

$$Dev_{RTT} = (1 - \beta) \cdot Dev_{RTT} + \beta \cdot |RTT_{Sample} - RTT_{Estimated}|$$

A typical value for β is 0.25.

The TCP retransmission timeout interval is defined as:

$$TimeoutInterval = RTT_{Estimated} + 4 \cdot Dev_{RTT}$$

4.5.4 Reliable Data Transfer

TCP provides rdt on top of IP's unreliable service. Some properties of TCP's rdt mechanism include pipelined segments, cumulative ACKs, and a single retransmission timer. Retransmissions are triggered by timeout events and duplicate ACKs.

The following sender events can take place:

- **Receiving data from the application layer.** In this case, a segment is created with the appropriate sequence number. Also, if the sender's single timer is not already running for some other segment, the timer is started.
- **Timeouts**, which result in a retransmission of the segment that caused the timeout, as well as a restart of the timer.
- **Arrival of ACK**, which updates the segments known be the ACKed (using cumulative ACK logic), and if there are no unACKed segments remaining, restarts the timer.

The following receiver events can take place:

- **Arrival of in-order segment.** If all data up to the expected sequence number is already ACKed, and the receiver waits 500ms for the next segment. If it does not arrive, then an ACK is sent. If there is another segment with an ACK pending, a single cumulative ACK is immediately sent.
- **Arrival of out of order segment with higher than expected sequence number.** A duplicate ACK is immediately sent, indicating the sequence number of the expected byte.
- **Arrival of segment that partially or completely fills in lower end of gap.** An ACK is immediately sent for the segment.

Duplicate ACKs are ACKs that reacknowledge a segment for which a sender has received a previous ACK. TCP does not use NAK signals, so if a segment arrives out of order, an ACK signal is re-sent for the last in-order byte of data received. Should three duplicate ACKs be received, the sender can re-transmit the next segment even if the timer has not yet expired, which is known as fast retransmit.

4.5.5 Flow Control

TCP provides a flow control service to prevent the sender from overflowing the receiver's buffer. The available buffer size is exchanged though the receive window variable, and since TCP is full duplex, this information is sent by both the sender and the receiver.

4.5.6 Connection Management

A two-way handshake involves a host sending a connection request to another host, which in turn replies with a message. However, two-way handshakes can lead to deadlocks or half-open clients, so TCP makes use of a three-way handshake. In a three-way handshake, a host sets the SYN bit to 1 and sets an initial sequence number (isn). The receiver replies with a SYN bit of 1, an acknowledgment of the client isn + 1, an ACK bit of 1, and its own initial sequence number. This part of establishing a connection is known as the SYNACK segment. Once the original sender receives this message it replies with a sequence number of the server isn + 1, and sets the ACK bit to 1.

Any two hosts in a connection can close the connection, which involves sending a TCP segment with a FIN bit of 1. The receiver of that segment responds with an ACK bit of 1. The original sender then replies with a message to acknowledge the connection shutdown, after which the connection is closed and the resources for that connection can be deallocated.

4.6 Principles of Congestion Control

4.6.1 Causes of Congestion Control

Below is a list of three increasingly complex scenarios that describe sample causes of network congestion:

1. **Two senders and a router with infinite buffers.** If the throughput of the network does not keep up with the sending rates of the hosts, the delay will grow asymptotically.
2. **Two senders and a router with finite buffers.** Packet loss or long delays due to network congestion may lead to packet retransmissions, which in turn contribute further to network congestion.
3. **Multiple senders, routers with finite buffers, multihop paths.** Network congestion can cause delays or packet loss between different hosts that happen to be communicating through the same infrastructure (i.e. same router). More routers involved in delivering packets opens the possibility for more wasted effort, since since upstream effort in delivering a packet that ends up getting dropped later is wasted.

4.6.2 Approaches to Congestion Control

The general approaches to congestion control are:

- **End-to-end congestion control**, which is a form of congestion control handled only by end systems, which infer network congestion based on observed network behavior. This is the method of congestion control used by TCP.
- **Network-assisted congestion control**, which is a form of congestion control where network routers provide feedback on congestion to end systems. This feedback can range from something simple, like a single congestion bit, or something more complex like an explicit rate at which the sender should send.

4.7 TCP Congestion Control

Since the IP layer provides no feedback regarding network congestion, TCP implements an end-to-end congestion control mechanism. TCP congestion control introduces a new variable known as a congestion window (cwnd), which limits the number of unACKed packets sent by a sender (should be the minimum of receive window and congestion window).

When a TCP connection is first established, the initiation cwnd is set very low (1 minimum segment size (MSS)). However, on each successful ACK, this segment size doubles. Once some packet loss in indicated by a triple ACK being received for the same packet, cwnd is either halved and then grows linearly (the newer TCP Reno, which implements this "fast recovery") or is set back to 1 (the older TCP Tahoe). In general, the cwnd increase should switch from exponential to linear once it reaches half of its previous value before timeout.

