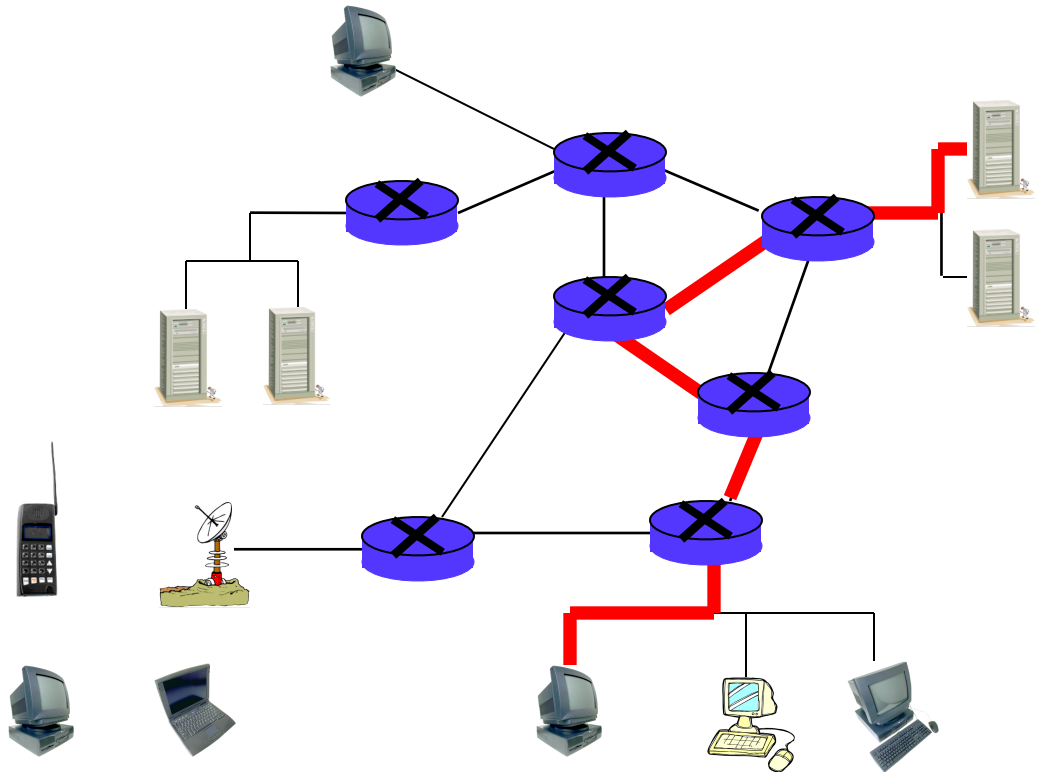


Networking Basics

There are three kinds of death in this world.
There's heart death, there's brain death, and there's being off the network.
— Guy Almes

