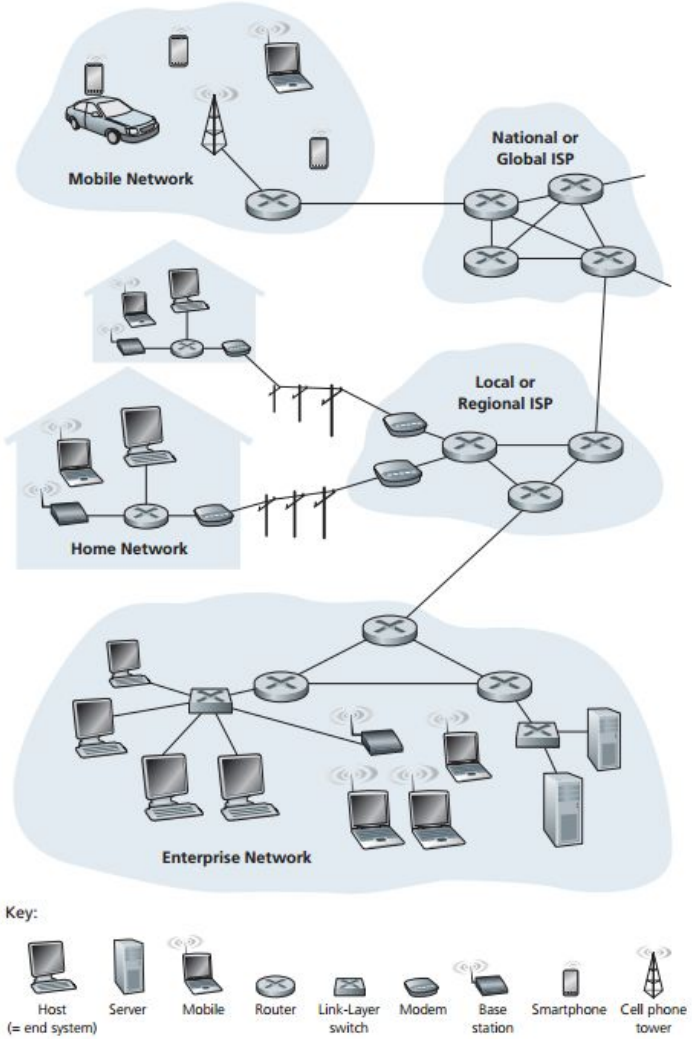
# Computer Networks
## An Introduction

Ameya Daigavane
Deepak Kumar Gouda

# What does the Internet look like?



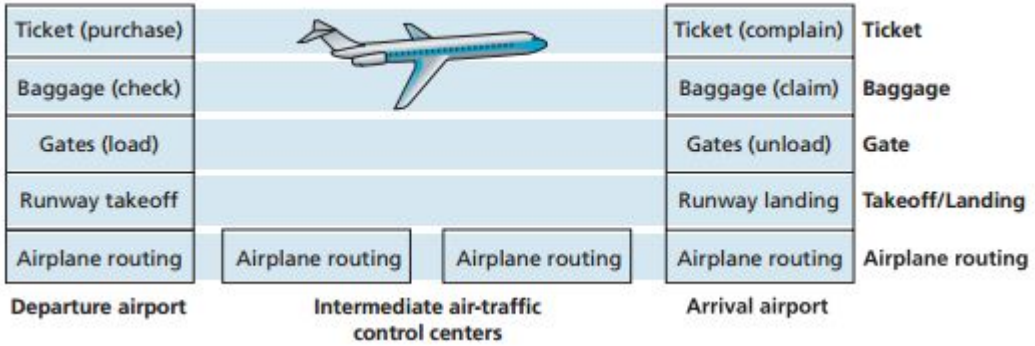
# Network Protocols

- Specify 3 aspects:
  - What kinds of messages are to be sent,
  - The format of these messages
  - The actions taken when messages are received/sent.

