

Network Basics and Security

Shuai Wang



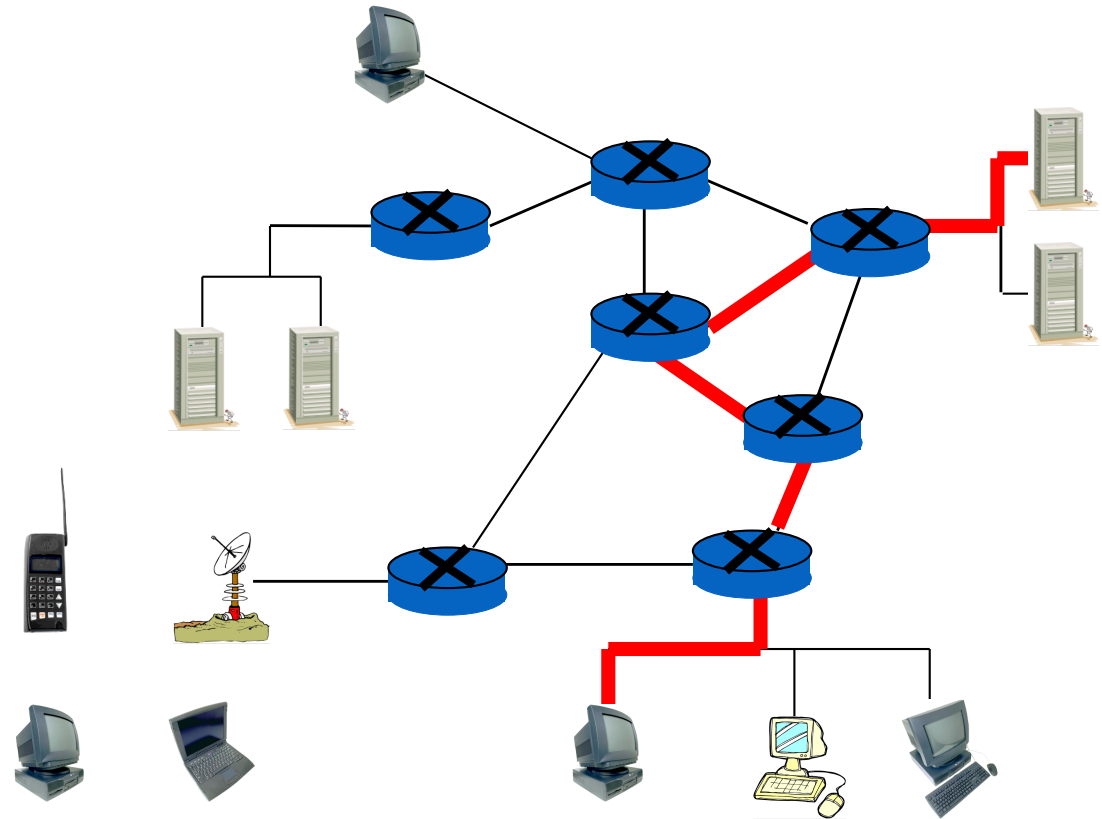
香港科技大學

THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Some of the slides are from Mark Stamp.