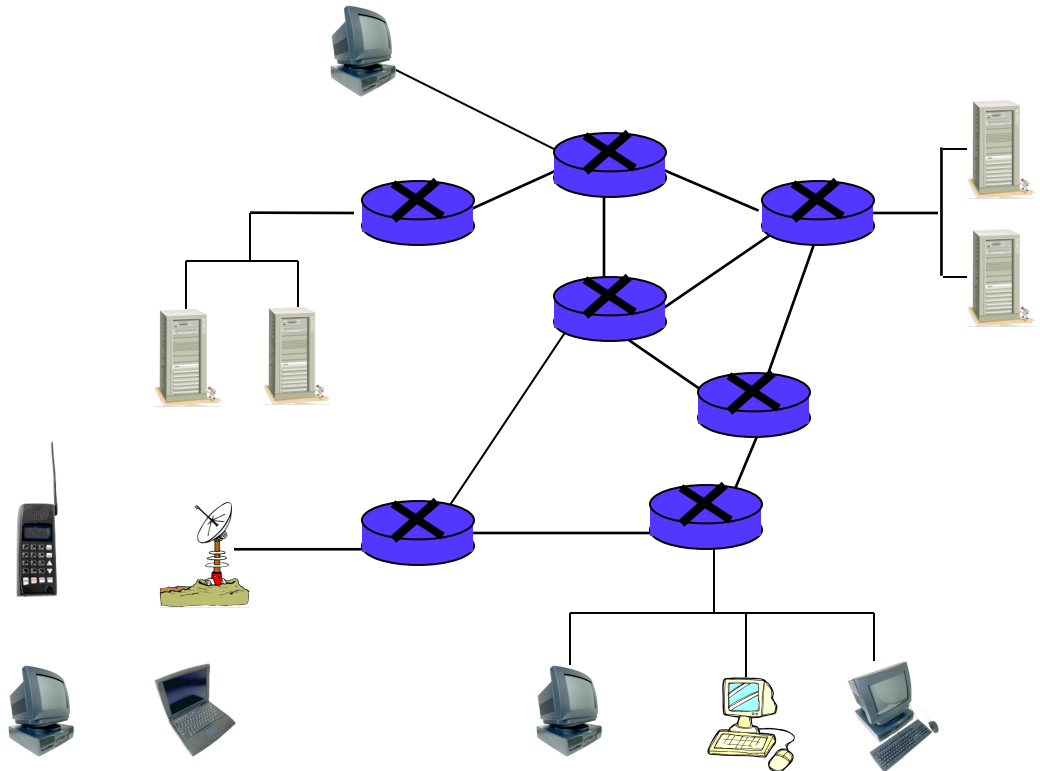
Network

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



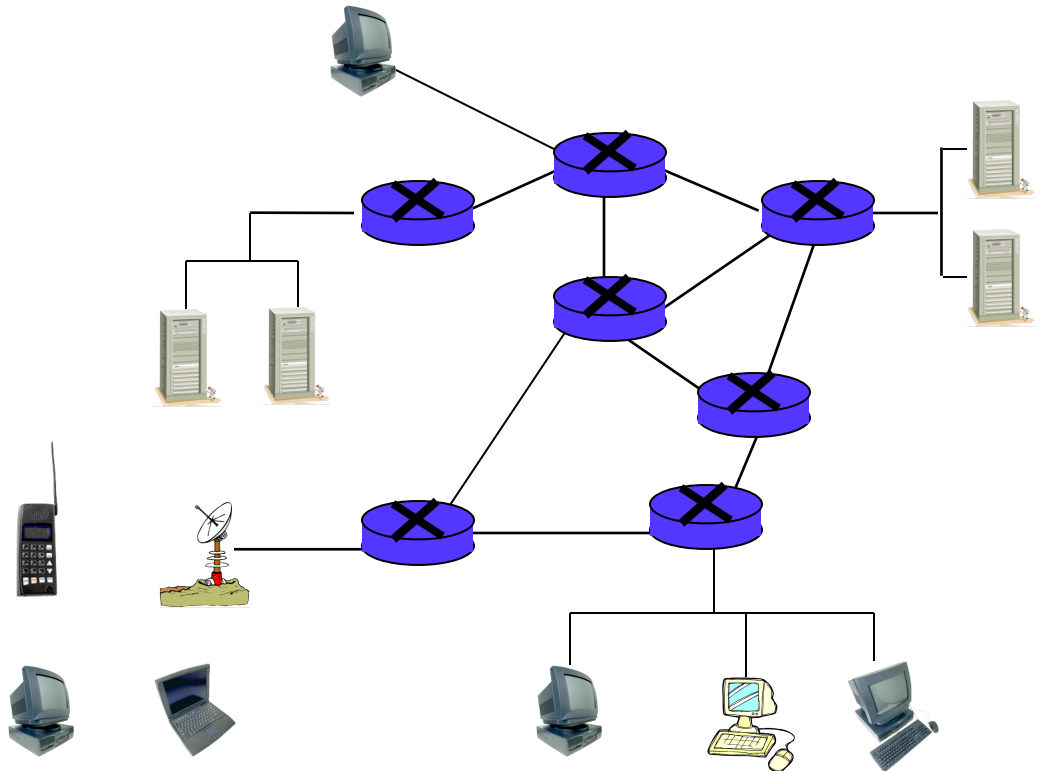
Network Edge

- ❑ 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

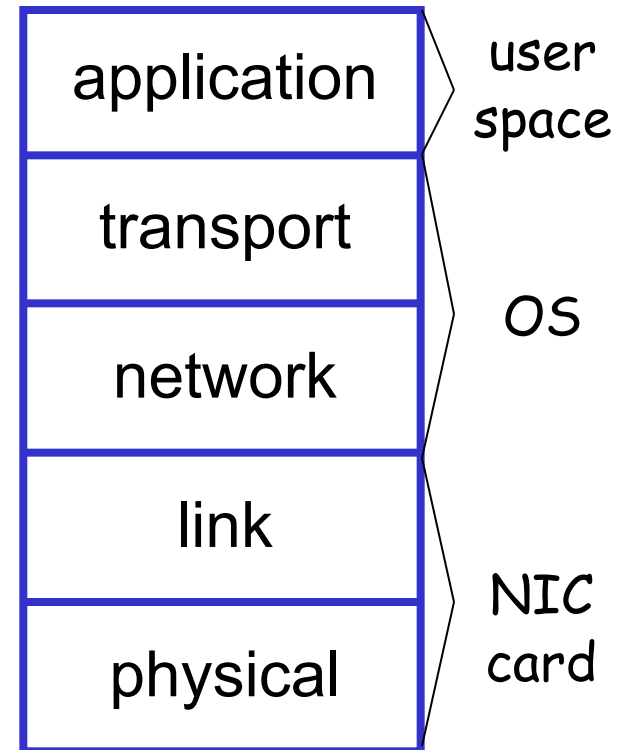
- ❑ Telephone network is/was **circuit switched**
 - For each call, a dedicated circuit established
 - Dedicated bandwidth
- ❑ Modern data networks are **packet switched**
 - Data is chopped up into discrete packets
 - Packets are transmitted independently
 - No dedicated circuit is established
 - +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

Network Protocols

- ❑ Study of networking focused on **protocols**
- ❑ Networking protocols precisely specify "communication rules"
- ❑ Details are given in **RFCs**
 - RFC is essentially an Internet standard
- ❑ **Stateless** protocols do not "remember"
- ❑ **Stateful** protocols do "remember"
- ❑ Many security problems related to state
 - E.g., DoS is a problem with stateful protocols

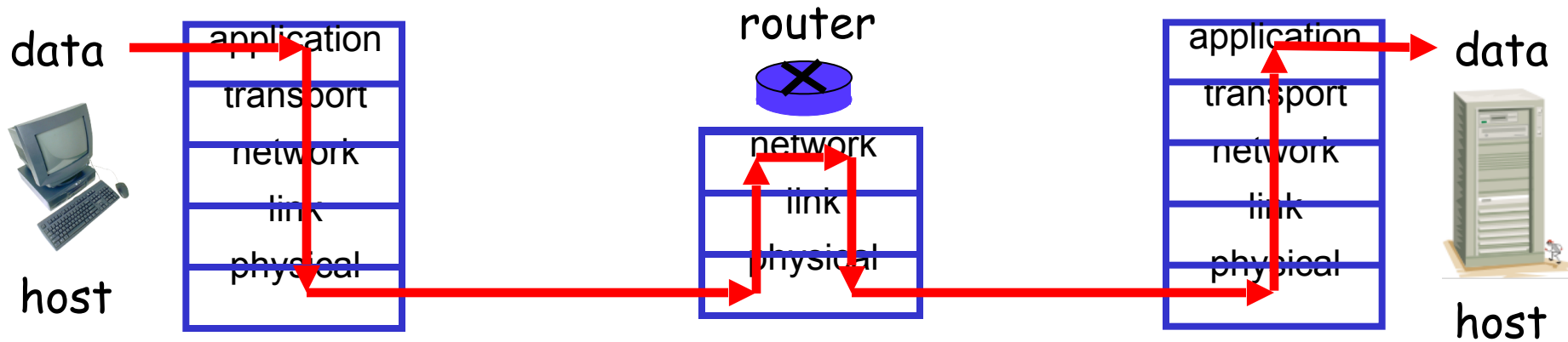
Protocol Stack

- ❑ 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



- **NIC:** Network Interface Controller is basically the hardware that allows the connection to your computer
- **PPP:** Point-to-Point Protocol (like the connection between two routers)

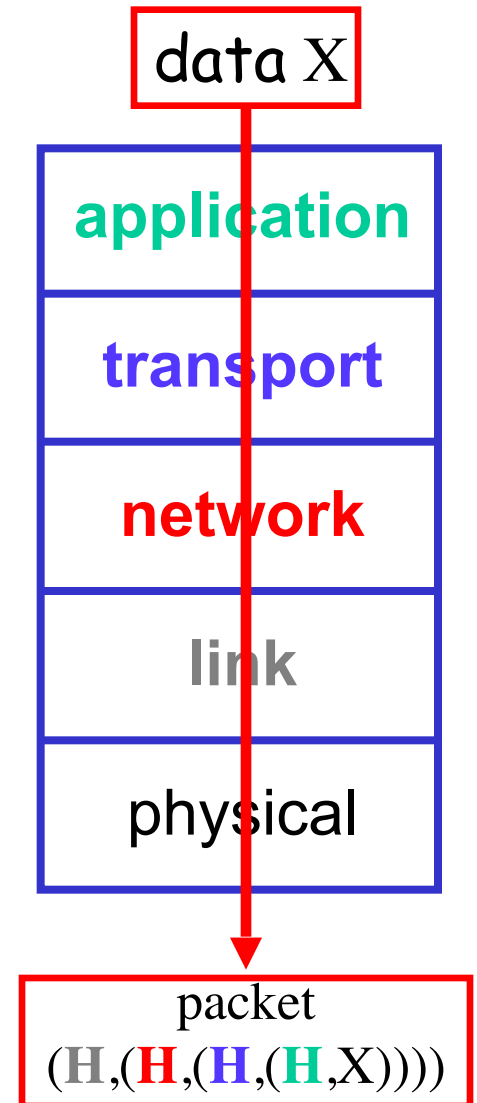
Layering in Action



- ❑ 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 packet up to application layer
 - That's where the application data lives

Encapsulation

- ❑ X = application data at 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 data is on the “inside”