4.7.1 TCP Throughput

Once a connection experiences packet loss, fast recovery implementations of TCP halve its value, after which it linearly grows back to its max value. Thus, ignoring the relatively short start phases, the average throughput of a connection is $\frac{3}{4} \cdot \frac{W}{RTT}$, where W is the window size where loss events begin to occur, and RTT is the round trip time.

When packet loss is factored in, the average throughput of a connection becomes around $\frac{1.22 \cdot MSS}{RTT\sqrt{L}}$, where L is the packet loss ratio. A key takeaway from this is that loss ratios often need to be very small in order for throughput to be close to its nominal maximum.

4.7.2 TCP Fairness

The goal of fairness is that K TCP connections sharing a link of bandwidth R should each have an average bandwidth of $\frac{R}{K}$. TCP's congestion control mechanism makes it fair, though since UDP does not implement congestion control, it does not adhere to such fairness. However, even with TCP, this fairness restriction can be circumvented by opening multiple parallel connections between the same hosts.

5 Network Layer

5.1 Overview

The network layer handles transporting segments from one host to another. Segments from the sending host are encapsulated into datagrams (i.e. network-layer packets), and segments at the receiving host have their transport-layer segments extracted and passed up to the transport layer.

5.1.1 Forwarding and Routing

To transport packets from one host to another, the network layer makes use of two main functions:

- **Forwarding**, which involves a router moving a packet from an input link to the appropriate output link.
- **Routing**, which involves the network layer applying a routing algorithm to determine the route taken by a packet between hosts.

Each router stores a forwarding table, which provides a map between a certain incoming header field and the output link to be used. Each received packet is looked up in this field to determine its output link.

5.1.2 Connection Setup

On top of forwarding and routing, some network architectures use a third function for connection setup. In such networks, before two end hosts can exchange data, the routers along the chosen path must also undergo some handshake procedure.

5.1.3 Network Service Models

A network service model defines some characteristics of end-to-end packet transport. Examples of such characteristics include:

- Whether delivery is guaranteed.
- Whether delivery delay is bounded.
- Whether packet delivery is in-order.
- Whether some minimum bandwidth is guaranteed.

5.2 Virtual Circuit and Datagram Networks

The network layer can either provide connection-oriented or connectionless service between two hosts. Computer networks that implement connection-oriented services are known as virtual-circuit (VC) networks, while computer networks that implement connectionless services are known as datagram networks. There may seem to be a parallel with connection-oriented and connectionless transport protocols. However, unlike with transport protocols, a network layer's service is implemented in the network core, and a network layer may provide only one of these types of services.

5.2.1 Virtual Circuits (VCs)

Virtual circuits implement a connection-oriented network layer service. A VC consists of a path between two hosts, with a number assigned for each link on the path. These numbers are used in forwarding tables to determine the outgoing link and also the outgoing VC number.

A VC circuit consists of the following three phases:

1. **VC Setup**, where the network layer determines the path between the sender and receiver, determines VC numbers, updates each router's forwarding table, and may reserve resources (e.g. bandwidth) along the path.
2. **Data Transfer**, where packets flow across the VC.
3. **VC Teardown**, which undoes the steps taken during setup.

Messages sent by end systems to initialize or terminate a VC are known as signaling messages, while protocols used to exchange these messages are known as signaling protocols.

5.2.2 Datagram Networks

Datagram networks implement a connectionless network layer service. Instead of storing connection state information, forwarding tables index by ranges of addresses. If addresses do not divide up nicely into ranges, longest prefix matches can be applied.

5.2.3 VC vs Datagram Networks

As a general rule, datagram networks are more elastic than VC networks, and can connect many hosts with different link types. Datagram networks tend to offer fewer service guarantees, and instead rely on "smart" end systems to perform flow control, error recovery, etc. The Internet is an example of a datagram network, while some older systems (e.g. ATM) use VC networks.

5.3 Router Architecture

A router has two key functions, which are forwarding (also known as switching) datagrams from input to output links, and running routing algorithms which are used to find paths between hosts. The main components of a router are input ports, output ports, switching fabric, which connect input ports to output ports, and a routing processor, which is a software component that runs routing algorithms.

5.3.1 Input Processing

A main forwarding table is managed by the routing processor. Each input port typically has a shallow copy of this table, which it uses to determine the appropriate output port. If datagrams arrive faster than the forwarding rate into the switch fabric, then they may be queued up. This can lead to a phenomenon knows as head-of-the-line (HOL) blocking, which occurs when a queued datagram at the front of a queue prevents others from moving forward, even when they have different destination ports.

