Networks II: ARP, IP, TCP, UDP

CS 1660: Introduction to Computer Systems Security

Internet Protocol (IP) Goals

- Addressing: Provide a unique identifier to every host on the Internet
- Routing: Unified abstraction to route between any two hosts, regardless of the type of networks involved (Ethernet, Wifi, Cellular, ...)

The Internet = > A network of networks!

IP Addressing

128.148.16.7

IP Version 4: Each address is a 32-bit number:

128.148.16.7

10000000 10010100 00010000 00000111

32 bits => 2^{32} possible addresses…
problem?

<u>Notation</u>

- Write each byte (“octet”) as a decimal number 0-255
- Called “dotted decimal” or ”dotted quad” notation

IP Addressing

IP Version 4: Each address is a 32-bit number:

128.148.16.7

10000000 10010100 00010000 00000111

<u>Notation</u>

- Write each byte ("octet") as a decimal number 0-255
- Called "dotted decimal" or "dotted quad" notation

IP Addressing

128.148.16.7

IP Version 4: Each address is a 32-bit number:

128.148.16.7

10000000 10010100 00010000 00000111

32 bits => 2^{32} possible addresses…
problem?

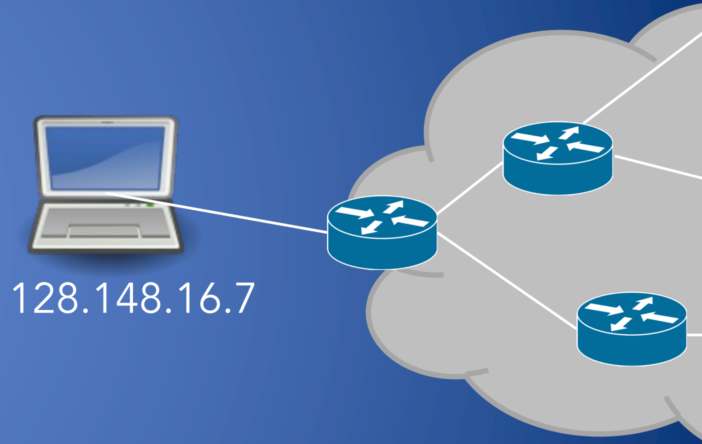
<u>Notation</u>

- Write each byte ("octet") as a decimal number 0-255
- Called "dotted decimal" or "dotted quad" notation

IP Addressing

An IP address identifies…

- *Who* a host is: A unique number
- *Where* it is on the Internet



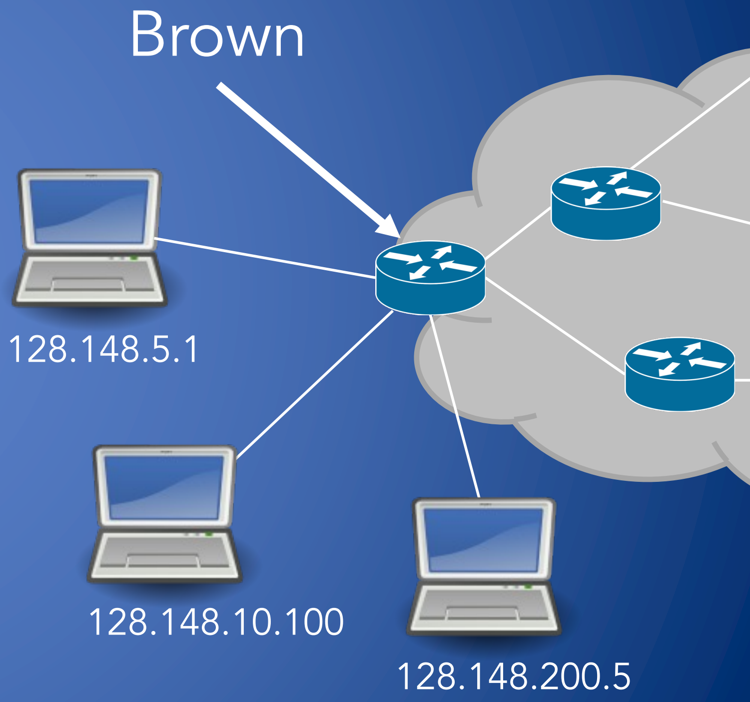
- Networks are allocated ranges of IPs by global authority (ICANN)
  - Further subdivided by regions, ISPs, …
  - US-biased, especially in early internet
- Some IPs have special uses (eg. 127.0.0.1)

eg. Brown owns 128.148.xxx.xxx, 138.16.xxx.xxx

*ICANN (Internet Corporation for Assigned Names and Numbers)