Application Layer

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

Client-Server Model

- ❑ **Client**

- "speaks first"

- ❑ **Server**

- responds to client's request

- ❑ **Hosts are clients or servers**

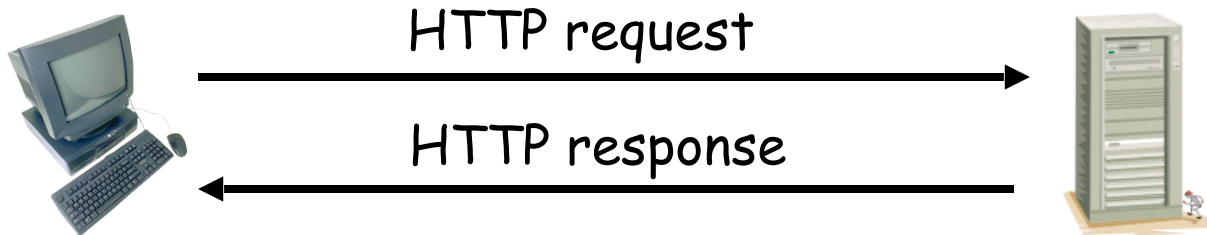
- ❑ **Example: Web browsing**

- You are the client (request web page)
 - Web server is the server

Peer-to-Peer Paradigm

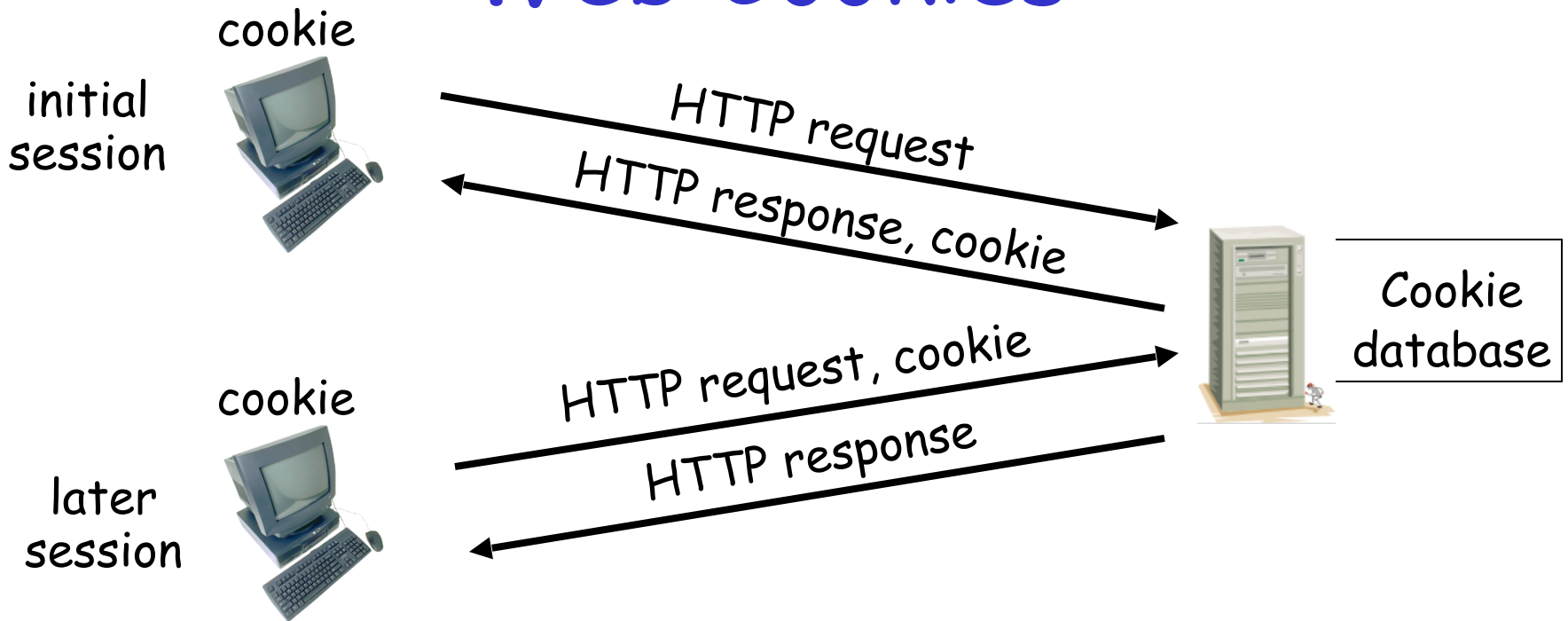
- ❑ Hosts act as clients and servers
- ❑ For example, when sharing music
 - You are client when requesting a file
 - You are a server when someone downloads a file from you
- ❑ In P2P, how does client find server?
 - Many different P2P models for this

HTTP Example



- ❑ HTTP — **H**yper**T**ext **T**ransfer **P**rotocol
- ❑ Client (you) requests a web page
- ❑ Server responds to your request

Web Cookies



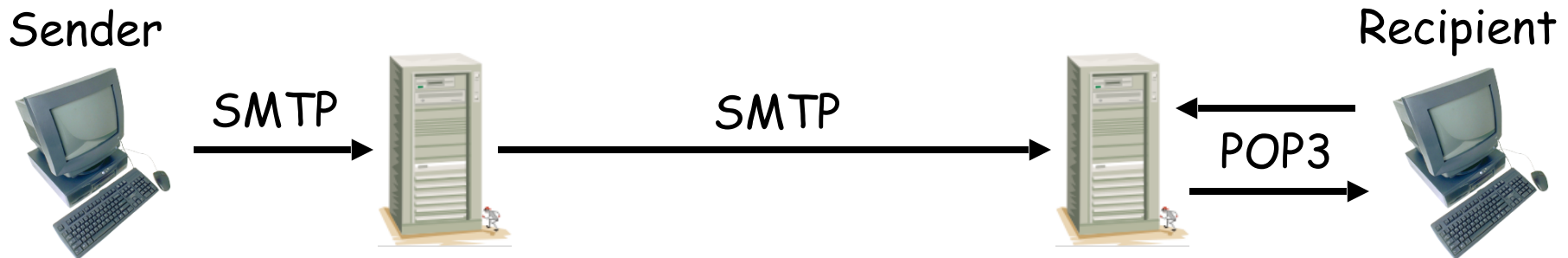
- HTTP is stateless — cookies used to add state
- Initially, cookie sent from server to browser
- Browser manages cookie, sends it to server
- Server uses cookie database to "remember" you

Web Cookies

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

SMTP

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



Application Layer

- ❑ DNS — Domain Name Service
 - Convert human-friendly names such as www.sjsu.com into 32-bit IP address
[https://130.65.218.11/]
 - A distributed hierarchical database
- ❑ Only 13 “root” DNS server clusters
 - Essentially, a single point of failure for Internet
 - Attacks on root servers have succeeded...
 - ...but, attacks did not last long enough (yet)