5.3.2 Switching

Switching fabrics transfer datagrams from an input buffer to the appropriate output buffer. The three main switching mechanisms are:

- **Switching via shared memory.** First generation routers performed switching by using shared memory controlled by a CPU. Most new routers use shared memory but with most processing performed by line cards.
- **Switching via a bus.** This is typically done by an input port appending a switch-internal header storing the output port of a datagram, which is sent over a bus connected to all output ports. Only the output port the datagram is actually destined for processes the datagram. Bus contention between different datagrams is a common problem, making the switching speed ultimately limited by bus bandwidth.
- **Switching via interconnection network.** These networks form a grid of buses that is capable of forwarding multiple datagrams in parallel.

5.3.3 Output Processing

Output processing involves datagrams stored in an output port's memory being transmitted over the output link. Buffering may occur if the arrival rate from the switch exceeds the output line speed, which may result in delay or packet loss. A good recommendation for the amount of buffering needed is $\frac{RTT \cdot C}{\sqrt{N}}$, where C is the link capacity and N is the number of flows passing through the link.

5.4 IP: Internet Protocol

IP is a network-layer protocol used by the Internet. This section focuses on IPv4. The major components that make up the Internet's network layer are the IP protocol, routing components, and a facility to report errors in datagrams and respond to requests for certain network-layer information.

5.4.1 IP Datagram Format

IP datagram headers are at least 20 bytes long. The main fields in a an IP datagram are:

- **Version number:** Specifies IP protocol version.
- **Header length:** Used to determine where the data actually begins (necessary because header has variable-length options field).
- **Type of service:** Allows for different types of IP datagrams to be distinguished.
- **Datagram length:** Total length of IP datagram, including header data, measured in bytes. Maximum theoretical size is 65535 bytes, but datagrams are rarely larger than 1500 bytes.
- **Identifier, flags, fragmentation offset:** Support a concept known as IP fragmentation, where IP packets are broken down into smaller packets.
- **Time-to-live (TTL):** A counter that is decremented by one each time the datagram is processed by a router. Once the field reaches 0, the datagram is dropped, preventing it from circulating in a loop forever.
- **Upper-layer protocol:** Indicated transport-layer protocol to use at destination.
- **Header checksum:** Aids router in detecting bit errors.
- **Source and destination IP addresses:** Used to identify source and destination hosts.
- **Options:** Variable-length field used to store data such as timestamp, route taken, or routers to visit.
- **Data (payload):** Typically carries transport-layer segment to deliver to host.

5.4.2 IP Fragmentation

Different network links have different maximum transmission units (MTUs). Thus, it is sometimes necessary to break up large IP datagrams into several smaller ones known as fragments, which must be reassembled before being transferred to the transport layer.

The identifier, flags, and fragmentation offset fields are necessary for successful reassembly. An identifier is set when the datagram is sent from the source host. Fragments of this

datagram use the same identifier. The flag bit indicates which fragment is the last one in the datagram (the last flag is 0, while all others are 1). The offset field is used to specify where the fragment fits in the original datagram.

5.4.3 IP Addressing

An interface is a connection between a host or router to a physical link. Hosts often have only one interface, while routers tend to have many. IP addresses are 32-bit identifiers for each interface connected to the Internet.

Subnets are a logical subdivision of an IP network, and consist of devices that can reach each other without an intervening router. In IP addressing, the high-order bits are dedicated to identifying the subnet, while the low-order bits are dedicated to identifying the host.

The IP address assignment strategy applied on the Internet is known as Classless Interdomain Routing (CIDR). IP addresses can be expressed in dotted decimal form a.b.c.d/x, where the first x bits form the prefix. The prefix refers to the subnet portion of the IP address, while the remaining 32 - x bits are used to identify devices within a subnet. x can be of various sizes, and routers outside the subnet only need to store the first x bits of an address to perform forwarding. CIDR differs from classful addressing, where the value of x must fall into value classes of 1, 2 or 3 bytes.

Obtaining a block of IP addresses to use within a subnet is usually handled by an ISP. IP addresses are managed by the Internet Corporation for Assigned Names and Numbers (ICANN).

5.4.4 Dynamic Host Configuration Protocol (DHCP)

The Dynamic Host Configuration Protocol (DHCP) allows for hosts to dynamically obtain an IP address when joining a network. This provides an advantage over having to manually assign an address to each device, and also supports mobile users who frequently join and leave networks. The four main steps in the DHCP protocol are:

1. DHCP server discovery, where the DHCP host can broadcast a discover message.
2. The DHCP server can respond by broadcasting an offer message. This message contains the proposed IP address, the address lease time (i.e. time to expiry), and may include additional info such as the network mask or a DNS server address.
3. The host requests an IP address by sending the configuration parameters to one of the DHCP servers.
4. The server responds with a DCHP ACK message.