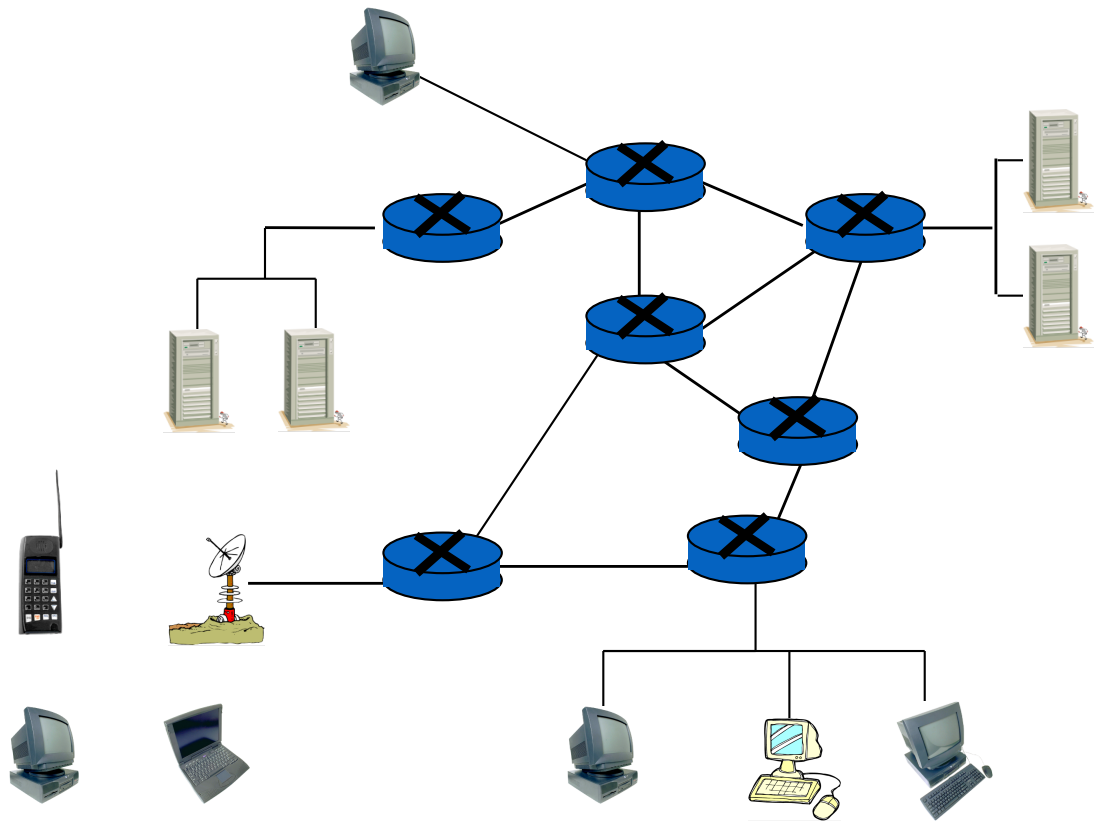
Networking Basics

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



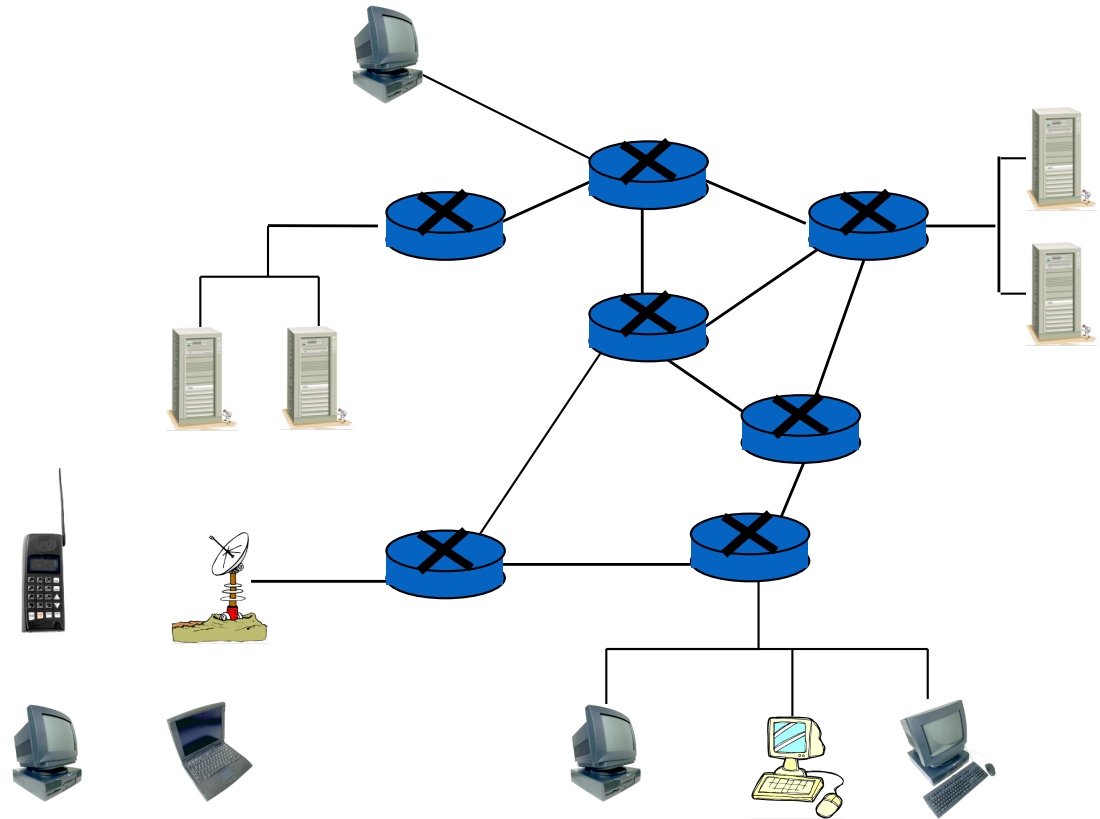
Network Hosts

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



Network Routers

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



Packet Switched Network

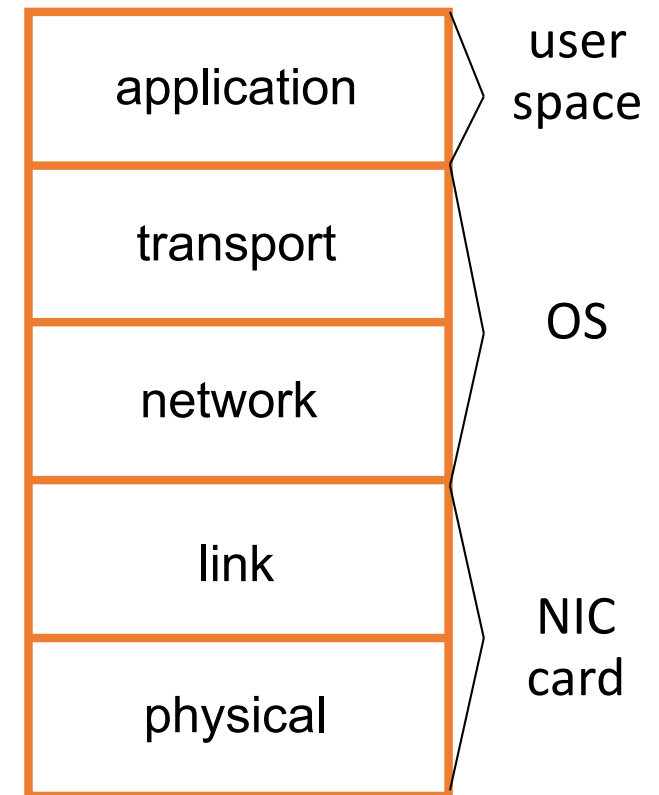
- Telephone network is/was **circuit switched**
 - For each call, a **dedicated** circuit established
 - Dedicated bandwidth → **lots of waste**
- Modern data networks are **packet switched**
 - Data is chopped up into **discrete packets**
 - Packets are transmitted independently
 - No dedicated circuit 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

