

Appendix

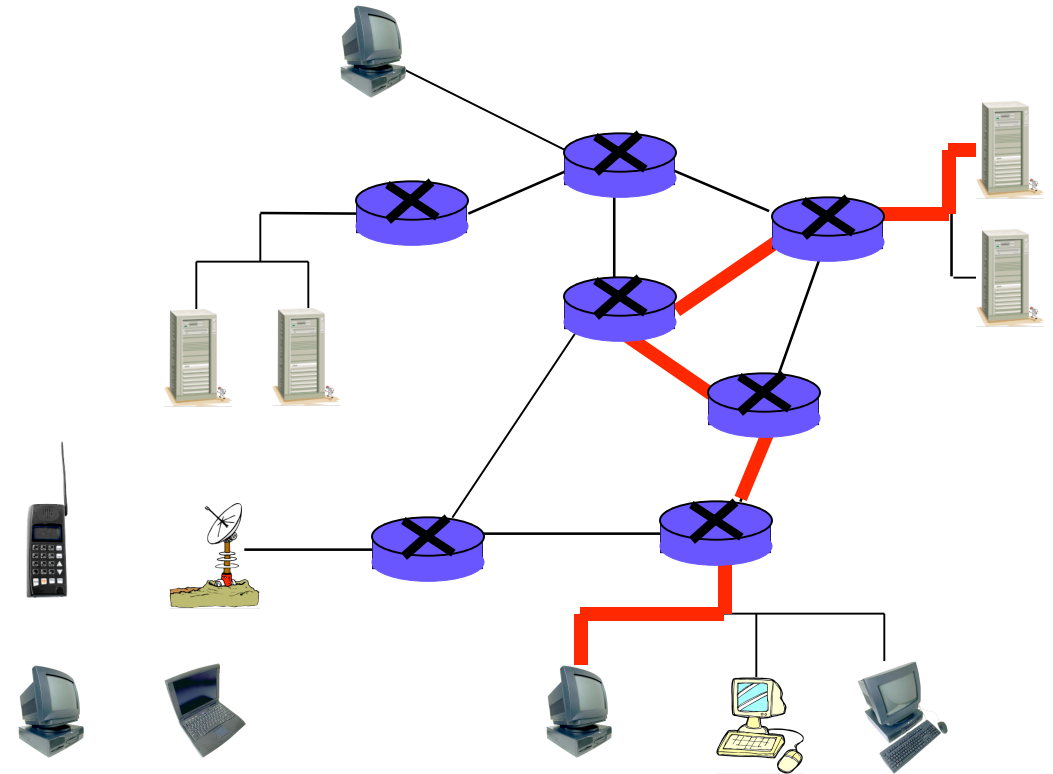
Appendix

- ❑ Networking basics
 - Protocol stack
 - Layers, etc.
- ❑ Math basics
 - Modular arithmetic
 - Permutations
 - Probability
 - Linear algebra

Networking Basics

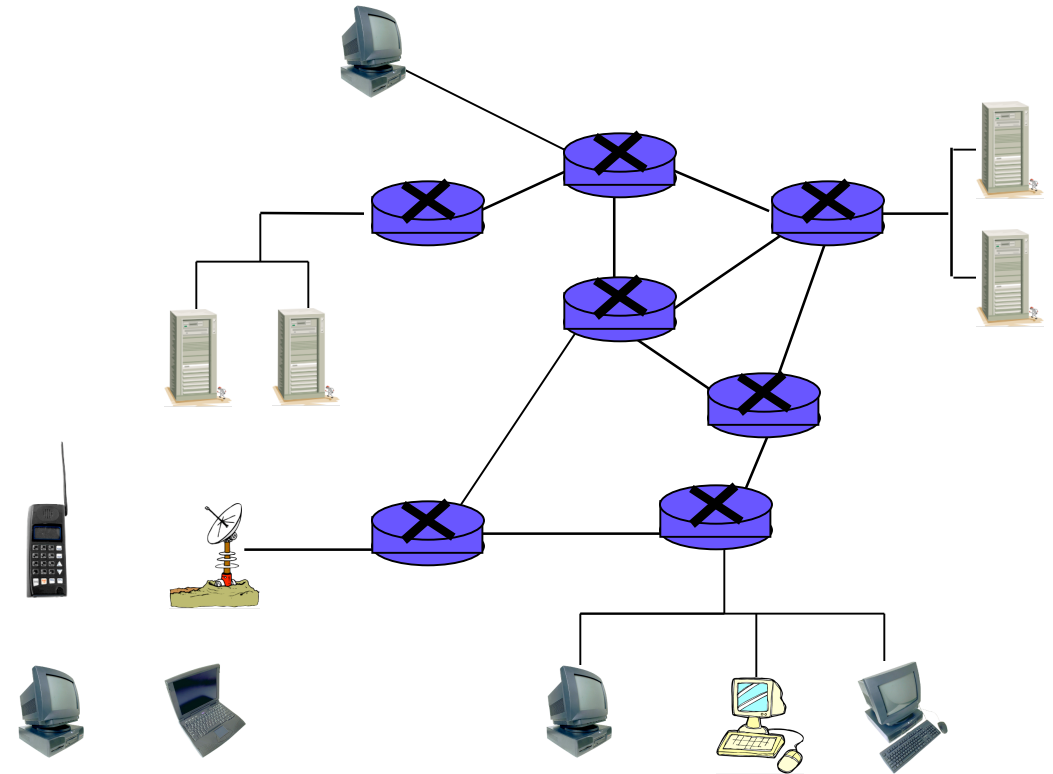
Network

- Includes
 - Computers
 - Servers
 - Routers
 - Wireless devices
 - Etc.
- Purpose is to transmit data