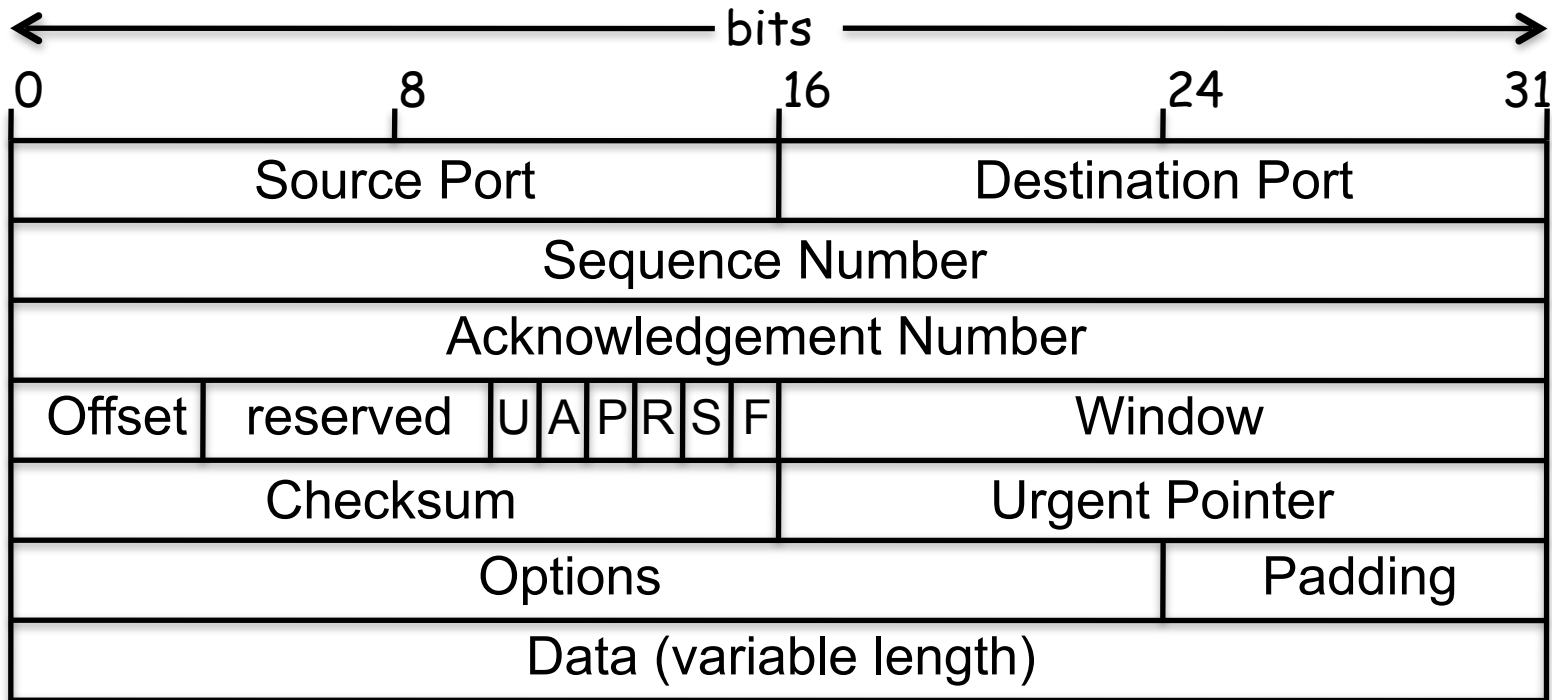
Transport Layer

- ❑ The network layer offers unreliable, “best effort” delivery of packets
- ❑ Any improved service must be provided by the hosts
- ❑ Transport layer: 2 protocols of interest
 - TCP — **more** service, **more** overhead
 - UDP — **less** service, **less** overhead
- ❑ TCP and UDP run on hosts, not routers

TCP

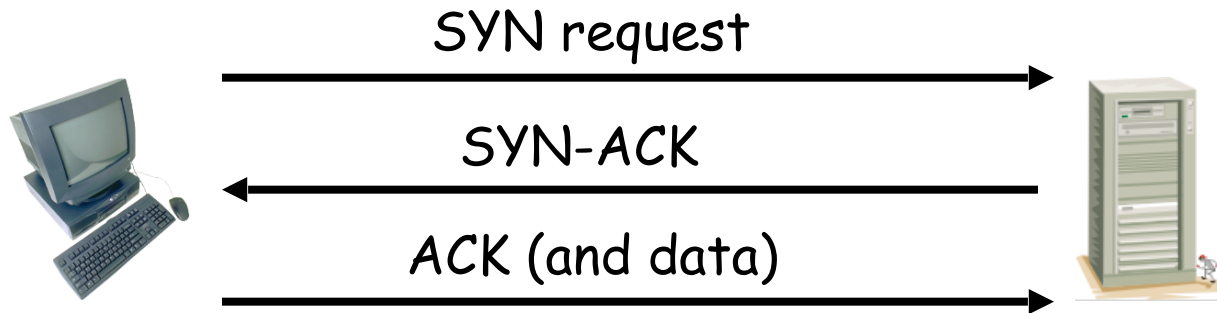
- ❑ TCP assures that packets...
 - Arrive at destination
 - Are processed in order
 - Are not sent too fast for receiver: flow control
- ❑ TCP also attempts to provide...
 - Network-wide congestion control
- ❑ TCP is connection-oriented
 - TCP contacts server before sending data
 - Orderly setup and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

TCP Header



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

TCP Three-Way Handshake



- ❑ **SYN** — synchronization requested
- ❑ **SYN-ACK** — acknowledge SYN request
- ❑ **ACK** — acknowledge SYN-ACK (send data)
- ❑ Then TCP "connection" established
 - Connection terminated by FIN or RST

Denial of Service Attack

- ❑ The TCP 3-way handshake makes denial of service (DoS) attacks possible
- ❑ Whenever SYN packet is received, server remembers this "half-open" connection
 - Remembering consumes resources
 - Too many half-open connections and server's resources will be exhausted, and then...
 - ...server can't respond to legitimate connections
- ❑ This occurs because TCP is **stateful**

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 (header only 8 bytes)
 - No flow control to slow down sender
 - No congestion control to slow down sender
- ❑ If packets sent too fast, will be dropped
 - Either at intermediate router or at destination
 - But in some apps this may be 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 of interest 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, ...