5.4.5 Network Address Translation (NAT)

Network Address Translation (NAT) allows for a local network to appear to the outside world as a single host (i.e. with one IP address). A few benefits of this technology include that a smaller range of IP addresses needs to be allocated from an ISP, device addresses are easy to change, and devices within a network are more secure when they are not explicitly addressable. NAT is implemented by NAT routers, which translate local device addresses and port numbers to the NAT router IP address and some port number.

Adaption of NAT was controversial for several reasons. For one, NAT uses port numbers to address hosts, instead of their original purpose of addressing processes. Moreover, NAT can prevent hosts in a P2P network from directly communicating with one another. However, this latter problem can often be overcome by the processes establishing connections to some other process not behind a NAT, which can then act as a relay the two original processes. Alternative solutions include statically configuring NAT routers to forward certain requests to a particular host and port, or dynamically altering entries in the forwarding table using the Internet Gateway Device (IGD) protocol.

5.4.6 Internet Control Message Protocol (ICMP)

Internet Control Message Protocol (ICMP) is used by hosts and routers to communicate network-level information, such as error messages, to one another. ICMP is technically a level above IP, with ICMP messages carried inside IP datagrams. ICMP messages have a type and code field, as well as the header and first eight bytes of the IP datagram that caused the message to be generated. The type and code contain much of the information concerning why the message was sent (e.g. type 3 and code 0 indicates destination network unreachable, type 3 and code 1 indicates destination host unreachable).

Traceroute uses ICMP to determine the path taken by packets through a network. This is done by sending packets with incrementing time-to-live values (first 1, then 2, etc.). When packets are dropped, ICMP messages indicating this are sent back to the source, which records information about the routers where the messages were dropped.

5.4.7 IPv6

So far, this section discussed features of IPv4. IPv6 is a newer version of the IP protocol initially motivated by the 32-bit address space running out. Additional motivations included increasing the speed with which packets are processed. A few new features of IPv6 include:

- 128 bit IP addresses.
- Header is fixed-length (40 bytes).
- Fragmentation and reassembly must be performed at source and destination, not at intermediate routers.

- Priority field in header.
- Flow label in header to identify datagrams in same "flow" (e.g. same video transmission).
- Next header field to identify upper layer protocol.
- ICMPv6, which supports some new message types (e.g. "packet too big").

Transitioning the Internet from IPv4 to IPv6 is a long process. One way to support adoption of IPv6 is a dual-stack approach, where IPv6 routers are also backwards-compatible with IPv4 routers. Alternatively, a tunneling approach can be used, where an IPv6 datagram may be carried as a payload by IPv4 datagrams.

5.5 Routing Algorithms

Router networks can be expressed as a graph where nodes represent routers and edges represent links between routers. This can be expressed as $G = (N, E)$, where G is the graph, N is the set of routers, and E is the set of links. A common problem in networking is finding the least-cost path between two routers. Cost can be defined in various ways (e.g. constant value, inverse of bandwidth, inverse of congestion).

5.5.1 Routing Algorithm Classification

Routing algorithms can be classified as global or decentralized. Global routing algorithms require complete knowledge of a network to calculate the lowest-cost path. Global routing algorithms are also known as link-state (LS) algorithms. Decentralized algorithms start with each node only having information on its surrounding links. The algorithm is then carried out in an iterative and distributed manner.

Secondly, routing algorithms can be classified as static or dynamic. Static algorithms tend to apply changes slowly over time, while dynamic algorithms change routing paths to react to changes in factors such as network traffic or topology.

5.5.2 Link-State Routing Algorithms

In practice, link-state algorithms can be applied once nodes in a network broadcast link-state packets to one another, providing a global view of the network. A common link-state routing algorithm is Dijsktra's algorithm, which finds the lowest-cost path from one source node to all other nodes. Efficient implementations of Dijkstra's algorithm have a runtime of $O(|E| + |N| \log |N|)$.

Note that making link costs depend on network congestion can lead to oscillations as traffic switches from a congested link to a non-congested one, which then quickly becomes congested.

One way to mitigate this problem is to stagger the times at which routers run the LS algorithm.

5.5.3 Distance-Vector Routing Algorithm

The distance-vector (DV) routing algorithm is a decentralized algorithm that is based on the Bellman-Ford algorithm. A key relation applied is that the minimum cost of a path between some nodes a and b is the minimum cost between a and some intermediate neighbor c plus c to b. This relation can be applied when setting up the forwarding table for a, since we know that the lowest-cost path to get from a to b is through c.

In this algorithm, each node contains the following routing information:

- Cost to directly attached neighbors
- Distance vector that estimates distances to all destinations
- Distance vector for each of its neighbors

The nodes asynchronously send their distance vectors to their neighbors. Once a distance vector is received, the Bellman-Ford relation above is applied to try finding lower-cost paths. If a lower-cost path is found, the changed node will send its new distance vector to its neighbors. This algorithm eventually converges on the actual least-cost path.