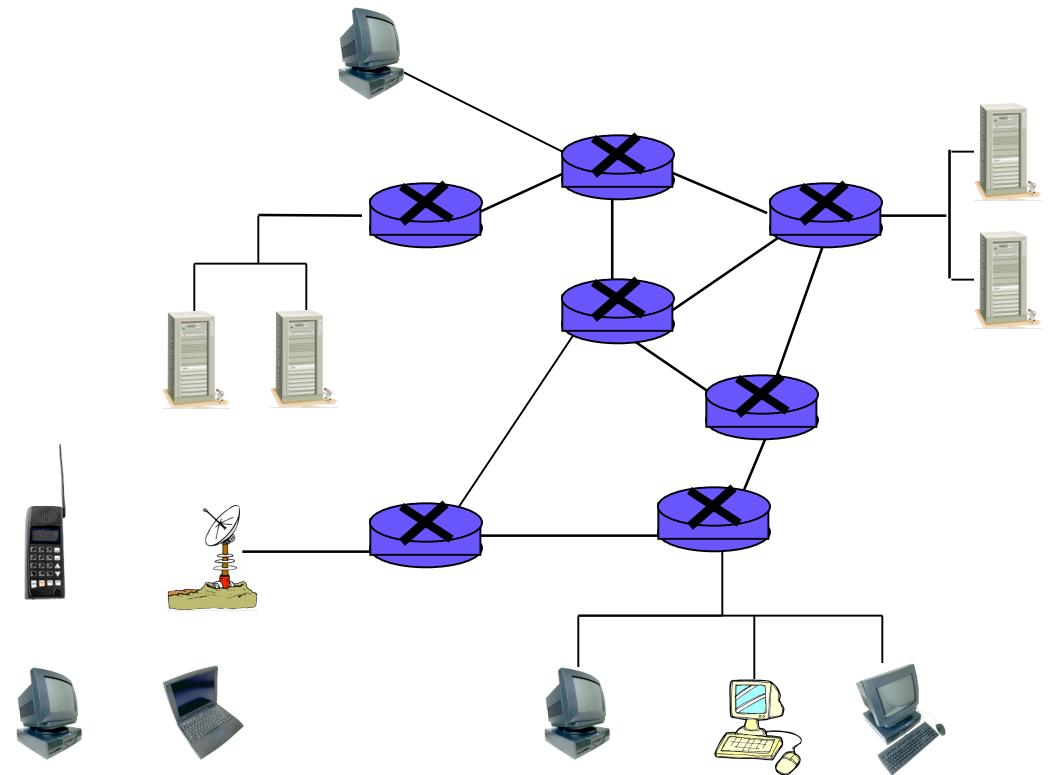
Network Edge

- Network edge includes
- Hosts
 - Computers
 - Laptops
 - Servers
 - Cell phones
 - Etc., etc.



Network Core

- Network core consists of
 - Interconnected mesh of routers
- Purpose is to move data from host to host



Packet Switched Network

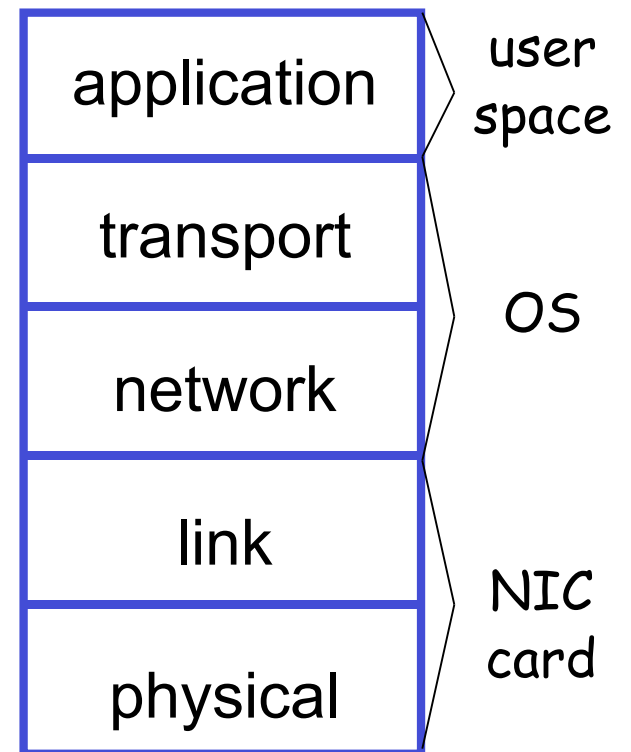
- ❑ Usual telephone network is circuit switched
 - For each call, a dedicated circuit is established
 - Dedicated bandwidth
- ❑ Modern data networks are **packet switched**
 - Data is chopped up into discrete packets
 - Packets are transmitted independently
 - No real circuit is established
 - More efficient bandwidth usage
 - But more complex than circuit switched

Network Protocols

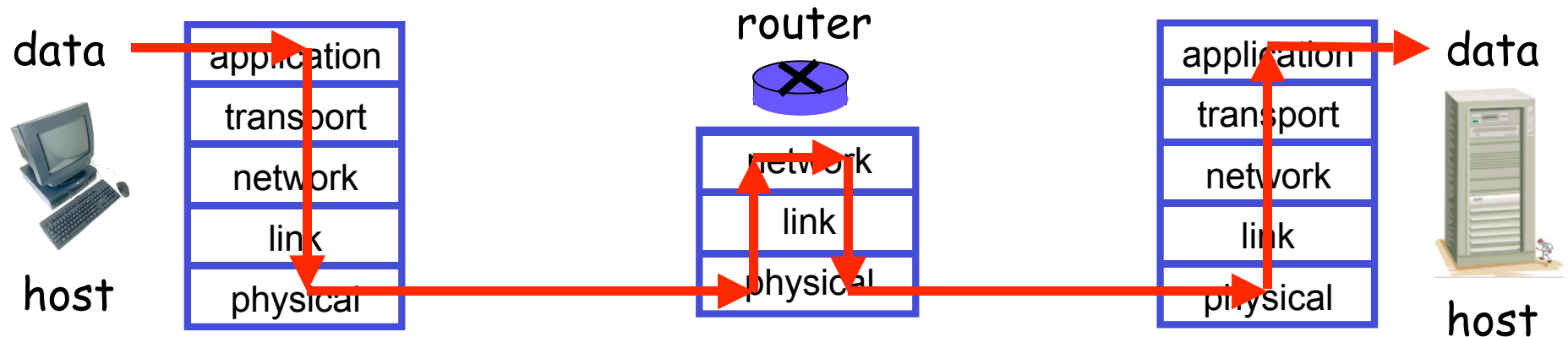
- ❑ Study of networking focused on protocols
- ❑ Networking protocols precisely specify the communication rules
- ❑ Details are given in **RFCs**
 - RFC is essentially an Internet standard
- ❑ **Stateless** protocols don't remember
- ❑ **Stateful** protocols do remember
- ❑ Many security problems related to state
- ❑ DoS easier against stateful protocols

Protocol Stack

- ❑ Application layer protocols
 - HTTP, FTP, SMTP, etc.
- ❑ Transport layer protocols
 - TCP, UDP
- ❑ Network layer protocols
 - IP, routing protocols
- ❑ Link layer protocols
 - Ethernet, PPP
- ❑ Physical layer



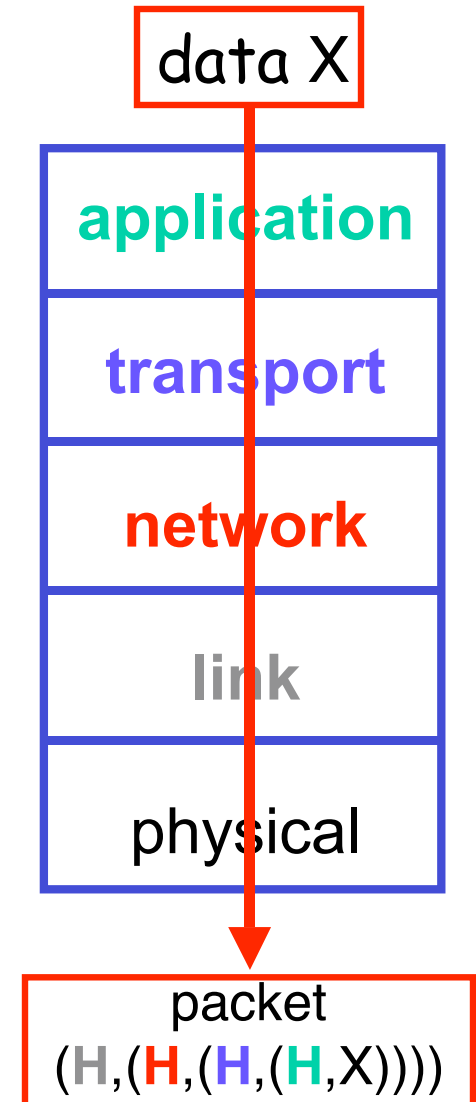
Layering in Action



- ❑ At source, data goes down the protocol stack
- ❑ Each router processes packet up to network layer
 - That's where routing info lives
- ❑ Router then passes packet down the protocol stack
- ❑ Destination processes up to application layer
 - That's where the data lives