IP Addresses

- ❑ **IP address** is 32 (IPv4) - 128 (IPv6) bits
- ❑ Every host has an IP address
- ❑ IP addresses given in dotted decimal notation
 - For example: 195.72.180.27
 - Each number is between 0 and 255
- ❑ Usually, a host's IP address can change

Socket

- ❑ Many processes can run on one host
 - E.g., you can browse web, send email at same time
- ❑ How to distinguish processes on a host?
- ❑ Each process has a 16 bit **port number**
 - Numbers below 1024 are "well-known" ports (HTTP is port 80, POP3 is port 110, etc.)
 - Port numbers above 1024 are dynamic (as needed)
- ❑ IP address + port number = **socket**
 - Socket uniquely identifies **process**, Internet-wide

Network Address Translation

- ❑ Network Address Translation (**NAT**)
 - Trick to extend IP address space
- ❑ Use one IP address (different port numbers) for multiple hosts
 - “Translates” outside IP address (based on port number) to inside IP address

NAT-less Example



Web
server

IP: 12.0.0.1

Port: 80

← source 11.0.0.1:1025
destination 12.0.0.1:80

source 12.0.0.1:80
→ destination 11.0.0.1:1025



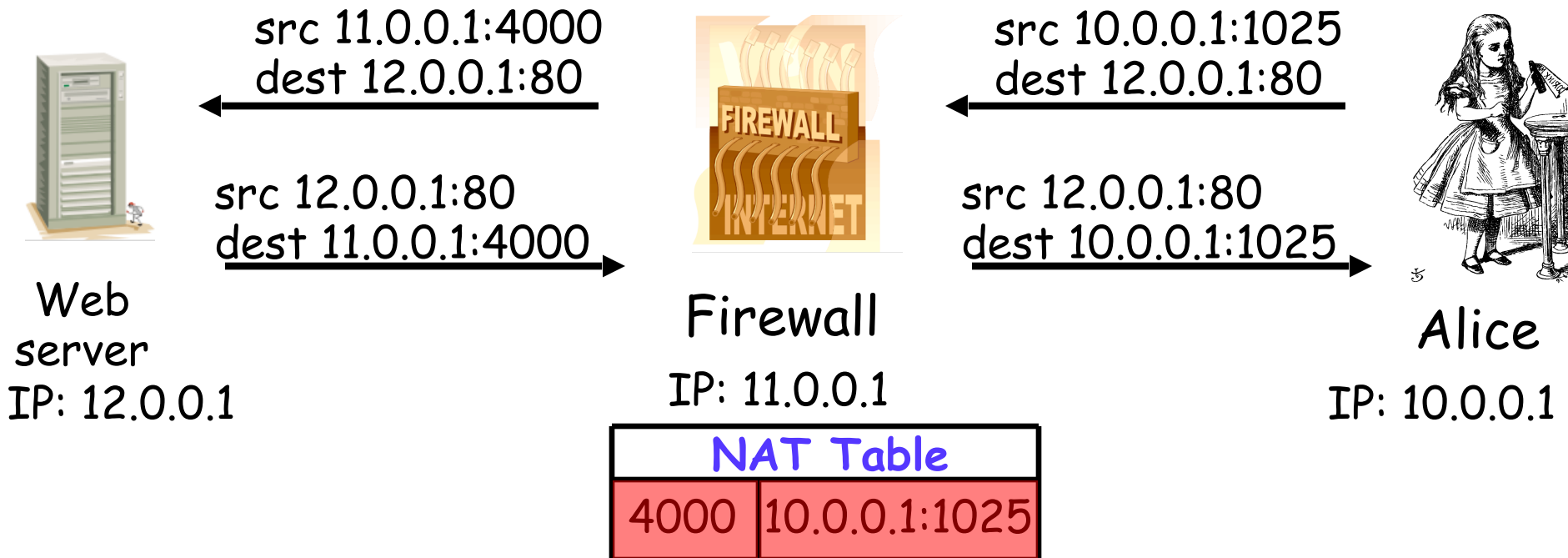
ct

Alice

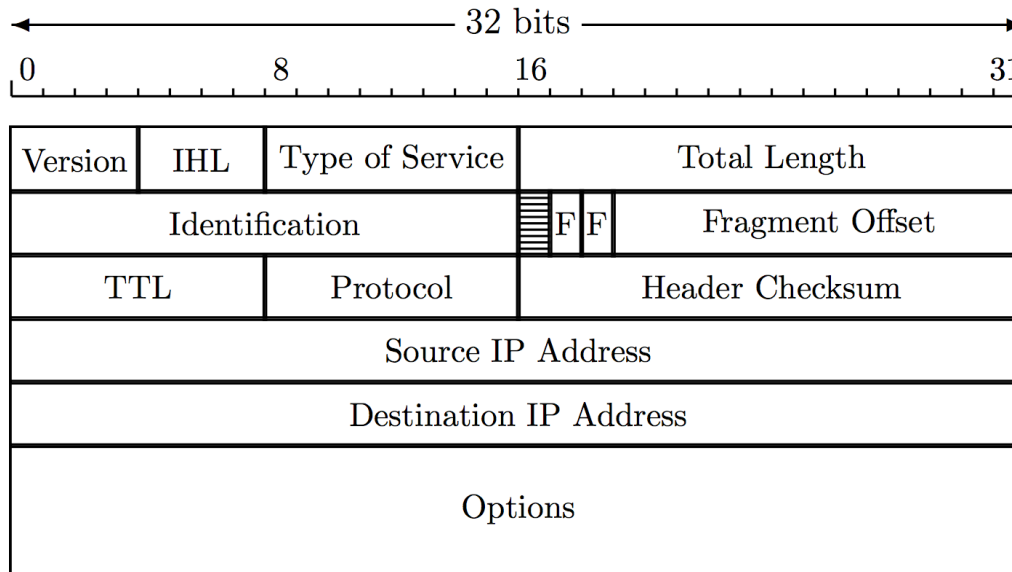
IP: 11.0.0.1

Port: 1025

NAT Example

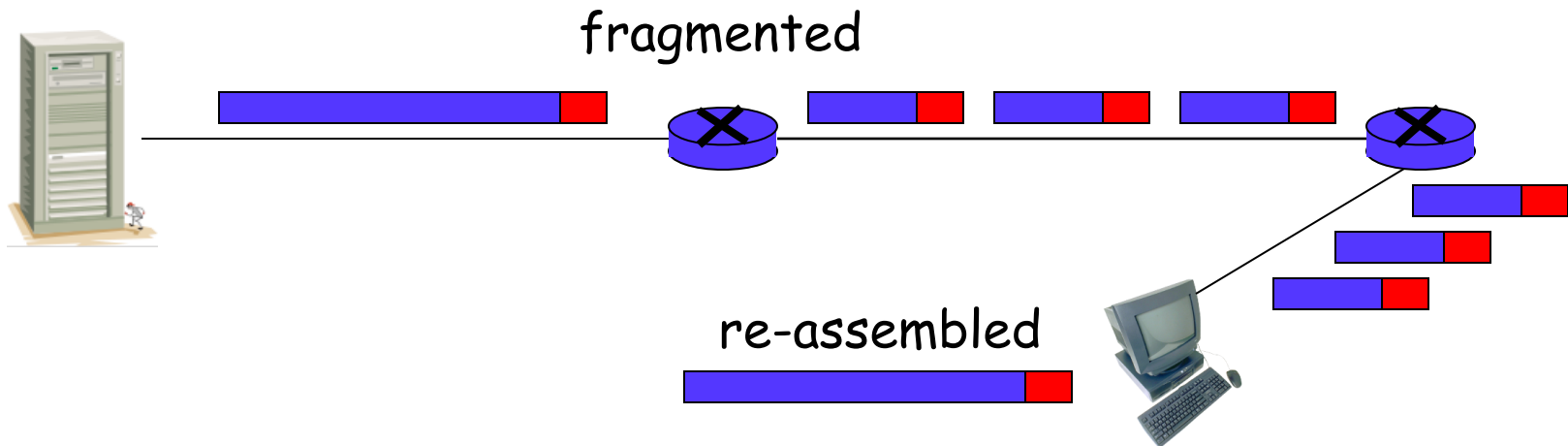


IP Header



- ❑ IP header has necessary info for routers
 - E.g., 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 fragmentation/reassemble
- ❑ Fragmentation is a **security issue**...
 - Fragments may **obscure real purpose of packet**
 - Fragments **can overlap** when reassembled
 - Must reassemble packet to fully understand it
 - Lots of work for firewalls, for example

