**Chapter 1: Introduction and Basic Concepts**

• Packet switching vs. circuit switching. X

• "Hosts" vs. "Switches/Routers" x

• The definition and purpose of a protocol. X

• Standards, and the historic role of RFCs in defining them. X

• “Network edge” vs. “Network core"

• LAN vs. WAN

* Local area network (LAN)
* Wide Area Network (WAN)
* Very different Protocols, factor in distance and needs

• Wi-Fi standard

* Wireless Fidelity

• Cellular communication

* Telephones talk to cell towers

• Guided vs. unguided media.

• Communication media speed and performance differences.

* Ethernet
* Fiber optics
* Different technologies have different media speeds

• “Store-and-forward” communication.

• Communication delays.

• The basic history and original intent of the Internet's design (why packet switching and not circuits like old phones?)

• “Routing” vs. “forwarding"/"switching”.

* Forwarding: sends data to the next link in the change

• Throughput

* Amount of data being passed through a system of process

• End-to-end communication delay.

* Source to Destination -> back to Source

• Protocol layers, and the concept of a protocol stack.

* Application , Transport, Network, Link, Physical
* Protocol Stack to have different layers to handle different functions used in the internet and overall transfer of information

• The OSI and TCP/IP protocol layers.

OSI

* Application
* Presentation
* Session
* Transport
* Network
* Data
* Physical

TCP/IP protocol Layer

* Application
* Transport
* Network
* Link
* Physical

• Protocol types: open vs. proprietary, and ad-hoc vs. de jure.

• An Application Programming Interface (API).

**Chapter 2: Application Laye**r

• “Client-server” vs “peer-to-peer” application architectures.

* Client-Server : Client receives all info, server handles all info (Centralized)
* P2P: Peers share all info, can act as both client and server, (decentralized

• Sockets and system calls.

• Inter-process communication.

• Transport services.

• TCP vs UDP as examples of transport layers (beneath the application layer).

• Basic web protocol concepts: HTTP, HTML, URL

* HTTP : deals with sending Webpages
  + Data is not encrypted
* HTML: Creating and structuring WebPages
* URL: Provides location of webpages

• Do web application-layers protocols use TCP or UDP?

• “stateless protocol” (like HTTP, and why HTTP is stateless)

• Persistent vs non-persistent HTTP?

• RTT

• HTTP security (lack thereof)

• “cookies” (and how they give servers a means to track clients'/state)

• FTP, SMTP, IMAP, and POP (basic awareness of their purpose and general nature)

• DNS (basic awareness of structure, and need for scalability)

• Basic DNS record types (“A”, “MX”, and “NS” resource records).

• Iterative vs. recursive lookup in DNS

**Chapter 3: Transport Layer**

• Communication between processes (transport layer) vs. between hosts (Network layer)

• TCP vs. UDP

• “Multiplexing” and “demultiplexing” (specifically in the transport layer).

* Multiplexing : Sending multiple streams from different applications towards a single socket
  + Allows multiple applications to send data and use the same network connection
* Demultiplexing, separating the data to there respective functions, based on port numbers and application

• Demultiplexing using more information under TCP than UDP.

* Based on the port numbers that are reliably given. Demultiplexing will separate the info based on port numbers to the respective application.

• The simple UDP checksum (its purpose).

• State machines and their use to describe a protocol.

* State machines describe the behavior of protocols based on its state and it’s transition

• Stop-and-wait vs. Pipelining

• Go-back-N (GBN) error recovery vs. Selective Repeat (SR).

GBN is easier use, typically used in streaming services

SR is used more when there needs to be minimal error

• The minimum number of sequence numbers required for GBN vs. SR.

• “Flow control” vs. “congestion control” (and how TCP ensures both)

• TCP’s "Slow Start"/AIMD and its relationship to "congestion control"

It has a slow start tat grows exponentially when sending data when it feels its close it will additionally increase (AI in AIMD) and once congestion is detected it will Multiplicative decrease to half of its original flow and do small increments (additionally increase) to a good data flow/,