Encapsulation

- ❑ X = application data at the source
- ❑ As X goes down protocol stack, each layer adds header information:
 - Application layer: (H, X)
 - Transport layer: $(H, (H, X))$
 - Network layer: $(H, (H, (H, X)))$
 - Link layer: $(H, (H, (H, (H, X))))$
- ❑ Header has info required by layer
- ❑ Note that app header is on the inside



Application Layer

- ❑ Applications
 - Web browsing, email, P2P, etc.
 - Running on hosts
 - Hosts want network to be transparent
- ❑ Application layer protocols
 - HTTP, SMTP, IMAP, Gnutella, etc.
- ❑ Protocol is one part of an application
 - For example, HTTP only a part of web browsing

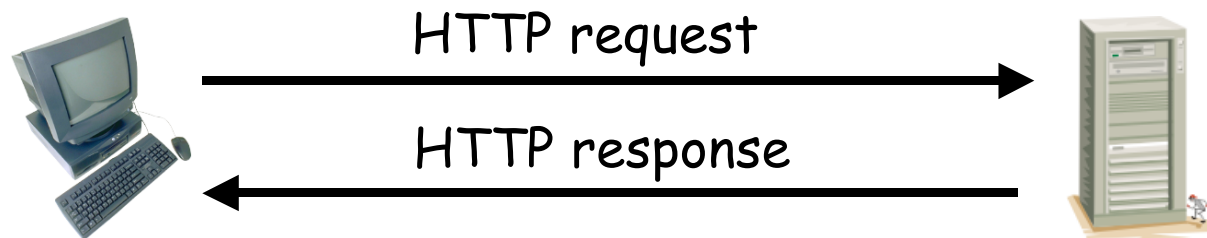
Client-Server Model

- **Client** "speaks first"
- **Server** tries to respond to request
- Hosts are clients and/or servers
- Example: Web browsing
 - You are the client (request web page)
 - Web server is the server

Peer-to-Peer Model

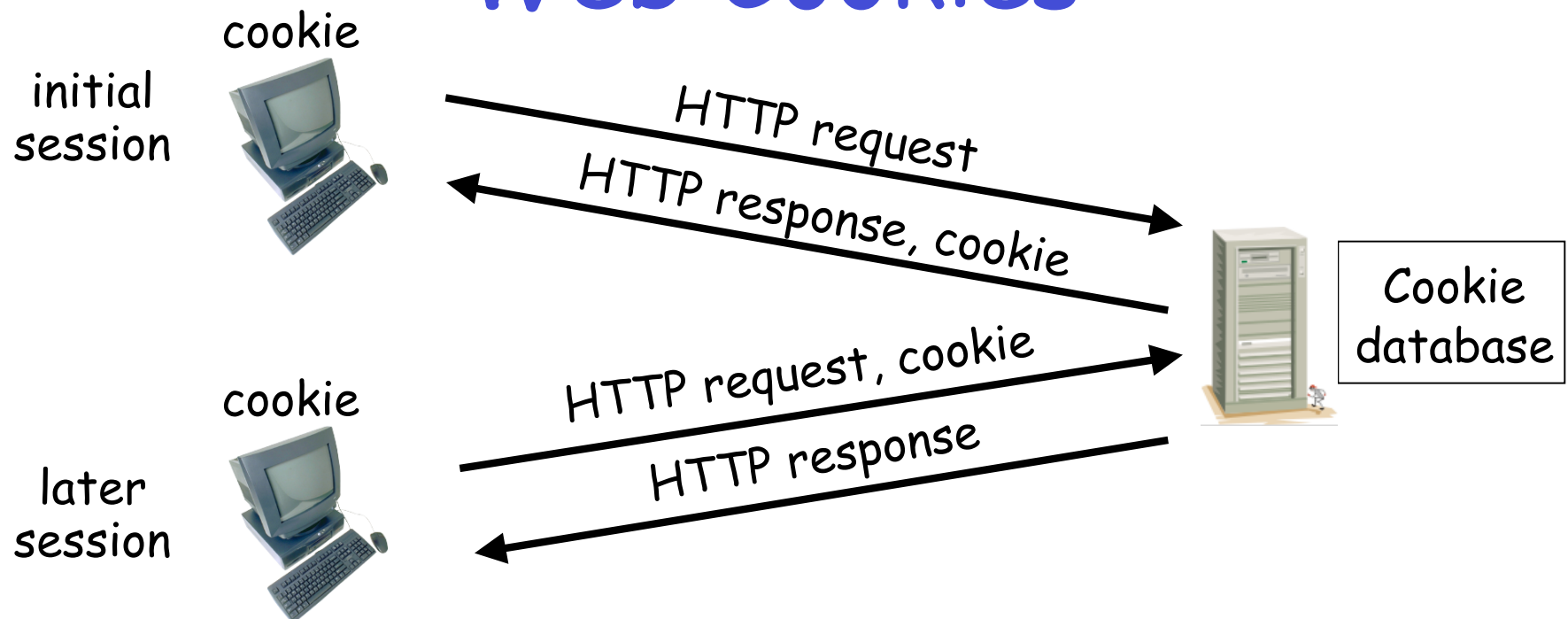
- ❑ Hosts act as clients and servers
- ❑ For example, when sharing music
 - You are client when requesting a file
 - You are a server when someone downloads a file from you
- ❑ In P2P model, more difficult for client to find a server
- ❑ Many different P2P models

HTTP Example



- HTTP --- HyperText Transfer Protocol
- Client (you) request a web page
- Server responds to your request