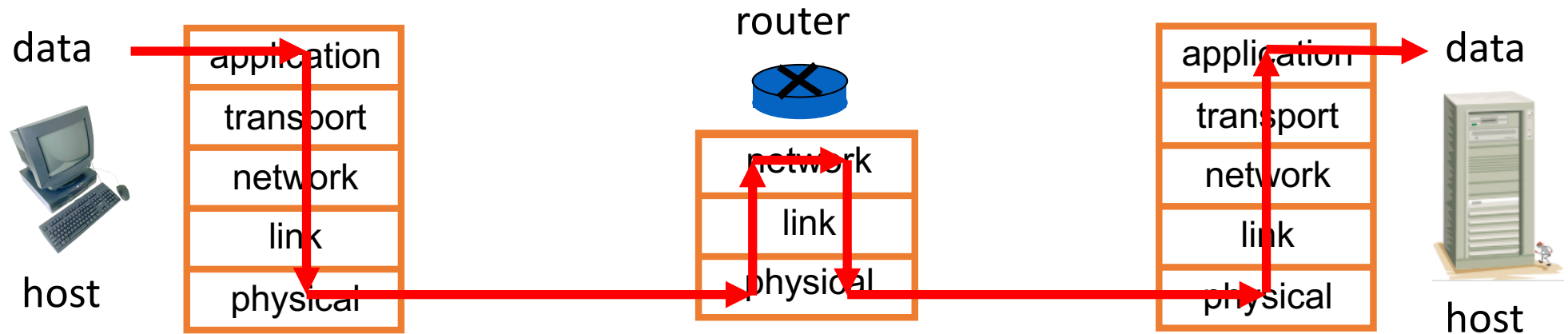
Protocol Stack (OSI Model)