IPv6

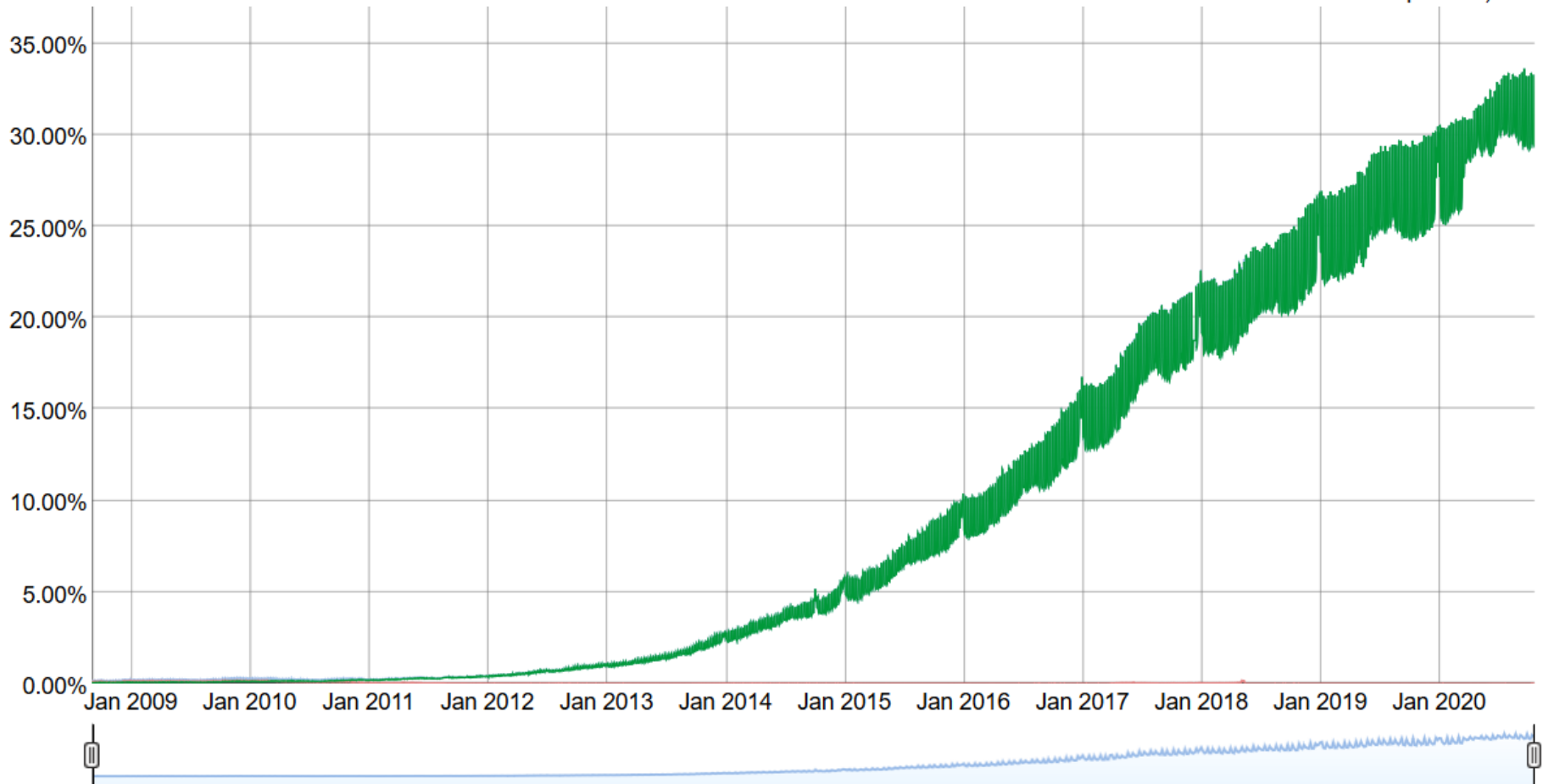
- ❑ Current version of IP is IPv4
- ❑ IPv6 is a "new-and-improved" version of IP
- ❑ IPv6 is "bigger and better" than IPv4
 - Bigger addresses: 128 bits
 - Better security: IPSec
- ❑ How to migrate from IPv4 to IPv6?
 - Unfortunately, nobody thought about that...
- ❑ So IPv6 has not really taken hold (yet?)

IPv6 adoption

IPv6 Adoption

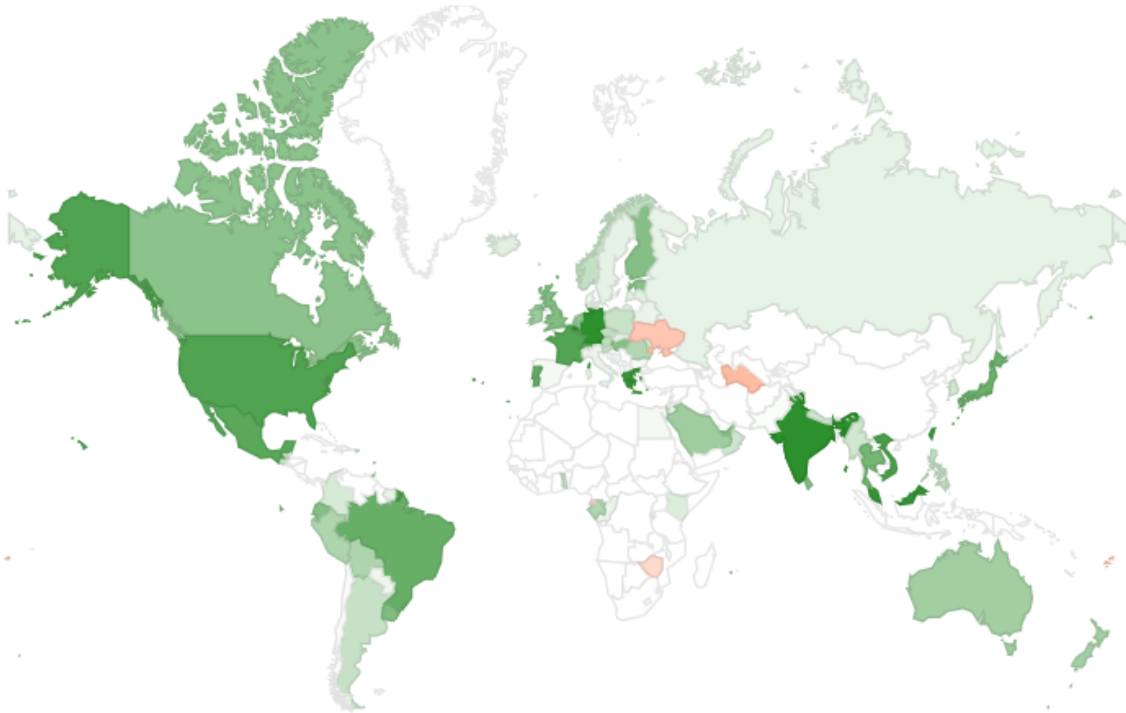
We are continuously measuring the availability of IPv6 connectivity among Google users. The graph shows the percentage of users that access Google over IPv6.