Web Cookies



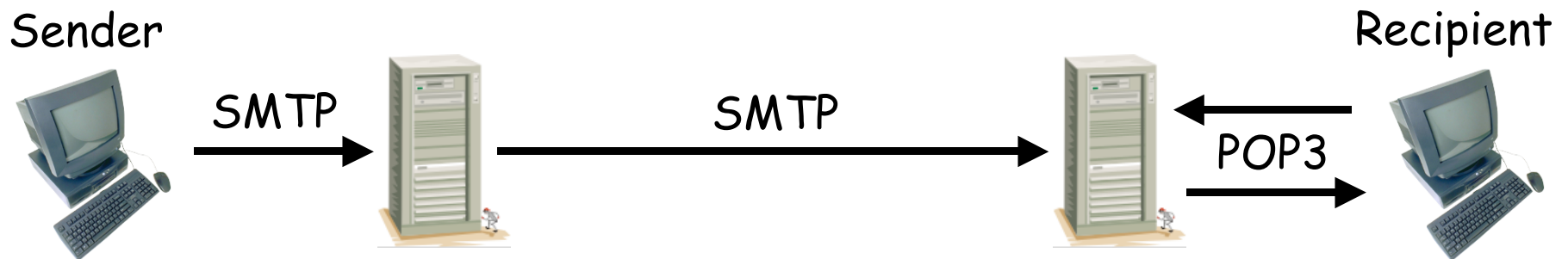
- ❑ HTTP is stateless --- cookies used to add state
- ❑ Initially, cookie sent from server to browser
- ❑ Browser manages cookie, sends it to server
- ❑ Server looks in cookie database to "remember" you

Web Cookies

- Web cookies can be used for
 - Shopping carts
 - Recommendations, etc.
 - A weak form of authentication
- Privacy concerns
 - Web site can learn a lot about you
 - Multiple web sites could learn even more

SMTP

- ❑ SMTP used to send email from sender to recipient's mail server
- ❑ Then use POP3, IMAP or HTTP (Web mail) to get messages from server
- ❑ As with many application protocols, SMTP commands are human readable



Spooferd email with SMTP

User types the red lines:

```
> telnet eniac.cs.sjsu.edu 25
220 eniac.sjsu.edu
HELO ca.gov
250 Hello ca.gov, pleased to meet you
MAIL FROM: <arnold@ca.gov>
250 arnold@ca.gov... Sender ok
RCPT TO: <stamp@cs.sjsu.edu>
250 stamp@cs.sjsu.edu ... Recipient ok
DATA
354 Enter mail, end with "." on a line by itself
It is my pleasure to inform you that you
are terminated
.
250 Message accepted for delivery
QUIT
221 eniac.sjsu.edu closing connection
```

Application Layer

- ❑ DNS --- Domain Name Service
 - Convert human-friendly names such as www.google.com into 32-bit IP address
 - A distributed hierarchical database
- ❑ Only 13 "root" DNS servers worldwide
 - A single point of failure for Internet
 - Attacks on root servers have succeeded
 - Attacks have not lasted long enough (yet...)

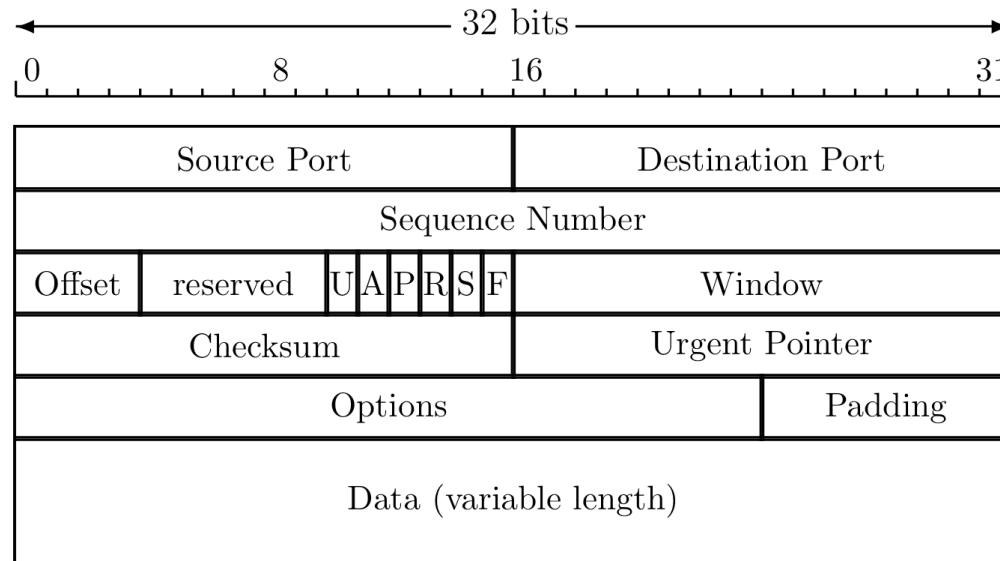
Transport Layer

- ❑ The network layer offers unreliable, “best effort” delivery of packets
- ❑ Any improved service must be provided by the hosts
- ❑ Transport layer has two protocols
 - TCP — better service, more overhead
 - UDP — minimal service, minimal overhead
- ❑ TCP and UDP run on hosts, not routers

TCP

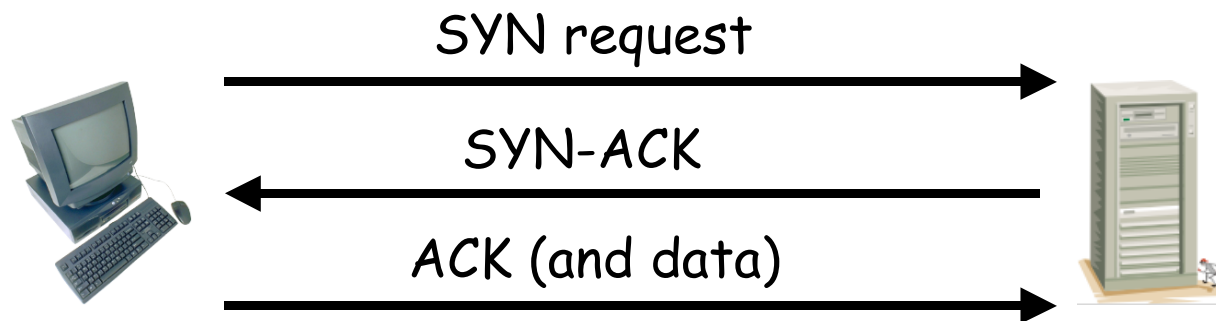
- ❑ TCP assures that packets
 - Arrive at destination
 - Are processed in order
 - Are not sent too fast for receiver (flow control)
- ❑ TCP also provides
 - Network-wide congestion control
- ❑ TCP is "connection-oriented"
 - TCP contacts server before sending data
 - Orderly setup and take down of "connection"
 - But no true connection, only a logical connection

TCP Header



- ❑ Source and destination port
- ❑ Sequence number
- ❑ Flags (ACK, SYN, RST, etc.)
- ❑ 20 bytes (if no options)

TCP Three Way Handshake



- ❑ SYN: synchronization requested
- ❑ SYN-ACK: acknowledge SYN request
- ❑ ACK: acknowledge msg 2 and send data
- ❑ Then TCP "connection" established
 - Connection terminated by FIN or RST packet

Denial of Service Attack

- ❑ The TCP 3-way handshake makes denial of service (DoS) attacks possible
- ❑ Whenever SYN packet is received, server must remember "half-open" connection
 - Remembering consumes resources
 - Too many half-open connections and server resources will be exhausted
 - Then server can't respond to new connections

UDP

- ❑ UDP is minimalist, “no frills” service
 - No assurance that packets arrive
 - No assurance packets are in order, etc., etc.
- ❑ Why does UDP exist?
 - More efficient (smaller header)
 - No flow control to slow down sender
 - No congestion control to slow down sender
- ❑ Packets sent too fast, they will be dropped
 - Either at intermediate router or at destination
 - But in some apps this is OK (audio/video)

Network Layer

- ❑ Core of network/Internet
 - Interconnected mesh of routers
- ❑ Purpose of network layer
 - Route packets through this mesh
- ❑ Network layer protocol is IP
 - Follows a “best effort” approach
- ❑ IP runs in every host and every router
- ❑ Routers also run routing protocols
 - Used to determine the path to send packets
 - Routing protocols: RIP, OSPF, BGP, etc.