- 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



Five layers

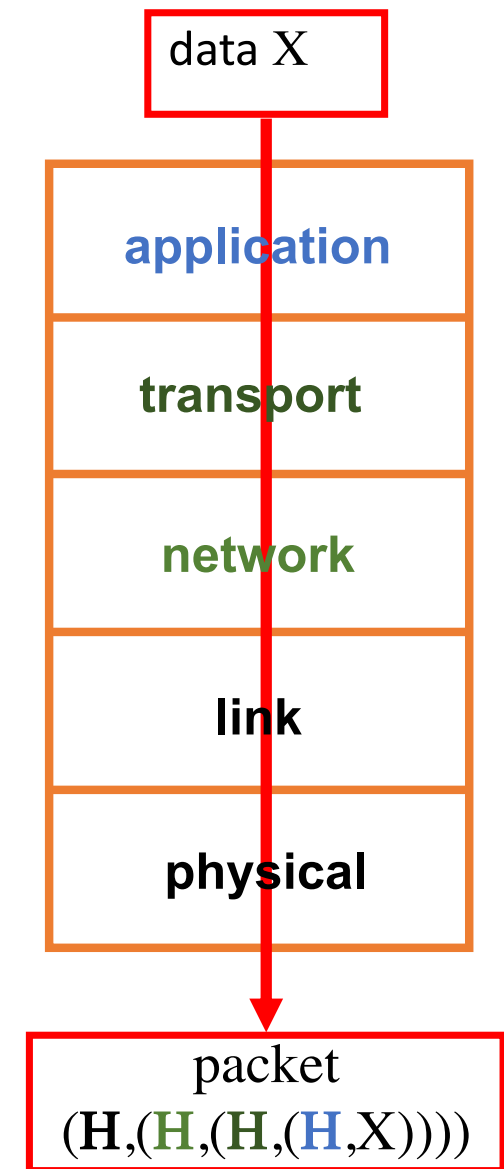
Layering in Action



- 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

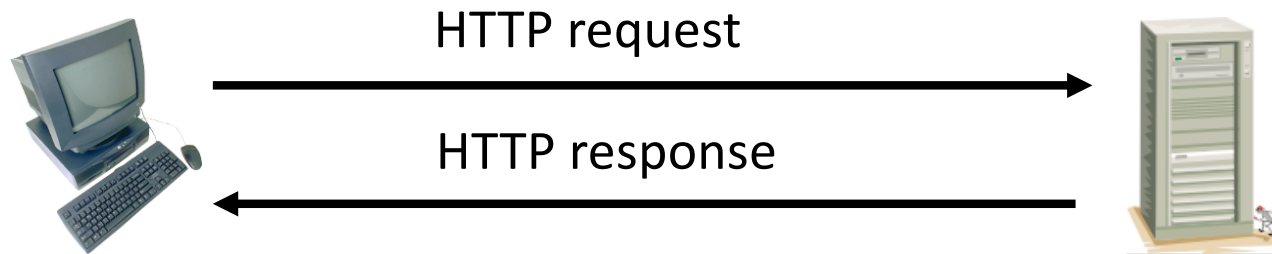
- Example: Web browsing

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

Peer-to-Peer Paradigm

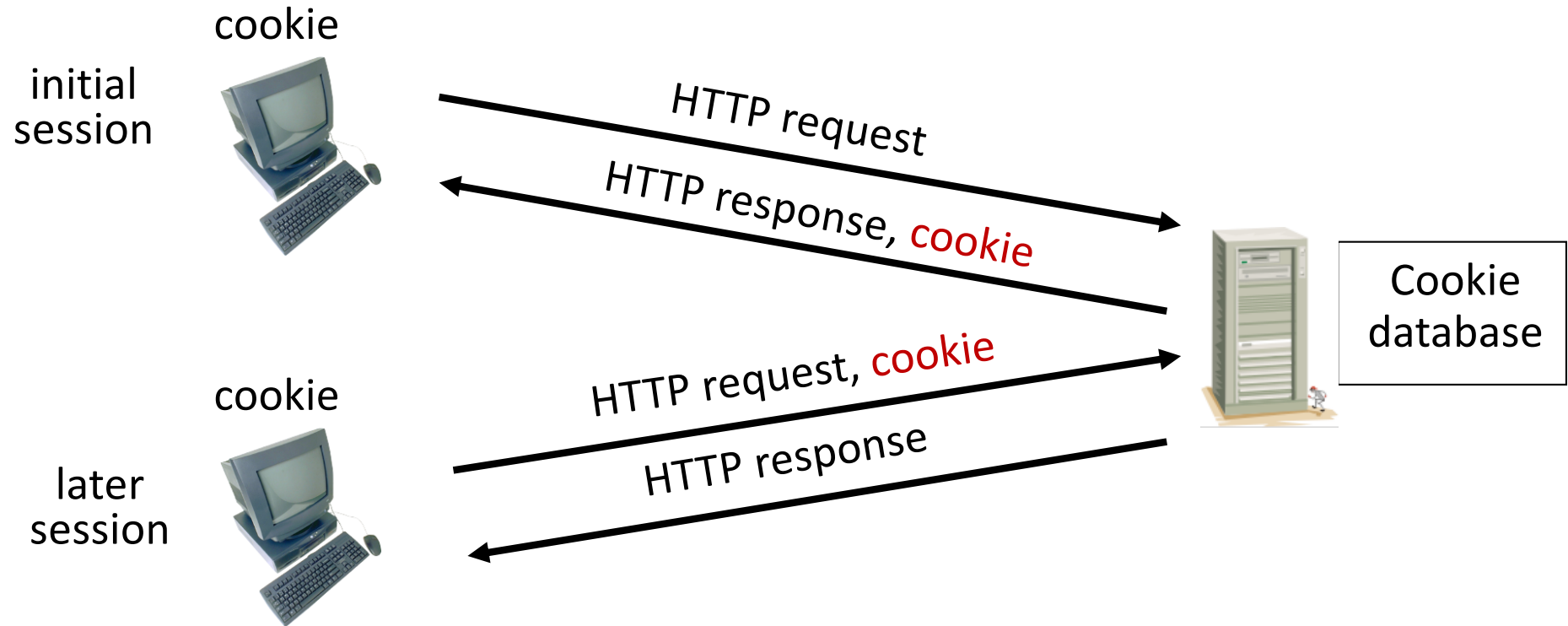
- Hosts act as clients and servers
- For example, when sharing **music**
 - You are client when requesting a file
 - You are a server when someone downloads a file from you
- In P2P, how does client find server?
 - Many different P2P models for this

HTTP Example



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

Web Cookies



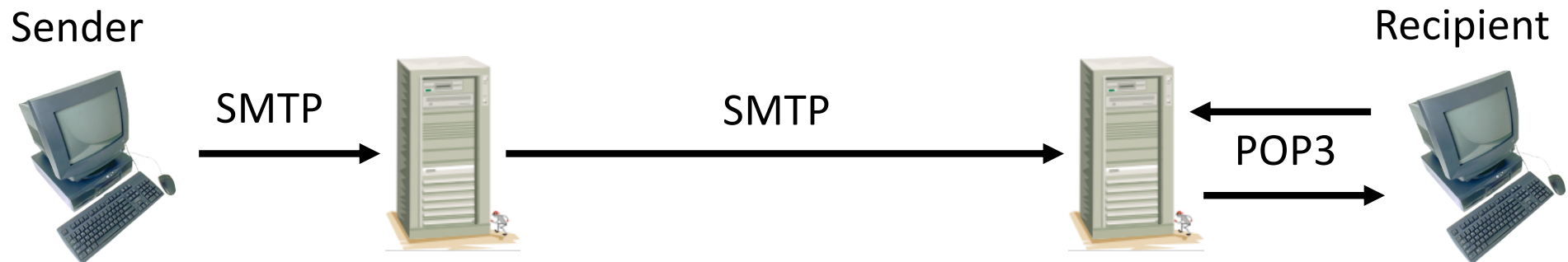
- HTTP is **stateless** — cookies used to add state
- Initially, cookie sent from server to browser
- Browser manages cookie, sends it to server
- Server uses cookie database to “remember” you

Web Cookies (Privacy Concerns)