Whenever a link cost changes, the distance vectors of the adjacent nodes are recalculated. Due to the implementation of the distance-vector routing algorithm, decreases in cost tend to propagate faster than increases in cost. One technique that can sometimes quicken the propagation of increases in cost is known a poisoned reverse, which sets a link that routers should stop using to temporarily have infinite cost.

5.5.4 Comparison of Routing Algorithms

LS requires that each node know the cost of each link in a network, while in DV, nodes only exchange information with their neighbors. Also, LS converges in the time complexity specified priorly ($O(|E| + |N| \log |N|)$), while DV convergence time tends to vary more. LS tends to be more robust, since each node calculates a forwarding table for itself; in DV, malfunctioning nodes can propagate incorrect information deep into a network.

5.5.5 Hierarchical Routing

It is impractical to apply the algorithms above for an entire, large, "flat" network. For one, in a large network, is it infeasible to store all destinations in a routing table. Moreover, issues of administrative autonomy must be considered; different network administrators may wish to administer their network in different ways.

Hierarchical routing consists of aggregating routers into autonomous systems (ASs), each typically under control of the same administrator. Routers within the same AS can run the same intra-AS routing protocols, also known as interior gateway protocols. Routers responsible for forwarding packets to different ASs are known as gateway routers.

An inter-AS routing protocol is responsible for propagating information among routers of an AS about the reachability of a different AS. Forwarding tables for routers within a network are constructed using a combination of the intra-AS and inter-AS routing protocol. Choosing a gateway router to send packets to a different AS typically follows the these steps:

- Learn gateways that access different AS.
- Use routing info from intra-AS protocol to determine least-cost path to gateways.
- Apply hot-potato routing, where the packet is sent down the gateway with the lowest-cost path.

5.6 Routing in the Internet

5.6.1 Routing Information Protocol (RIP)

RIP is one of the earliest intra-AS routing protocols and remains in widespread use. RIP operates based on a distance-vector algorithm. The cost of a path is simply its number of hops (up to a max of 15) from a source router to a destination subnet. Routing updates are exchanged with neighbors using an RIP response message, also known as an RIP advertisement. This message contains the sender's distance to up to 25 destination subnets.

If no advertisement is heard from a router for 180 seconds, it is considered dead. The neighboring routing tables are updated, and this information propagates throughout the network. A poison reverse can be used to accelerate this propagation.

5.6.2 Open Shortest Path First (OSPF)

OSPF is an intra-AS link-state routing algorithm. Routers periodically advertise their state to the entire AS, and each router applies Dijkstra's algorithm to find the shortest path to each gateway router, where path cost is configurable. Exchanges between routers can be authenticated to ensure only trusted routers can participate. Another protocol known as Intermediate System to Intermediate System (IS-IS) is very similar to OSPF.

OSPF can also be organized hierarchically within an AS. There is typically a two-level hierarchy, which consists of local areas and a backbone. Routers between different areas are known is area border routers. The backbone, whose main purpose is to route traffic between local areas, is connected to all the area border routers.

5.6.3 Border Gateway Protocol (BGP)

BGP is the standard inter-AS routing protocol used for the Internet. BGP is known for being highly complex, so this section only provides a very high-level overview.

BGP supports TCP connections between routers in different ASs (external BGP sessions), as well as TCP connections between routers in the same AS (internal BGP sessions). These sessions are used to collect information on "good" routes to other ASs and propagate it within the AS. The types of messages exchanged during these sessions include:

- **OPEN:** Opens TCP connection between two peer routers.
- **UPDATE:** Advertises new path, or withdraws old path.
- **KEEPALIVE:** Keeps connection alive in absence of UPDATES, and also ACKs OPEN message.
- **NOTIFICATION:** Used to report errors in previous messages or close connection.

When a prefix (subnet or collection of subnets) is advertised across of BGP session, some of its key attributes include:

- **AS-PATH:** Contains ASs through which prefix advertisement traveled.
- **NEXT-HOP:** Router interface that begins AS-PATH.

When gateway routers receive a prefix advertisement, they apply an import policy to determine whether to accept or reject the route.

If a router receives multiple routes to a prefix, the router selects the "best" route by sequentially invoking these elimination rules:

1. Local preference value, which is setup by a network administrator
2. Shortest AS-PATH
3. Closest NEXT-HOP (hot-potato routing)
4. Any additional BGP criteria

An output port is then found for the prefix, after which the prefix-port entry is added to the forwarding table.

An admin is able to control who routes through their AS by not advertising paths to other ASs. While intra-AS routing protocols tend to focus on performance, policy and business play a larger role in inter-AS routing protocols.