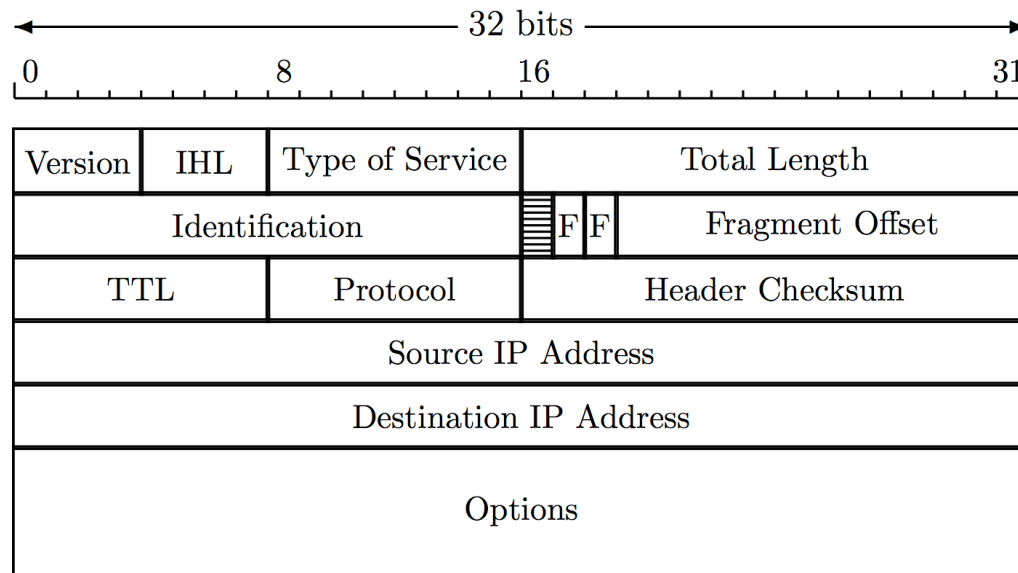
IP Addresses

- ❑ IP address is 32 bits
- ❑ Every host has an IP address
- ❑ Not enough IP addresses!
 - Lots of tricks to extend address space
- ❑ IP addresses given in dotted decimal notation
 - For example: 195.72.180.27
 - Each number is between 0 and 255
- ❑ Host's IP address can change

Socket

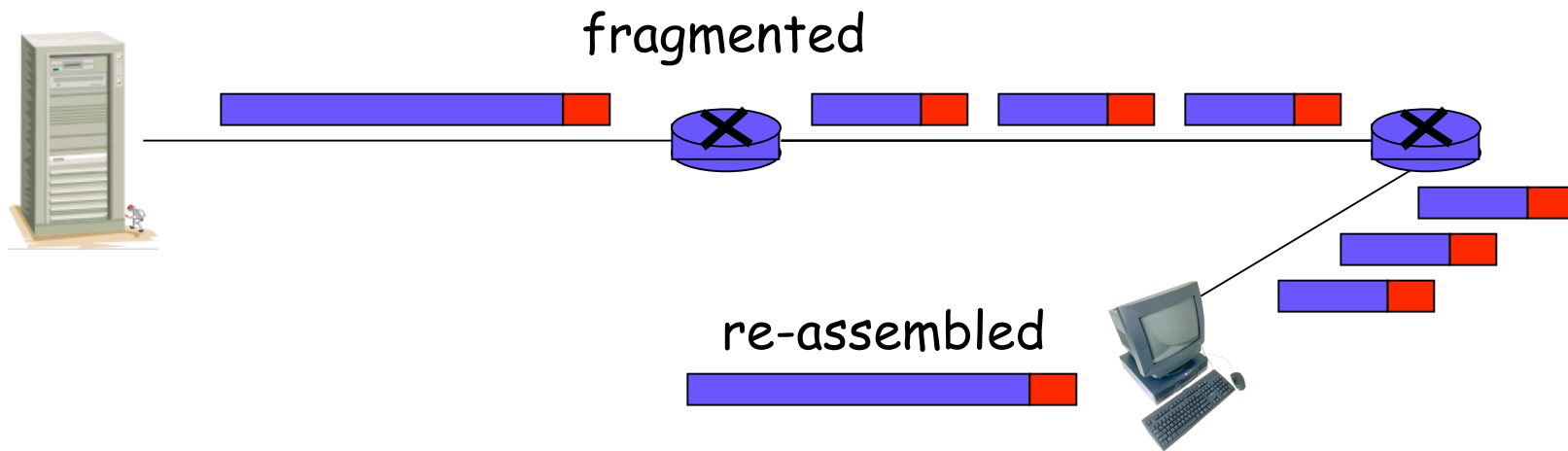
- ❑ Each host has a 32 bit IP address
- ❑ But many processes on one host
 - You can browse web, send email at same time
- ❑ How to distinguish processes on a host?
- ❑ Each process has a 16 bit **port number**
 - Port numbers < 1024 are "well-known" ports (HTTP port 80, POP3 port 110, etc.)
 - Port numbers above 1024 are dynamic (as needed)
- ❑ IP address and port number define a **socket**
 - Socket uniquely identifies a process

IP Header



- ❑ IP header used by routers
 - Note source and destination IP addresses
- ❑ Time to live (TTL) limits number of "hops"
 - So packets can't circulate forever
- ❑ Fragmentation information (see next slide)

IP Fragmentation



- ❑ Each link limits maximum size of packets
- ❑ If packet is too big, router fragments it
- ❑ Re-assembly occurs at destination