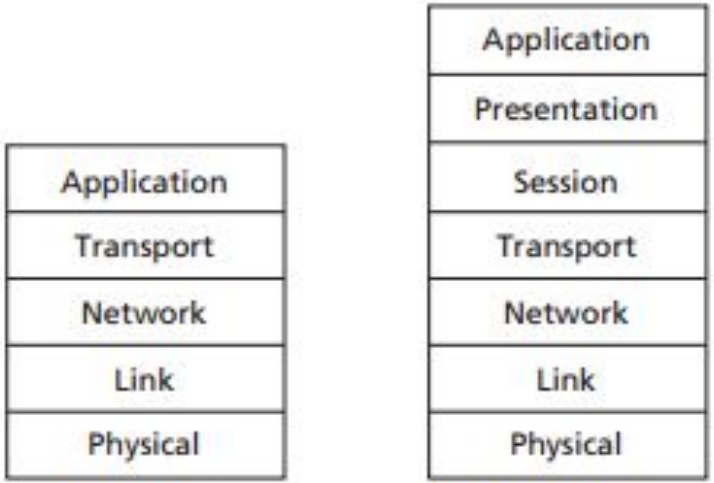
The Internet is completely based on protocols! Examples: HTTP, TCP, IP!

# A Functional Description

Based on the functions/services these protocols provide!



# The Protocol Stack




a. Five-layer Internet protocol stack

b. Seven-layer ISO OSI reference model

# The Application Layer

# Sockets

Provide an interface to applications to read/write data.

# HyperText Transfer Protocol (HTTP)

- Web's application-layer protocol (heart of the Web)
- It uses TCP as its underlying transport protocol
- Requests: GET, POST
- http:// vs https://, HTTP vs HTML