Native: 0.22% 6to4/Teredo: 0.03% Total IPv6: 0.25% | Mar 17, 2011






IPv6 adoption

Per-Country IPv6 adoption



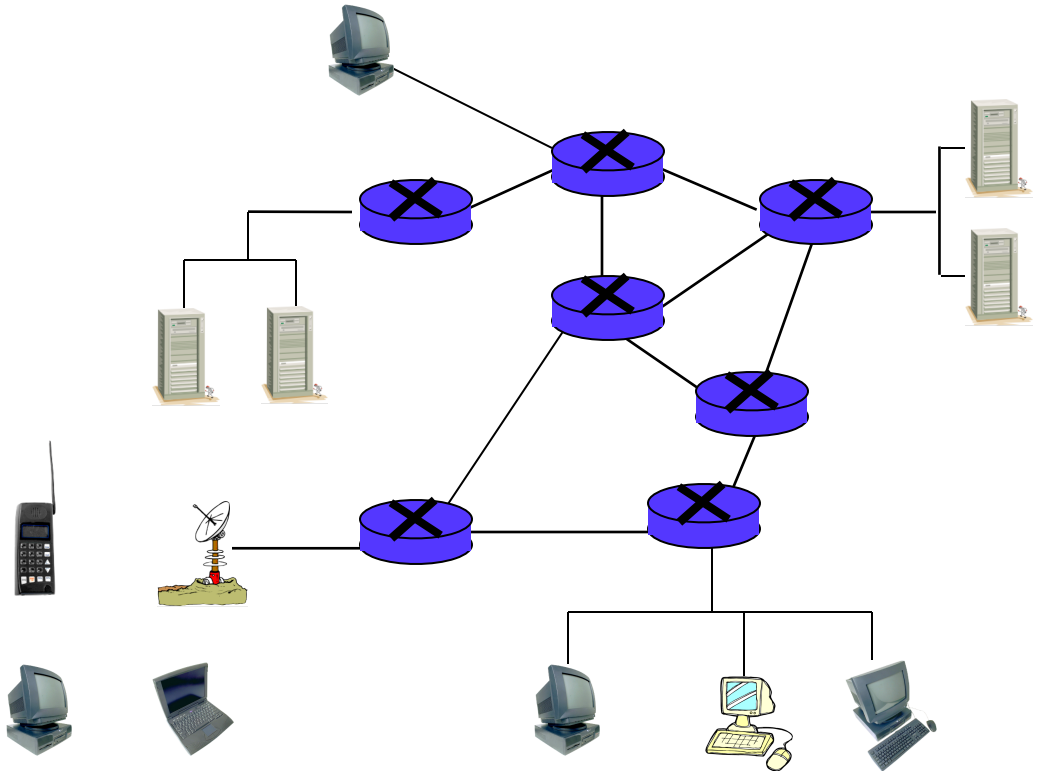
[World](#) | [Africa](#) | [Asia](#) | [Europe](#) | [Oceania](#) | [North America](#) | [Central America](#) | [Caribbean](#) | [South America](#)

The chart above shows the availability of IPv6 connectivity around the world.

-  Regions where IPv6 is more widely deployed (the darker the green, the greater the deployment) and users experience infrequent issues connecting to IPv6-enabled websites.
-  Regions where IPv6 is more widely deployed but users still experience significant reliability or latency issues connecting to IPv6-enabled websites.
-  Regions where IPv6 is not widely deployed and users experience significant reliability or latency issues connecting to IPv6-enabled websites.

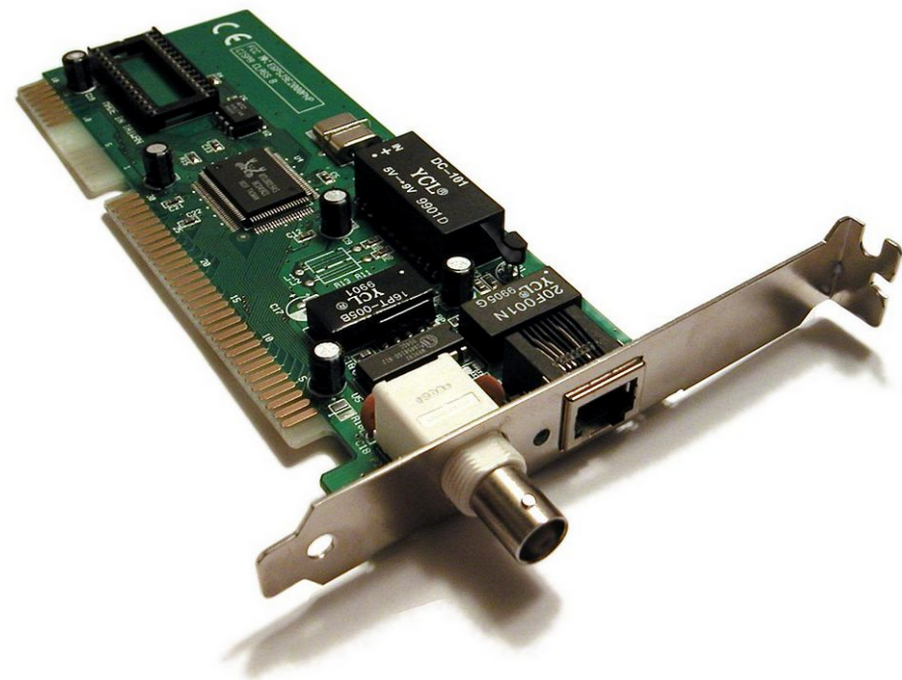
Link Layer

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

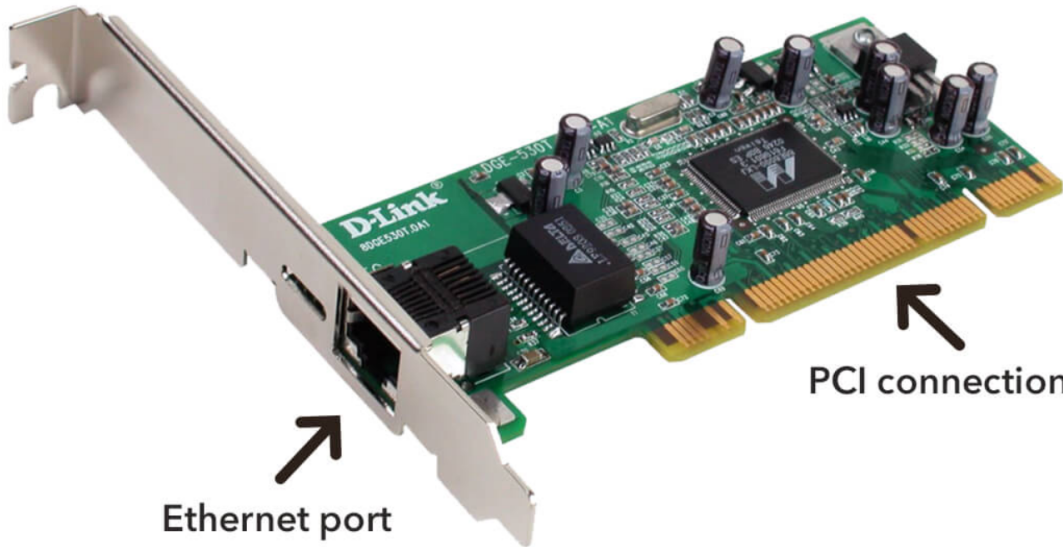


Link Layer

- ❑ On host, implemented in adapter:
Network Interface Card (NIC)
 - Ethernet card, wireless 802.11 card, etc.
 - NIC is "semi-autonomous" device
- ❑ NIC is (mostly) out of host's control
 - Implements both link and physical layers