IP Fragmentation

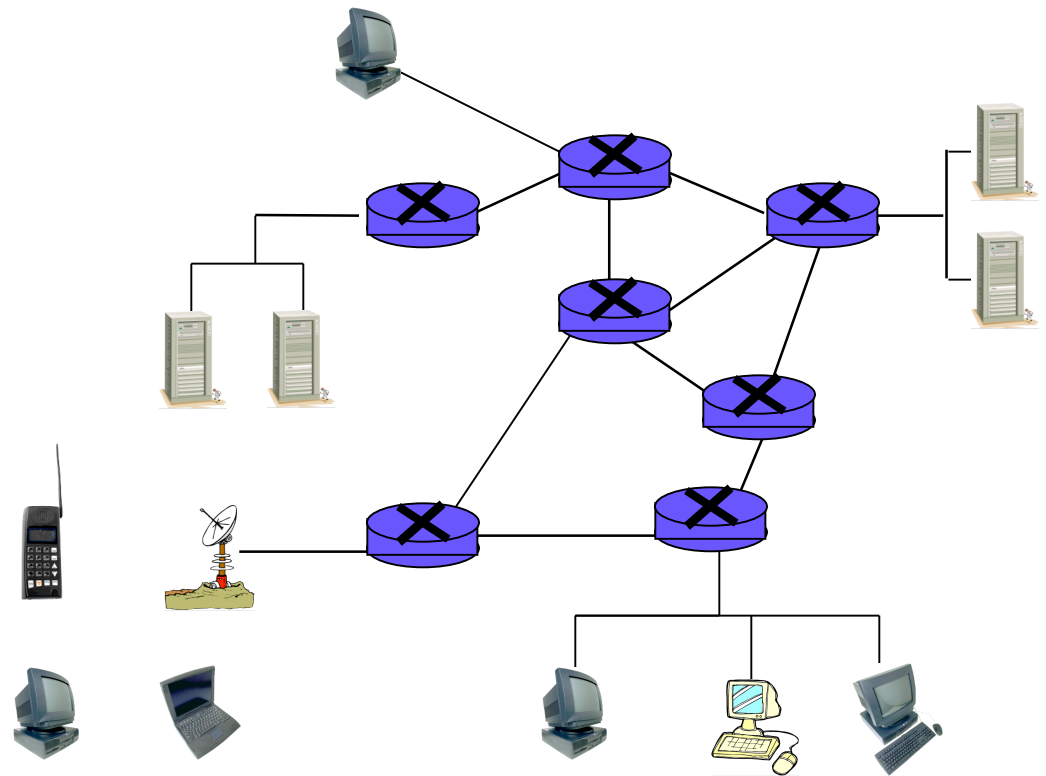
- ❑ One packet becomes multiple packets
- ❑ Packets reassembled at destination
 - Prevents multiple fragment/re-assemble
- ❑ Fragmentation is a security issue!
 - Fragments may obscure real purpose of packet
 - Fragments can overlap when re-assembled
 - Must re-assemble packet to fully understand it
 - Lots of work for firewalls, for example

IPv6

- ❑ Current version of IP is IPv4
- ❑ IPv6 is a new-and-improved version
- ❑ IPv6 provides
 - Longer addresses: 128 bits
 - Real security (IPSec)
- ❑ But difficult to migrate from v4 to v6
- ❑ So IPv6 has not taken hold yet

Link Layer

- Link layer sends packet from one node to next
- Each link can be different
 - Wired
 - Wireless
 - Ethernet
 - Point-to-point...



Link Layer

- ❑ Implemented in adapter known as network interface card, or NIC
 - Ethernet card
 - Wireless 802.11 card, etc.
- ❑ NIC is (mostly) out of host's control
 - Implements both link and physical layers

Ethernet

- ❑ Ethernet is a multiple access protocol
- ❑ Many hosts access a shared media
 - On a local area network, or LAN
- ❑ In ethernet, two packets can collide
 - Then data is corrupted
 - Packets must be resent
 - How to be efficient in distributed environment?
 - Many possibilities, ethernet is most popular
- ❑ We won't discuss details here

Link Layer Addressing

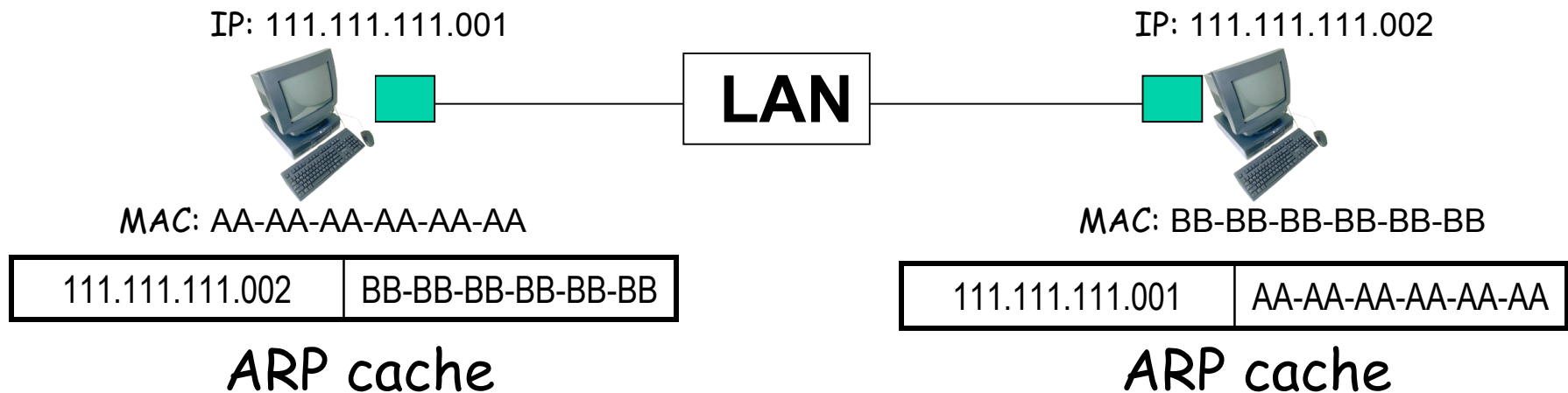
- ❑ IP addresses live at network layer
- ❑ Link layer also requires addresses
 - MAC address (LAN address, physical address)
- ❑ MAC address
 - 48 bits, globally unique
 - Used to forward packets over one link
- ❑ Analogy
 - IP address is like home address
 - MAC address is like social security number

ARP

- ❑ Address resolution protocol, ARP
- ❑ Used at link layer to find MAC address of given IP address
- ❑ Each host has ARP table
 - Generated automatically
 - Entries expire after some time (20 min)
 - ARP used to find ARP table entries
 - ARP table also known as **ARP cache**