5.7 Broadcast and Multicast Routing

Broadcast routing involves delivering a packet from one source to all other nodes, while multicast routing involves delivering a packet from one source to some subset of nodes.

5.7.1 N-Way-Unicast

N-way-unicast broadcasting involve sending a separate copy of a packet to each node. This approach is simple, but can lead to inefficiencies (e.g. network congestion). Also, obtaining the addresses of all recipients likely involves additional overhead.

5.7.2 Uncontrolled Flooding

Uncontrolled flooding involves a source node sending a copy of a received packet to all neighbors (except the sender). While less likely to immediately cause network congestion, drawbacks include:

- Cycles in networking causing broadcast packets to propagate indefinitely
- Endless multiplication of broadcast packets, leading to a phenomenon known as a broadcast storm or broadcast radiation

5.7.3 Controlled Flooding

Several techniques can be applied to prevent the broadcast storms that result from uncontrolled flooding. One such technique is sequence-number-controlled flooding, where a host puts a unique identifier (i.e. its address) and a broadcast sequence number into the broadcast packets sent to neighbors. Each router stores a list of received packets, and packets are only duplicated and forwarded if they have not yet been received.

Another technique to control flooding is called reverse path forwarding (RPF). In this technique, a router only duplicates and sends a packet if the router which sent the packet is on the shortest path to the unicast source.

Another technique for controlling flooding is to construct a spanning tree from a network. Unlike the previous two techniques, this completely removes transmission of redundant packets, since packets are only forwarded along the tree rather than to all neighbors.

6 Link Layer

The purpose of a link layer is to transfer datagrams between physically adjacent nodes over a link. These nodes can include hosts, routers, switches, WiFi access points, or any other device linked to a network. Links refer to communication channels connecting adjacent nodes.

6.1 Services

Different protocols used in the link layer can provide different services over links. Some common services include:

- **Framing:** Involves encapsulating network layer datagram within a link layer frame.
- **Link access:** A medium access protocol (MAC) specifies how a frame is transferred onto a link. Sometimes, multiple nodes share a single broadcast link, a situation referred to as the multiple access problem.
- **Reliable delivery:** Involves ensuring that a network layer datagram is moved across a link without error. This is rarely implemented on links with low error rates (e.g. fiber optic links), but is more common when error rates are higher, such as on wireless links.
- **Error detection and correction:** Error detection refers to finding bit errors, and correction refers to correcting them without resorting to retransmission.
- **Flow control:** Concerns pacing of datagram transmission across links.

These services are implemented in a host's network adapter, also known as a network interface card (NIC). These implementations exist as a combination of hardware, software, and firmware. Adaptors of adjacent nodes communicate with one another across a communication link. The sender encapsulates a datagram in a frame and sends it across a link. The sender may also provide additional services such as reliable data transfer or flow control. The receiver extracts this datagram, and may also provide additional services such as error detection.

6.2 Error Detection and Correction

Error detection and correction involves protecting data (D) using error detection and correction bits (EDC). The receiver attempts to use the received EDC and data to determine whether there were errors in transmission, and attempts to correct them if possible.

6.2.1 Parity Checks

A very simple form of error detection involves adding a single bit that makes the total number of 1 bits either even or odd. For example, in an even parity schema, if the number of 1s in the data is even, then the parity bit is 0. If the number of 1s in the data is odd, then the parity bit is a 1.

This technique fails at detecting multiple bit errors in a single datagram, which can commonly happen in practice. A more robust approach involves dividing the protected data into rows and columns, each of which has its own parity bits. This two-dimensional approach allows for multiple bit errors to be detected in some cases. This approach also allows for detecting which bit was flipped (i.e. the intersection of an errored row and column).

6.2.2 Cyclic Redundancy Check

Cyclic redundancy checking (CRC) is a widely used technique for error detection. CRC involves a sender and receiver agreeing upon a generator (G) bit pattern of length $r + 1$. When sending data of length d, r bits are appended by the sender such that the result is exactly divisible by G. The receiver considers data to have errors when the received $d + r$ bit pattern is not divisible by G. These r bits can be calculated as:

$$R = remainder(\frac{D \cdot 2^r}{G})$$

6.3 Multiple Access Links and Protocols

Links may either be point-to-point, where senders are only connected to a single receiver, or broadcast, where senders can be connected to multiple receivers. Multiple access protocols are used to regulate transmission into shared broadcast channels to prevent the collision of frames. A few desirable characteristics of multiple access protocols are:

- When only one node transmits, it can use the channel's full capacity.
- When multiple nodes transmit, their average channel usage is the total usage divide by the number of transmitting nodes.
- The protocol is fully decentralized (i.e. no master node that may introduce a single point of failure).
- The protocol is simple.