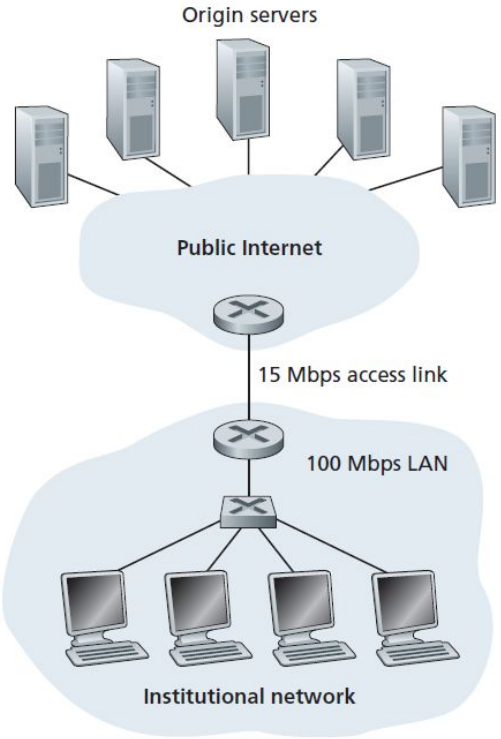
```
GET /somedir/page.html HTTP/1.1
Host: www.someschool.edu
Connection: close
User-agent: Mozilla/5.0
Accept-language: fr
```

```
HTTP/1.1 200 OK
Connection: close
Date: Tue, 09 Aug 2011 15:44:04 GMT
Server: Apache/2.2.3 (CentOS)
Last-Modified: Tue, 09 Aug 2011 15:11:03 GMT
Content-Length: 6821
Content-Type: text/html

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

# Web Caches (Server-Side Proxies)

- Faster responses due to caching
- Saves internet bandwidth
- Traffic monitoring



## Other Protocols

- FTP (file transfer)
- SSH (remote logins)
- SMTP (emails)


Control and data connections

## Domain Name Service

- Maps domain names (like *www.google.com* ) to IP addresses (like *173.121.64.3*).
- Can be used to implement simple load distribution.
- Works in a distributed manner over multiple servers

What happens when you type a URL in the web browser?

# The Transport Layer

# The Transport Layer

- Takes **messages** from applications and converts them to **segments**, by adding headers (control information for the transport layer).