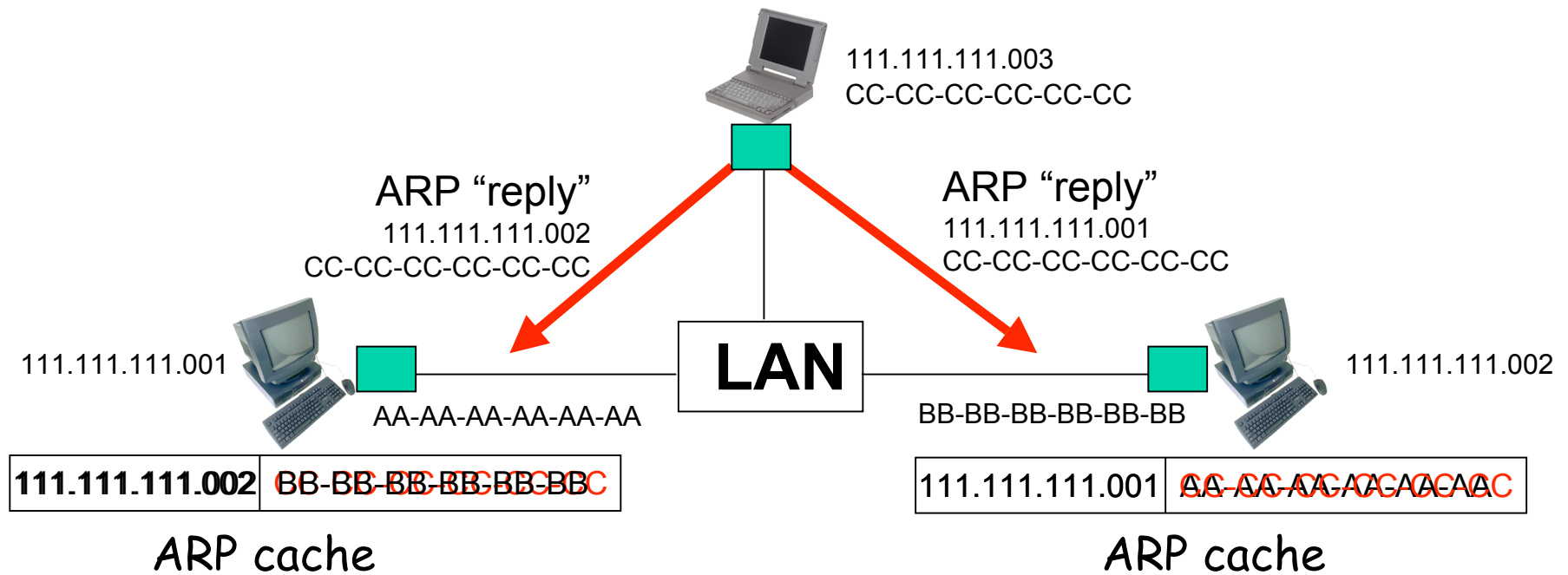
ARP

- ❑ ARP is stateless
- ❑ ARP sends **request** and receives ARP **reply**
- ❑ Replies used to fill ARP cache



ARP Cache Poisoning

- ARP is stateless
- Accepts any **reply**, even if no **request** sent!



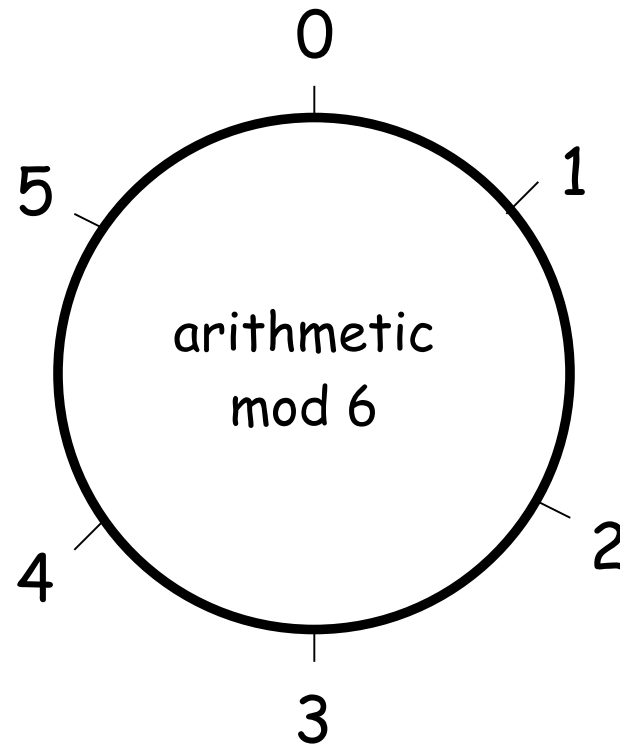
- Host CC-CC-CC-CC-CC-CC is "man-in-the-middle"

Math Basics

Modular Arithmetic

"Clock" Arithmetic

- For integers x and n , $x \bmod n$ is the remainder of $x \div n$
- Examples
 - $7 \bmod 6 = 1$
 - $33 \bmod 5 = 3$
 - $33 \bmod 6 = 3$
 - $51 \bmod 17 = 0$
 - $17 \bmod 6 = 5$



Modular Addition

□ Notation and facts

- $7 \bmod 6 = 1$
- $7 = 13 = 1 \bmod 6$
- $((a \bmod n) + (b \bmod n)) \bmod n = (a + b) \bmod n$
- $((a \bmod n)(b \bmod n)) \bmod n = ab \bmod n$

□ Addition Examples

- $3 + 5 = 2 \bmod 6$
- $2 + 4 = 0 \bmod 6$
- $3 + 3 = 0 \bmod 6$
- $(7 + 12) \bmod 6 = 19 \bmod 6 = 1 \bmod 6$
- $(7 + 12) \bmod 6 = (1 + 0) \bmod 6 = 1 \bmod 6$

Modular Multiplication

□ Multiplication Examples

- $3 \cdot 4 = 0 \pmod{6}$
- $2 \cdot 4 = 2 \pmod{6}$
- $5 \cdot 5 = 1 \pmod{6}$
- $(7 \cdot 4) \pmod{6} = 28 \pmod{6} = 4 \pmod{6}$
- $(7 \cdot 4) \pmod{6} = (1 \cdot 4) \pmod{6} = 4 \pmod{6}$

Modular Inverses

- Additive inverse of $x \bmod n$, denoted $-x$, is the number that must be added to x to get $0 \bmod n$
 - $-2 \bmod 6 = 4$, since $2 + 4 = 0 \bmod 6$
- Multiplicative inverse of $x \bmod n$, denoted x^{-1} , is the number that must be multiplied by x to get $1 \bmod n$
 - $3^{-1} \bmod 7 = 5$, since $3 \cdot 5 = 1 \bmod 7$

Modular Arithmetic Quiz

- ❑ Q: What is $-3 \pmod{6}$?
- ❑ A: 3
- ❑ Q: What is $-1 \pmod{6}$?
- ❑ A: 5
- ❑ Q: What is $5^{-1} \pmod{6}$?
- ❑ A: 5
- ❑ Q: What is $2^{-1} \pmod{6}$?
- ❑ A: No number works!
- ❑ Multiplicative inverse might not exist

Relative Primality

- x and y are **relatively prime** if they have no common factor other than 1
- $x^{-1} \bmod y$ exists only when x and y are relatively prime
- $x^{-1} \bmod y$ is easy to find (when it exists) using the Euclidean Algorithm

Totient Function

- $\varphi(n)$ is the number of numbers (positive integers) less than n , relatively prime to n
- Examples
 - $\varphi(4) = 2$ since 4 is relatively prime to 3 and 1
 - $\varphi(5) = 4$ since 5 is relatively prime to 1,2,3 and 4
 - $\varphi(12) = 4$
 - $\varphi(p) = p-1$ if p is prime
 - $\varphi(pq) = (p-1)(q-1)$ if p and q prime