- 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

- SMTP 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



Spoofer 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
```

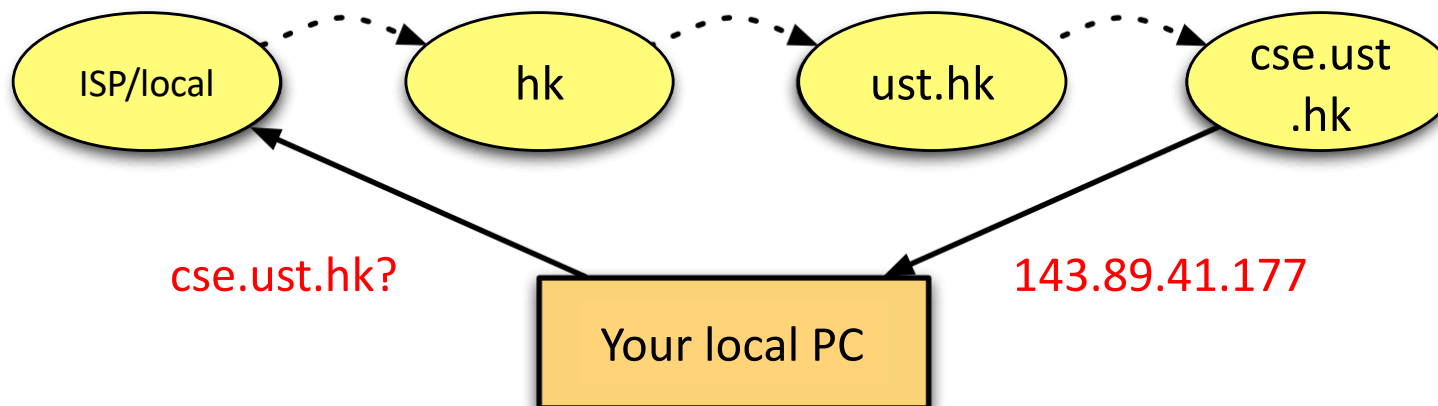
*This is just a PoC, an ideal case. SMTP protocol requires **more than that** (depends on the implementation). Don't do this to the CSE mail server and it won't work.*

Application Layer

- DNS — Domain Name Service
 - Convert human-friendly names such as www.google.com into 32-bit IP address
 - A distributed 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)

DNS - The domain name system

- DNS maps between IP address (143.89.41.177) and domain and host names (cse.ust.hk)
 - How it works: the “root” servers redirect you to the top level domains (TLD) DNS servers, which redirect you to the appropriate sub-domain, and recursively
 - Note: there are 13 “root” servers that contain the info for .org, .edu, and locality specific registries



DNS Vulnerabilities (DNS Poison)

- Normally if the server does not know a requested translation it will **ask another server**, recursively.
 - To increase performance, a server will typically remember (**cache**) these translations for a certain amount of time.
- When a DNS server has received a **false translation** and caches it for performance optimization, it is considered **poisoned**, and it supplies the false data to clients.
 - If a DNS server is **poisoned**, it may return an **incorrect IP address**, diverting traffic to another computer → often the *attacker's computer*

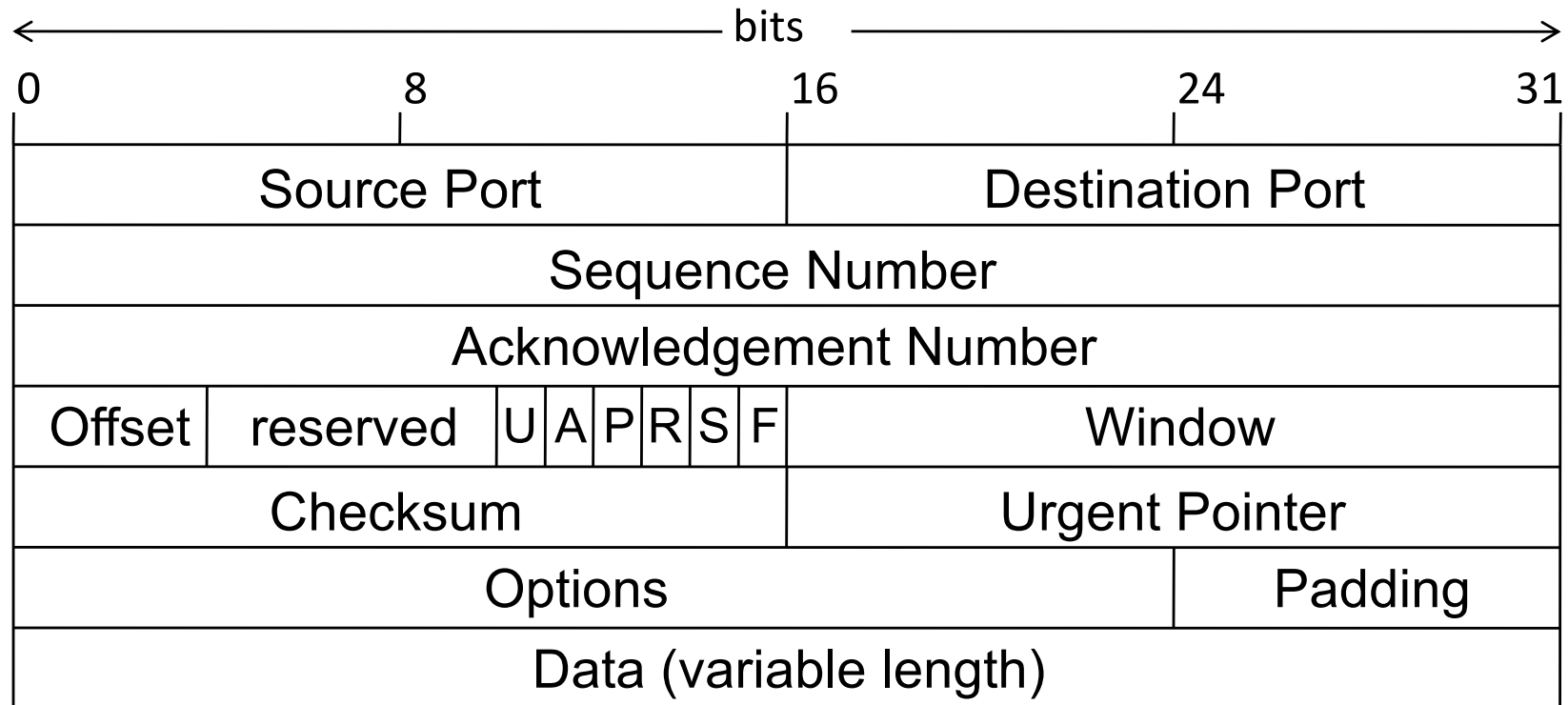
Transport Layer

- 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

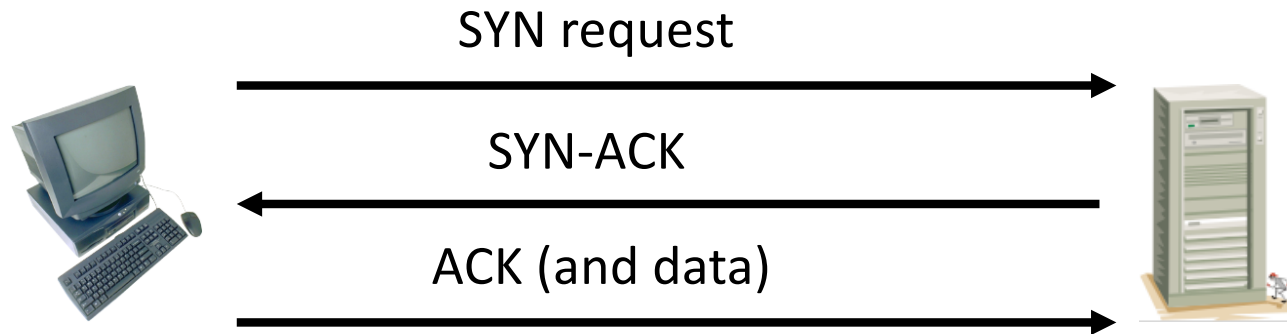