- The network layer connects two hosts. The transport layer connects two processes running on these hosts.
- Also, provides **error-checking** (via checksums)!

Two important protocols - **UDP (User Datagram Protocol)** and **TCP (Transmission Control Protocol)**.

# Which Protocol Do I Choose?

| 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 |
| Routing protocol | RIP | Typically UDP |
| Name translation | DNS | Typically UDP |

# TCP Functions

UDP only performs process-to-process multiplexing/demultiplexing, and error-checking.

TCP does a lot more more:

- Reliable Data Transfer
- Congestion Control

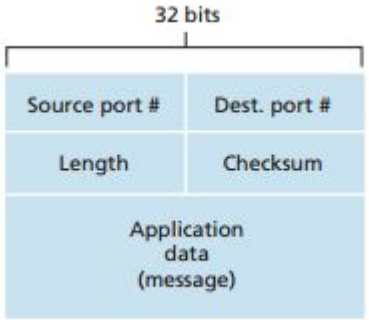
# Demultiplexing/Multiplexing for Processes

Many processes (application-layer) use the same transport layer protocol. How do transport-layer protocols distinguish between messages for different processes?

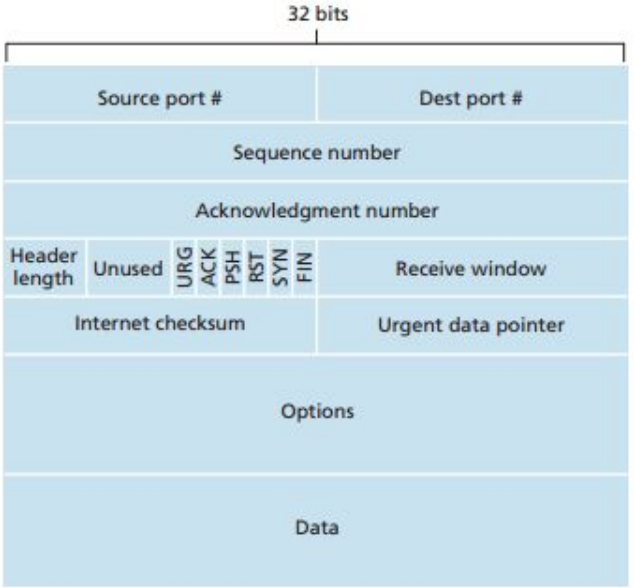
**Solution:** Identify sockets by port number, and list the ports in the headers.

- UDP sockets: 2-tuple (destination IP, destination port number)
- TCP sockets: 4-tuple (destination IP, destination port number, source IP, source port number)