6.3.1 Channel Partitioning

Channel partitioning protocols involve dividing a channel into pieces that can be allocated to a single node for exclusive use.

One technique for channel partitioning is time division multiple access (TDMA). Exclusive access to a channel is organized in rounds, and each node has a slot in this round where it can send data. This technique is fair and collision free, but inefficient, since time is wasted by allocating slots to nodes with no data to send.

Another technique for channel partitioning is frequency division multiple access (FDMA). A channel is divided into frequency bands, and each frequency band is assigned to a single node. FDMA is fair and collision free, but bandwidth is still wasted by allocating frequency bands to nodes not transmitting data.

6.3.2 Random Access Protocols

In random access protocols, senders attempt to use a channel's full bandwidth when sending data. This introduces the possibility of collisions. When a collision occurs, the sender waits a random window of time before retransmitting. A few common random access protocols are described in this section.

Slotted ALOHA makes the following assumption:

- All transmitted frames are of equal size.
- Time is windowed into fixed intervals that are as long as its takes a transmit a single frame.
- Nodes are synchronized in that they only begin transmission at the beginning of a time window.
- If two or more nodes transmit into a slot, all nodes detect this collision.

When a node transmits a frame without a collision, it can immediately send another one. Otherwise, if a collision is detected, the node attempts to retransmit the frame in the next window with a probability p.

A few advantages of slotted ALOHA are that it is simple, highly decentralized, and allows a single active node to transmit at the full rate of the channel. However, slotted ALOHA is often not efficient when handling multiple sending nodes. Suppose there are N nodes with frames to transmit, and the probability of sending a frame in the next window is p. The probability that any node has a success is $Np(1-p)^{N-1}$. As the number of nodes increases, the optimal efficiency approaches only $\frac{1}{e} \approx 37\%$.

Pure (unslotted) ALOHA is a fully decentralized version of ALOHA (nodes are not synchronized). However, the optimal efficiency is even poorer, approaching $\frac{1}{2e} \approx 18\%$ as the number of sending nodes increases.

Carrier sense multiple access (CSMA) is based on the principle that nodes should defer their transmissions until the channel is not busy (i.e. "listen before talking" analogy). Collisions are still possible, because propagation delay prevents nodes from immediately realizing when another node has begun sending data.

Carrier sense multiple access with collision detection (CSMA/CD) adds a collision detection mechanism to CSMA. Collisions are detected by comparing the sent and received signal strengths, which is much easier to do in wired than wireless connections. If a collision is detected, both nodes stop their transmission, and independently resume it after a random window of time. A common implementation for determining this random window is a binary backoff algorithm, where if a frame has already experienced n collisions, its waiting window is chosen from $\{0, 1, 2, ...2^{n-1}\}$ (i.e. the average waiting time increases exponentially as more collisions occur).

The efficiency of the CSMA/CD algorithm is $\frac{1}{1+5\frac{d_{prop}}{d_{trans}}}$.

6.3.3 Taking-Turns Protocols

Channel partitioning protocols share a channel efficiently when nodes are at high loads, but waste bandwidth when nodes are not sending. Random access protocols are efficient when a single node is sending data, but have collision overhead when multiple nodes are sending. Taking-turns protocols seek to combine the advantages of both types of protocols.

One taking-turns protocol is a polling protocol, where a single node designated as a master polls other nodes in a round robin fashion to allow them to transmit frames. This can eliminates empty slots and collisions, but introduces a polling delay and a single point of failure (i.e. the master node).

Another taking-turns protocol is a token-passing protocol. Instead of there being a master node, special-purpose token frame is passed between nodes. Only the node with the token can send data, which it passes off after it has sent a certain number of frames or has no more frames to send. Token-passing is more decentralized than polling, but still experiences additional latency from moving the token and has a single point of failure (i.e. the token itself).

6.4 Switched Local Area Networks (LANs)

6.4.1 MAC Addresses and ARP

Media access control (MAC) addresses, also known as LAN or physical addresses, serve to indicate to which physically connected interface a frame should be sent. MAC addresses are 6 bytes long. Each LAN network adapter has a unique MAC address, with IEEE managing the MAC address space. MAC addresses are flat (i.e. non-hierarchical) and portable (i.e. do not change depending on position in network).

The address resolution protocol (ARP) involves finding physical address from network addresses. Hosts and routers contain ARP tables that store the mapping between IP and MAC addresses, as well as a time to live field after which the entry is forgotten. If a sender needs to add a missing entry to the ARP table, it broadcasts (i.e. uses MAC address FF-FF-FF-FF-FF-FF) a special packet called an ARP packet. This packet contains the IP address for

which a MAC address should be found. If a receiver matches with this IP address, it replies with its MAC address, and the response is cached in the sender's ARP table. These ARP tables are built automatically (i.e. ARP is plug-and-play).