- TCP assures that packets...
 - Arrive at destination
 - Are processed in order
 - Are not sent **too fast** for receiver: **flow control**
- TCP is **connection-oriented**
 - TCP contacts server **before** sending data
 - Orderly setup and take down of “connection”
 - But no true connection, only **logical “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

Denial of Service Attack (**SYN flooding**)

- 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 running out of memory and crashing
 - Therefore, server can’t respond to legitimate connections
- This occurs because TCP is ***stateful***

Network Layer (Internet 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 bits
- Every host has an IP address
- Big problem — Not enough IP addresses!
 - Lots of tricks used to extend address space
- 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

- Each host has a 32 bit IP address
- But, 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



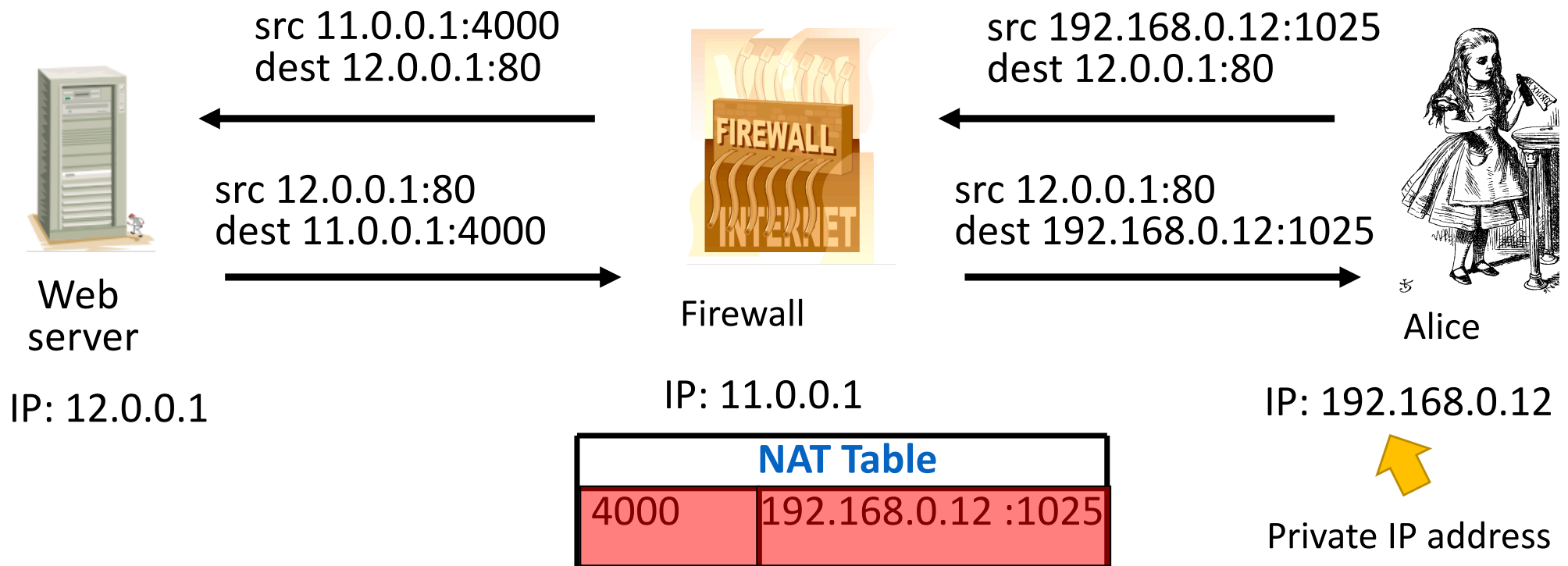
5

Alice

IP: 11.0.0.1

Port: 1025

NAT Example



NAT Security

- Advantage(s)?
 - Extends IP address space
 - One (or a few) IP address(es) can be shared by many users
 - (implicitly) make the **original source addresses hidden**.
- Disadvantage(s)?
 - Denial-of-service attack is feasible.

Ping Attack (Ping of Death/Ping flood)

- Work by abusing the **ping** command
 - Ping operates by sending **ICMP** echo request packets to the **target host** and waiting for an **ICMP** echo reply
- A **ping flood** is a simple **DoS attack** where **many attackers** conspire to overwhelm victim with ICMP packets
 - But it's possible that before you PoD a host, you saturate your own bandwidth...