# Segment Structures



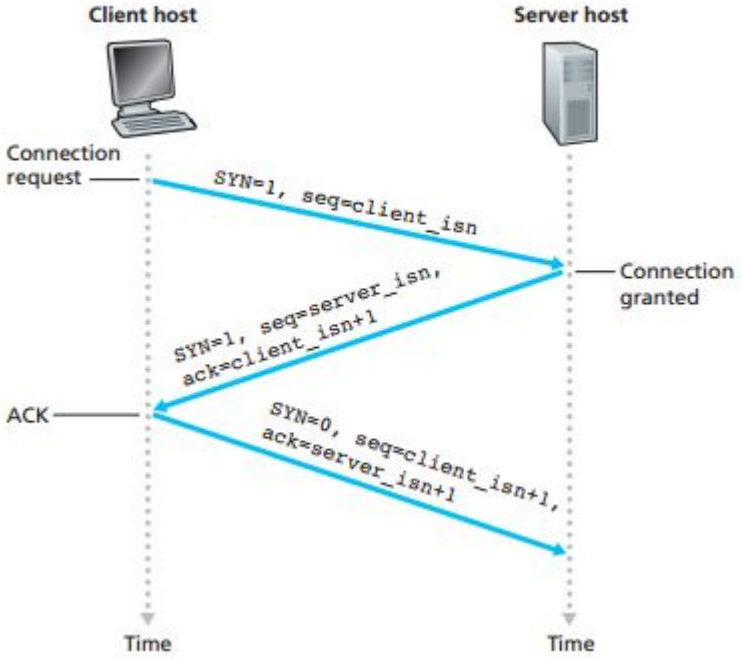

UDP
8 bytes




TCP
20 bytes

# TCP Connection Establishment
## (The 3-way Handshake)

Note how the sequence numbers change!



# Reliable Data Delivery

- Ensure that the receiver has actually received the data.
- Multiple aspects:
  - Acknowledgements
  - Sequence Numbers
  - Timers
  - Flow Control

# Flow Control

- Any sender should not overflow the receiver's buffer!
- The receiver indicates how much space it has in the 'receive window' field in the TCP header.

$$\texttt{LastByteSent} - \texttt{LastByteAcked} \leq \texttt{rwnd}$$

# Congestion Control

- Senders should also not overflow the queues/buffers over links.
- Sender maintains a variable ‘congestion window’.

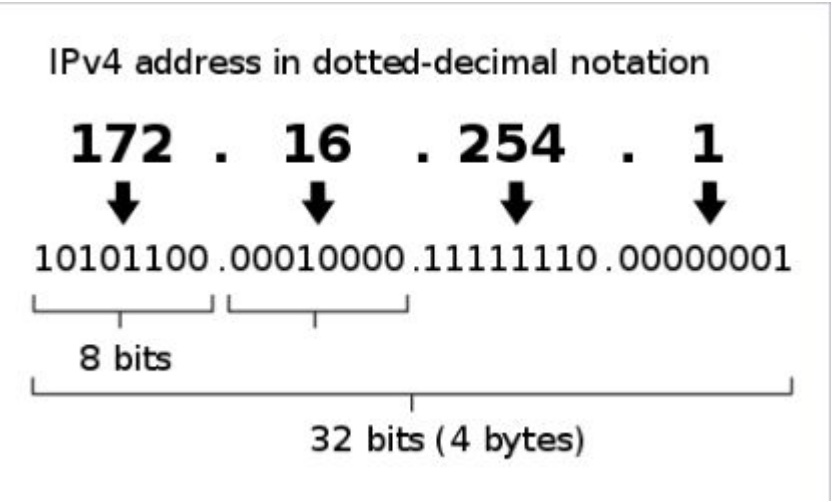
$$\text{LastByteSent} - \text{LastByteAcked} \leq \min\{\text{cwnd}, \text{rwnd}\}$$

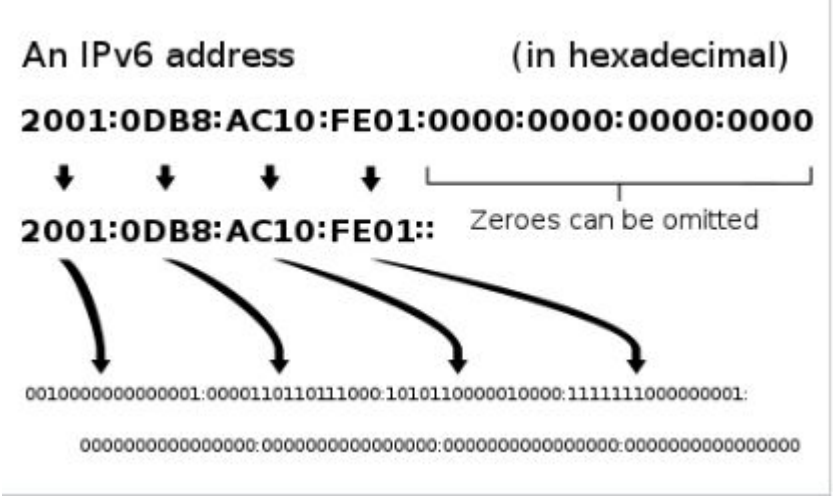
# The Network Layer

# Internet Service Model

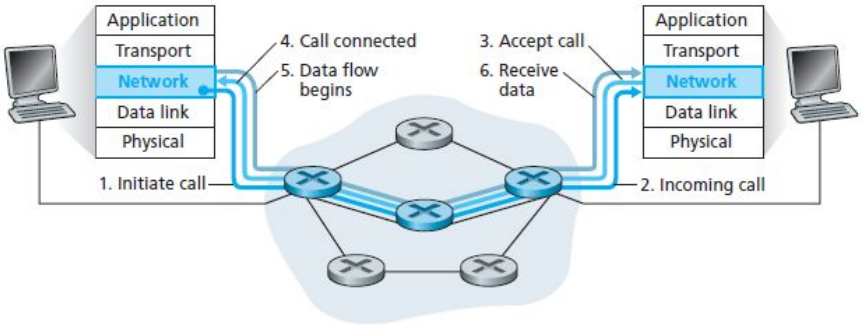
| Network Architecture | Service Model | Bandwidth Guarantee | No-Loss Guarantee | Ordering | Timing | Congestion Indication |