6.4.2 Ethernet

Ethernet is the dominant LAN technology, being fast and cheap relative to competitors such as FDDI and ATM. Original Ethernet topologies used a coaxial bus to interconnect all nodes, and worked as a broadcast LAN. More recent Ethernet topologies are known as star topologies, and have a switch in the center. The addition of the switch can prevent collision between nodes.

In addition to a network-layer datagram, Ethernet frames consist of the following fields:

- **Preamble (8 bytes):** First 7 bytes store the pattern 10101010 and the last byte stores 10101011. Used to synchronize receiver and sender clock rates by sending a predictable message.
- **Destination address (6 bytes):** If an adapter receives a frame with a matching address (or broadcast address), its data is extracted and forwarded to the network layer. Otherwise, the frame is discarded.
- **Source address (6 bytes)**
- **Type (2 bytes):** Indicates the higher-level protocol (e.g. IP), allowing for multiplexing among different types of protocols.
- **CRC (4 bytes)**

Ethernet offers a connectionless service; network interfaces do not perform a handshake before exchanging data. Ethernet does not ensure reliable data transfer, with network interfaces not exchanging ACKs or NAKs. Ethernet technologies use a CSMA/CD MAC protocol with binary backoff. Modern Ethernet technologies have this MAC protocol and the above frame format in common, but can differ in other ways, such as through different speeds or different physical layer media.

6.4.3 Link Layer Switches

Switches are devices that connect hosts and routers together by receiving incoming link layer frames and selectively forward them to outgoing links based on their MAC address. Switches are transparent to connected hosts and routers in a network. Each connected host and router has a dedicated, full duplex connection to a switch. Each separate link is its own collision domain, with CSMA/CD used as a MAC protocol.

Filtering is a function of a switch that determines whether a frame should be forwarded or dropped. Forwarding involves moving frames to the appropriate outgoing links. Filtering

and forwarding are implemented using switch tables, which contains entries with a MAC address, switch interface leading to that MAC address, and a timestamp for the entry.

Switches support plug-and-play functionality by self-learning the switch table. When a switch receives a message through one of its links, it associates that link with the source MAC address. These entries exist for a period of time called the aging time, after which they are deleted. If entries exist for the destination MAC address, the frame is forwarded through the associated interfaces. Otherwise, the frame is forwarded through all links expect the incoming one (known as flooding).

Routers and switches are similar in that they both forward packets based on some addresses (IP addresses for routers, MAC addresses for switches). Sometimes, either a router or switch can be used to connect multiple devices.

A few advantages of routers are:

- Hierarchical addressing allows for better traffic isolation. This can help prevent cycling, and can help find better paths between nodes because the network topology is not limited to a spanning tree.
- Less susceptible to broadcast storms.

A few advantages of switches are:

- Plug-and-play support.
- Faster packet processing.

6.4.4 Virtual LANs

LANs can often be configured in a hierarchy, with different LANs connecting their switches to a switch at a higher hierarchy. A few drawbacks to this approach include:

- **Lack of traffic isolation:** Broadcast traffic traverses the entire network, which has performance and security implications.
- **Inefficient use of switches:** Extra switches are used to create a hierarchical structure of switches.
- **Issues with user management:** For example, if a user switches LANs, then physical cabling needs to be redone.

Virtual local area networks (VLANs) address this problem by allowing for multiple virtual LANs over a single physical LAN. In port-based VLANs, switch ports are grouped so that a single physical switch can operate as multiple virtual switches. Forwarding between VLANs is done via routing (some devices can act as both a switch and a router).

VLANs can be defined over multiple physical switches, in which case special ports known as trunk ports are used to carry their frames between switches. Special Ethernet frames known as 802.1Q frames carry VLAN ID info.

6.5 Link Virtualization

This section gives a brief overview of multiprotocol label switching (MPLS), which is a technique for improving the performance of IP forwarding by using fixed length identifiers to determine the next node in a path rather than longest prefix matching of IP addresses. An MPLS header is wrapped around an IP header, and contains a label field based on which routing decisions are made (i.e. similar to a virtual-circuit identifier). MPLS-capable routers, which store an MPLS forwarding table, are described as label-switched routers.

MPLS also allows for more flexible traffic routing (e.g. different flows to same destination, pre-computed backup paths) than is possible by only using IP routing protocols. This is because MPLS routing is based on both the source and destination address while IP routing is only based on the destination address.

6.6 Data Center Networks

Large data centers can have hundreds of thousands of connected hosts servicing cloud applications. The hosts in data centers are called blades, and are organized in racks of around 20 to 40 blades. A top of rack (TOR) switch connects the blades within racks. Requests received in a data center are first directed to load balancers, whose job is to balance workload among hosts. To increase reliability, data center networks often include redundant network equipment and links.