```
Microsoft Windows [Version 6.0.6000]
Copyright (c) 2006 Microsoft Corporation. All rights reserved.

C:\Users\Z>ping 127.0.0.1 -n 5 -l 65500

Pinging 127.0.0.1 with 65500 bytes of data:

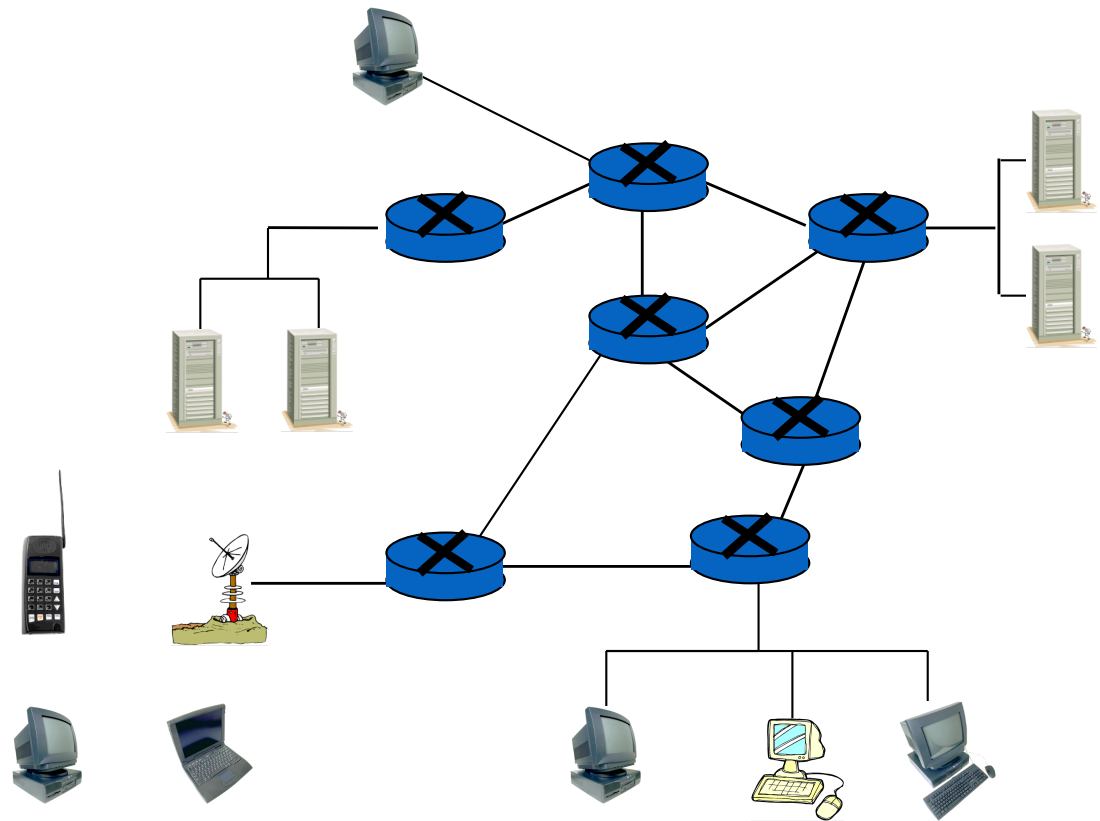
Reply from 127.0.0.1: bytes=65500 time<1ms TTL=128
Reply from 127.0.0.1: bytes=65500 time<1ms TTL=128
Reply from 127.0.0.1: bytes=65500 time<1ms TTL=128
Reply from 127.0.0.1: bytes=65500 time<1ms TTL=128
Reply from 127.0.0.1: bytes=65500 time<1ms TTL=128