|---|---|---|---|---|---|---|
| Internet | Best Effort | None | None | Any order possible | Not maintained | None |
| ATM | CBR | Guaranteed constant rate | Yes | In order | Maintained | Congestion will not occur |
| ATM | ABR | Guaranteed minimum | None | In order | Not maintained | Congestion indication provided |

# IP Addresses

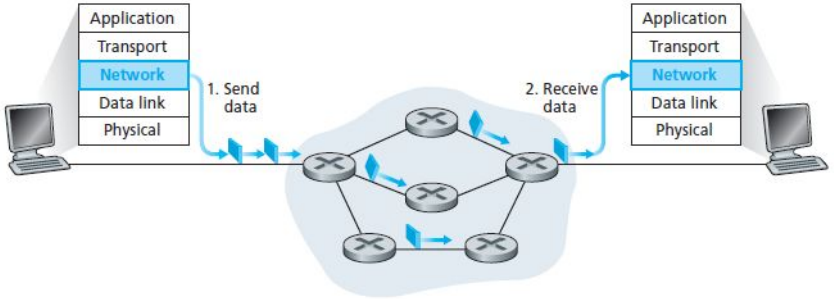


# Circuit Switching vs Packet Switching




Circuit Switching




Packet Switching

# Routing, Subnet

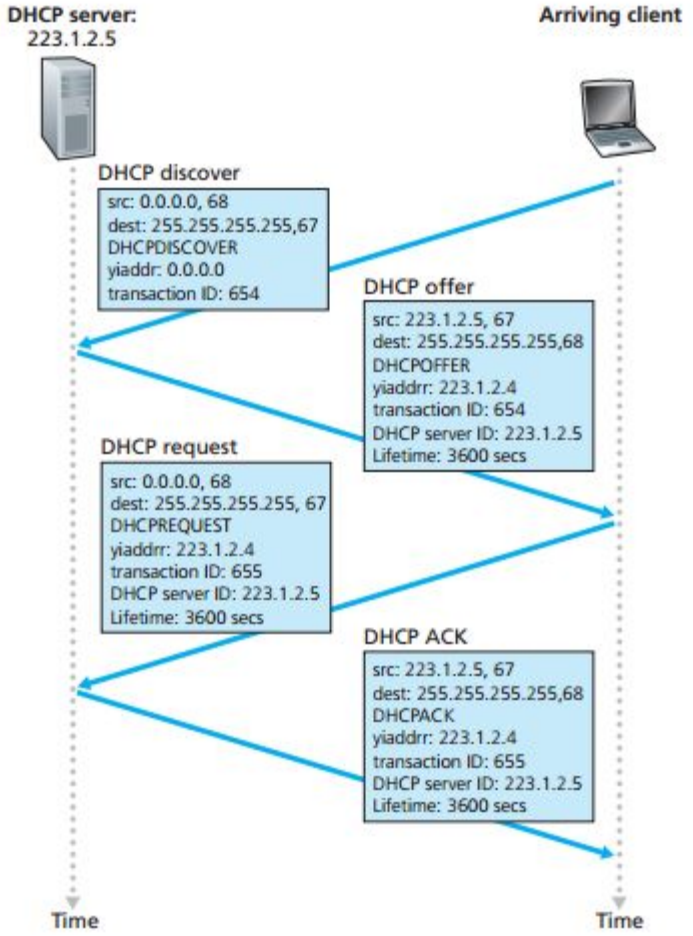
- Forwarding tables
- Public IP vs Private IP

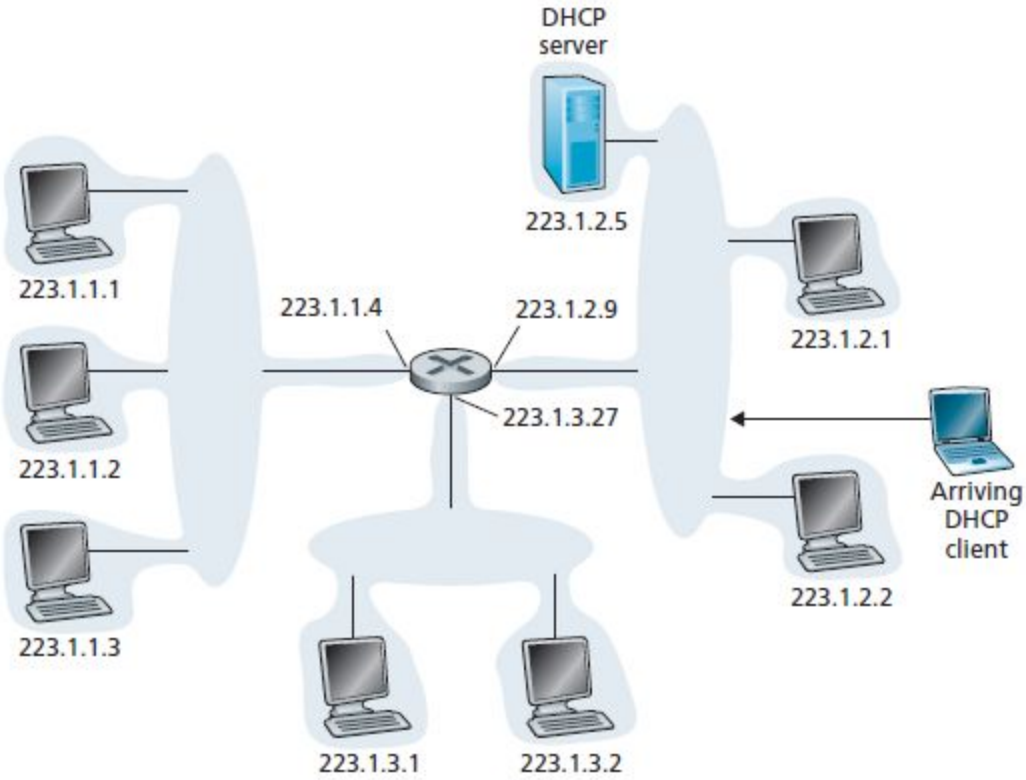
| Prefix Match | Link Interface |
|---|---|
| 11001000 00010111 00010 | 0 |
| 11001000 00010111 00011000 | 1 |
| 11001000 00010111 00011 | 2 |
| otherwise | 3 |

# DHCP

## Dynamic Host Configuration Protocol

What if you don’t have an IP?




DHCP
server
223.1.2.5
223.1.1.1
223.1.1.4
223.1.2.9
223.1.2.1
223.1.1.2
223.1.3.27
Arriving
DHCP
client
223.1.2.2
223.1.1.3
223.1.3.1
223.1.3.2

# Thank You!


## Acknowledgements

Images taken from *Computer Networking: A Top Down Approach* (by Kurose and Ross) 6th edition.