Gigabit Ethernet NIC



Ethernet

- ❑ Ethernet is a **multiple access** protocol
- ❑ Many hosts access a shared media
 - On a local area network, or LAN
- ❑ With multiple access, packets can “collide”
 - Data is corrupted and packets must be resent
- ❑ How to efficiently deal with collisions in distributed environment?
 - Many possibilities, ethernet is most popular
- ❑ We won't discuss details here...

Link Layer Addressing

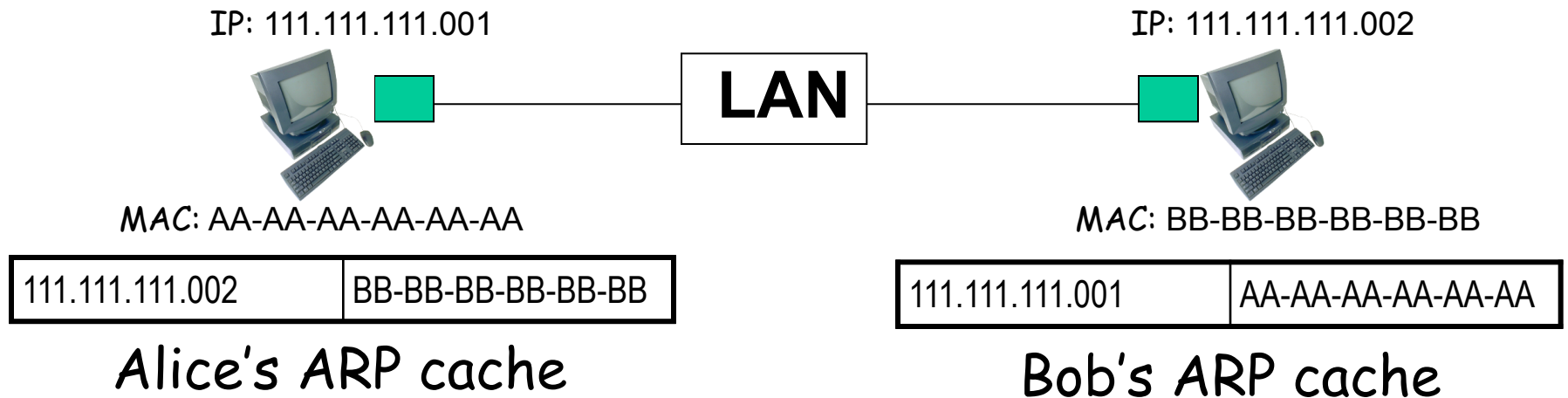
- ❑ IP addresses live at network layer
- ❑ Link layer also needs addresses — Why?
 - **MAC address** (LAN address, physical address)
- ❑ MAC address (Media Access Control)
 - 48 bits, globally unique
 - Used to forward packets over one link
- ❑ Analogy...
 - IP address is like your home address
 - MAC address is like a social security number

ARP

- ❑ Address Resolution Protocol (ARP)
- ❑ Used by link layer — given IP address, *find corresponding MAC address*
- ❑ Each host has **ARP table**, or **ARP cache**
 - Generated automatically
 - Entries expire after some time (about 20 min)
 - ARP used to find ARP table entries

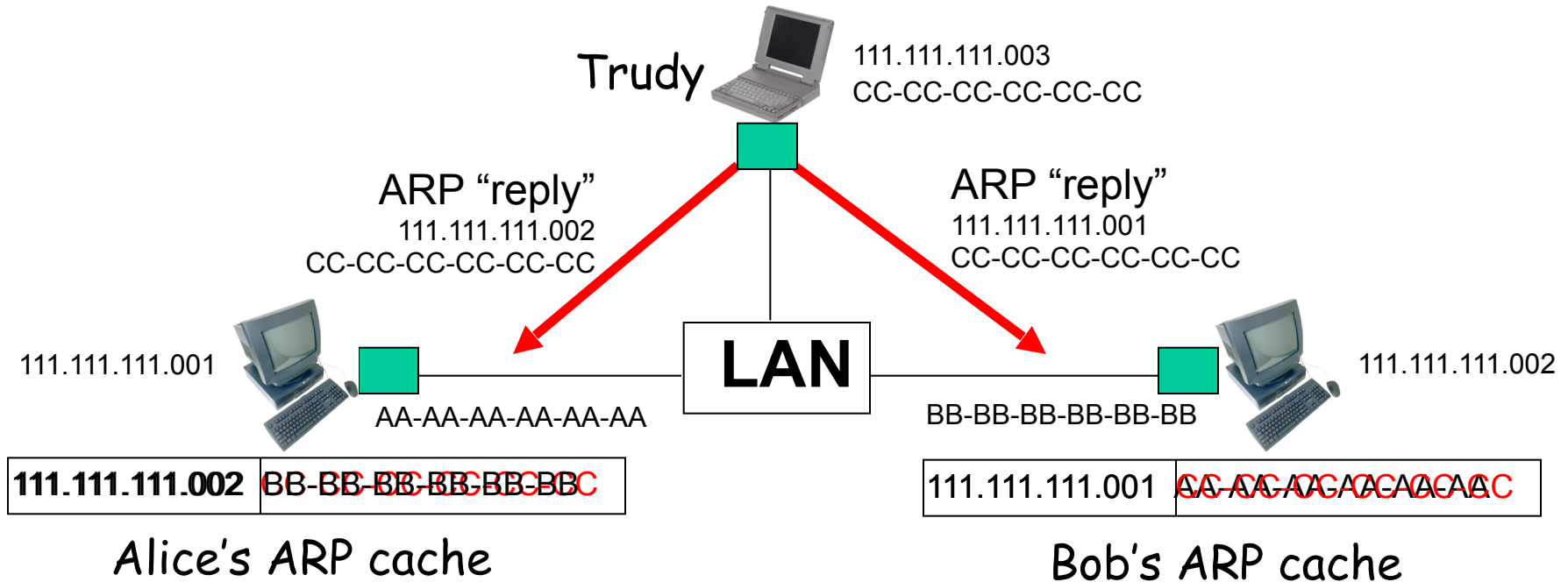
ARP

- ❑ ARP is **stateless**
- ❑ ARP can send **request** and receive **reply**
- ❑ Reply msgs used to fill/update ARP cache



ARP Cache Poisoning

- ARP is stateless, so...
- Accept “reply”, even if no request sent



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