Ping statistics for 127.0.0.1:
    Packets: Sent = 5, Received = 5, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\Users\Z>_
```

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

Link Layer Addressing

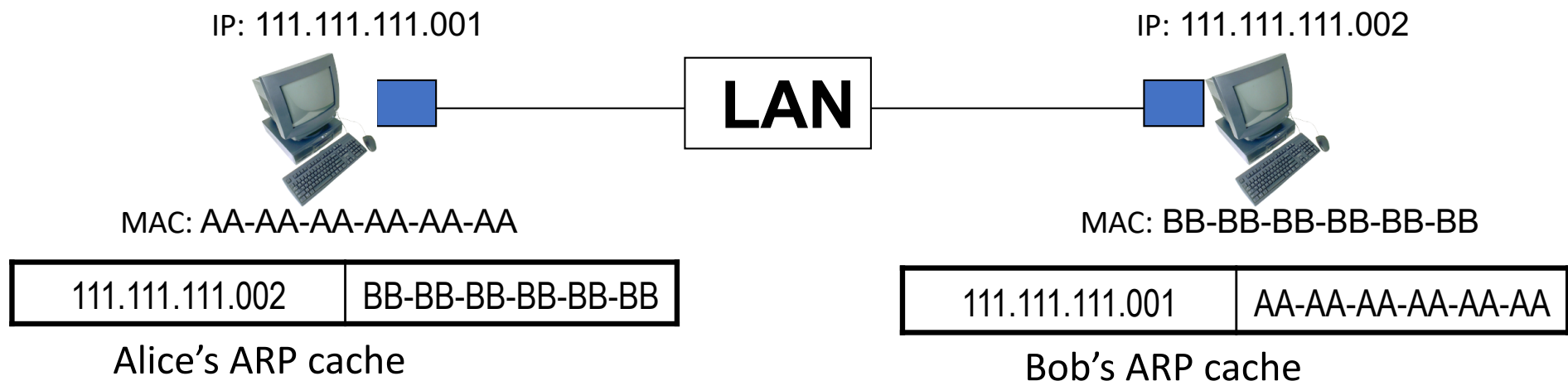
- IP addresses live at network layer
- Link layer also needs addresses
 - **MAC address** (LAN address, physical address)
- MAC address
 - 48 bits, globally unique
 - Used to forward packets over one link
- Analogy...
 - IP address is like your **home address**
 - MAC address is like a **social security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

ARP

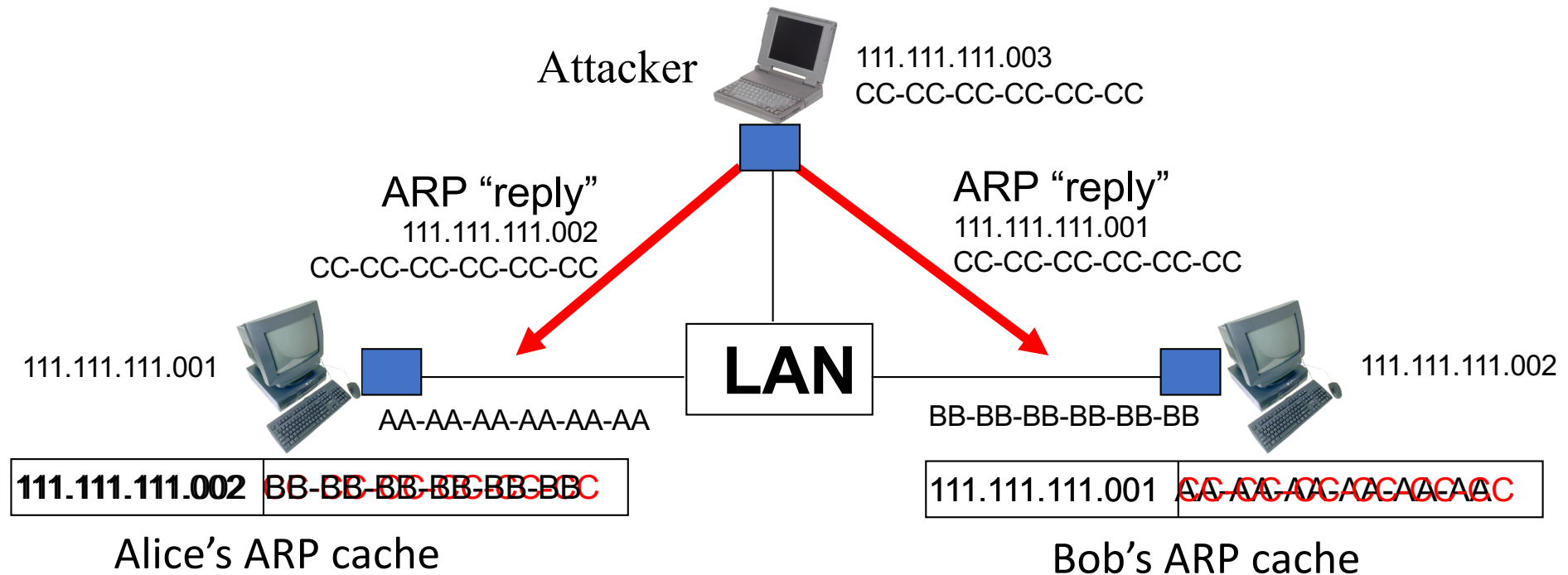
- Address Resolution Protocol (ARP)
- Used by link layer — given IP address, find corresponding MAC address
- Each host has ARP table, or **ARP cache**
 - Generated automatically
 - Entries expire after some time (about 20 min)
 - ARP used to find ARP table entries

ARP

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



ARP Cache Poisoning



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