Permutations

Permutation Definition

- Let S be a set
- A permutation of S is an ordered list of the elements of S
 - Each element of S appears exactly once
- Suppose $S = \{0, 1, 2, \dots, n-1\}$
 - Then the number of perms is...
 - $n(n-1)(n-2) \cdots (2)(1) = n!$

Permutation Example

- Let $S = \{0,1,2,3\}$
- Then there are 24 perms of S
- For example,
 - $(3,1,2,0)$ is a perm of S
 - $(0,2,3,1)$ is a perm of S , etc.
- Perms are important in cryptography

Probability Basics

Discrete Probability

- We only require some elementary facts
- Suppose that $S = \{0, 1, 2, \dots, N-1\}$ is the set of all possible outcomes
- If each outcome is equally likely, then the probability of event $E \subseteq S$ is
 - $P(E) = \# \text{ elements of } E / \# \text{ elements of } S$

Probability Example

- For example, suppose we flip 2 coins
- Then $S = \{hh, ht, th, tt\}$
 - Suppose $X = \text{"at least one tail"} = \{ht, th, tt\}$
 - Then $P(X) = 3/4$
- Often, it's easier to compute
 - $P(X) = 1 - P(\text{complement of } X)$

Complement

- Again, suppose we flip 2 coins
- Let $S = \{hh, ht, th, tt\}$
 - Suppose $X = \text{"at least one tail"} = \{ht, th, tt\}$
 - Complement of X is "no tails" = $\{tt\}$
- Then
 - $P(X) = 1 - P(\text{comp. of } X) = 1 - 1/4 = 3/4$
- We'll make use of this trick often!

Linear Algebra Basics

Vectors and Dot Product

- Let \mathfrak{R} be the set of real numbers
- Then $v \in \mathfrak{R}^n$ is a vector of n elements
- For example
 - $v = [v_1, v_2, v_3, v_4] = [2, -1, 3.2, 7] \in \mathfrak{R}^4$
- The dot product of $u, v \in \mathfrak{R}^n$ is
 - $u \cdot v = u_1v_1 + u_2v_2 + \dots + u_nv_n$

Matrix

- A matrix is an $n \times m$ array
- For example, the matrix A is 2×3

$$A = \begin{bmatrix} 3 & 4 & 2 \\ 1 & 7 & 9 \end{bmatrix}$$

- The element in row i column j is a_{ij}
- We can multiply a matrix by a number

$$3A = \begin{bmatrix} 3 \cdot 3 & 3 \cdot 4 & 3 \cdot 2 \\ 3 \cdot 1 & 3 \cdot 7 & 3 \cdot 9 \end{bmatrix} = \begin{bmatrix} 9 & 12 & 6 \\ 3 & 21 & 27 \end{bmatrix}$$

Matrix Addition

- We can add matrices of the same size

$$\begin{bmatrix} 3 & 2 \\ 1 & 5 \end{bmatrix} + \begin{bmatrix} -1 & 4 \\ 6 & 2 \end{bmatrix} = \begin{bmatrix} 2 & 6 \\ 7 & 7 \end{bmatrix}.$$

- We can also multiply matrices, but this is not so obvious
- We do **not** simply multiply the elements

Matrix Multiplication

- Suppose A is $m \times n$ and B is $s \times t$
- Then $C=AB$ is only defined if $n=s$, in which case C is $m \times t$
- Why?
- The element c_{ij} is the dot product of row i of A with column j of B

Matrix Multiply Example

□ Suppose

$$B = \begin{bmatrix} -1 & 2 \\ 2 & -3 \end{bmatrix} \quad A = \begin{bmatrix} 3 & 4 & 2 \\ 1 & 7 & 9 \end{bmatrix}$$

□ Then

$$BA = C_{2 \times 3} = \begin{bmatrix} [-1, 2] \cdot \begin{bmatrix} 3 \\ 1 \end{bmatrix} & [-1, 2] \cdot \begin{bmatrix} 4 \\ 7 \end{bmatrix} & [-1, 2] \cdot \begin{bmatrix} 2 \\ 9 \end{bmatrix} \\ [2, -3] \cdot \begin{bmatrix} 3 \\ 1 \end{bmatrix} & [2, -3] \cdot \begin{bmatrix} 4 \\ 7 \end{bmatrix} & [2, -3] \cdot \begin{bmatrix} 2 \\ 9 \end{bmatrix} \end{bmatrix} = \begin{bmatrix} -1 & 10 & 16 \\ 3 & -13 & -23 \end{bmatrix}$$

□ And AB is undefined

Matrix Multiply Useful Fact

- Consider $AU = B$ where A is a matrix and U and B are column vectors
- Let a_1, a_2, \dots, a_n be columns of A and u_1, u_2, \dots, u_n the elements of U
- Then $B = u_1 a_1 + u_2 a_2 + \dots + u_n a_n$

Example:

$$\begin{bmatrix} 3 & 4 \\ 1 & 5 \end{bmatrix} \begin{bmatrix} 2 \\ 6 \end{bmatrix} = 2 \begin{bmatrix} 3 \\ 1 \end{bmatrix} + 6 \begin{bmatrix} 4 \\ 5 \end{bmatrix} = \begin{bmatrix} 30 \\ 32 \end{bmatrix}$$

Identity Matrix

- A matrix is square if it has an equal number of rows and columns
- For square matrices, the identity matrix I is the multiplicative identity
 - $AI = IA = A$
- The 3 x 3 identity matrix is

$$I = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Block Matrices

- Block matrices are matrices of matrices
- For example

$$M = \begin{bmatrix} I_{n \times n} & C_{n \times 1} \\ A_{m \times n} & B_{m \times 1} \end{bmatrix} \quad \text{and} \quad V = \begin{bmatrix} U_{n \times \ell} \\ T_{1 \times \ell} \end{bmatrix}$$

- We can do arithmetic with block matrices
- Block matrix multiplication works if individual matrix dimensions “match”

Block Matrix Multiplication

- Block matrices multiplication example
- For matrices

$$M = \begin{bmatrix} I_{n \times n} & C_{n \times 1} \\ A_{m \times n} & B_{m \times 1} \end{bmatrix} \quad \text{and} \quad V = \begin{bmatrix} U_{n \times \ell} \\ T_{1 \times \ell} \end{bmatrix}$$

- We have

$$MV = \begin{bmatrix} X_{n \times \ell} \\ Y_{m \times \ell} \end{bmatrix}$$

- Where $X = U + CT$ and $Y = AU + BT$

Linear Independence

- Vectors $u, v \in \mathfrak{R}^n$ **linearly independent** if $au + bv = 0$ implies $a=b=0$
- For example,

$$\begin{bmatrix} 1 \\ -1 \end{bmatrix} \text{ and } \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

- Are linearly independent

Linear Independence

- Linear independence can be extended to more than 2 vectors
- If vectors are linearly independent, then none of them can be written as a *linear combination* of the others
 - None of the independent vectors is a sum of multiples of the other vectors