IP Addressing

A network can designate IP addresses for its own hosts within its address range

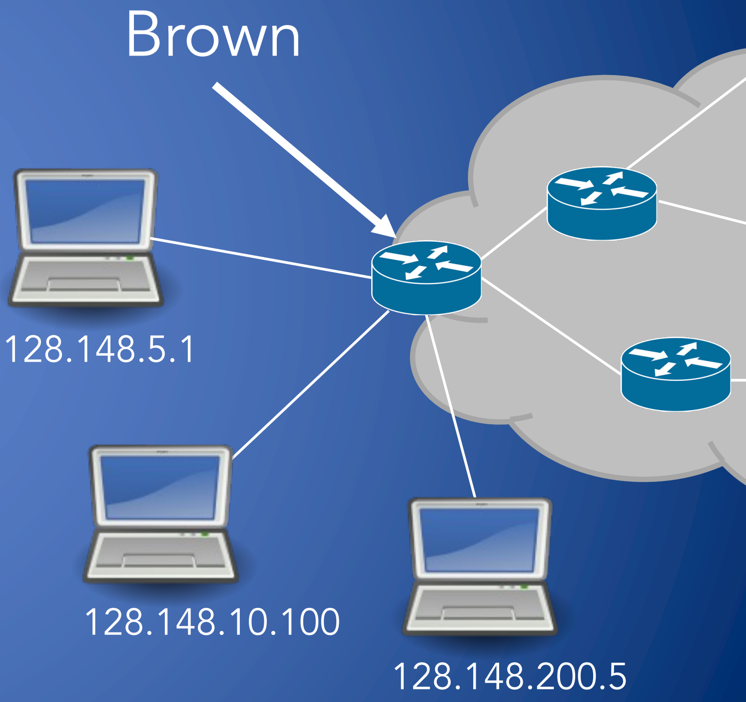


IP Addressing

A network can designate IP addresses for its own hosts within its address range

How? Every address has two parts:

- Network part: identifies the network (eg. "Brown") to the Internet
- Host part: identifies individual hosts within Brown

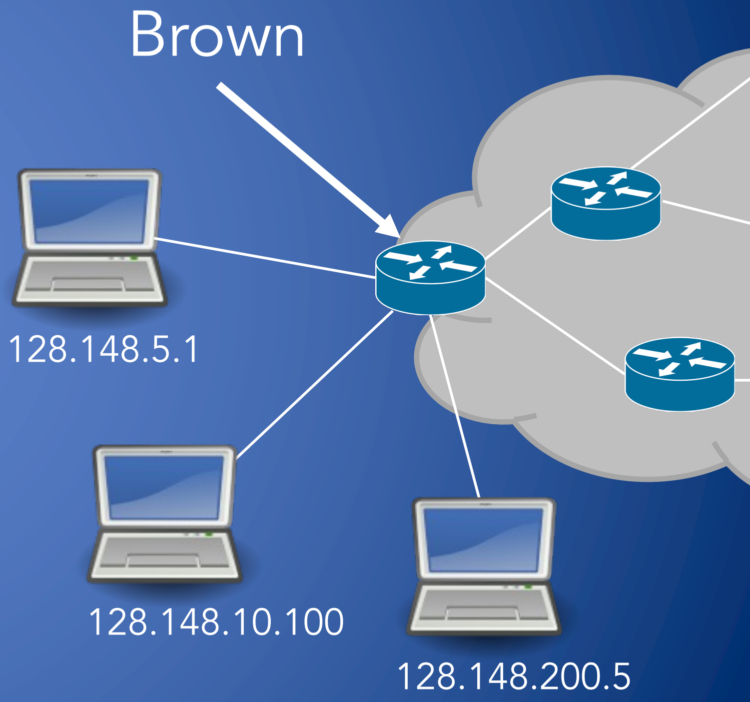


IP Addressing

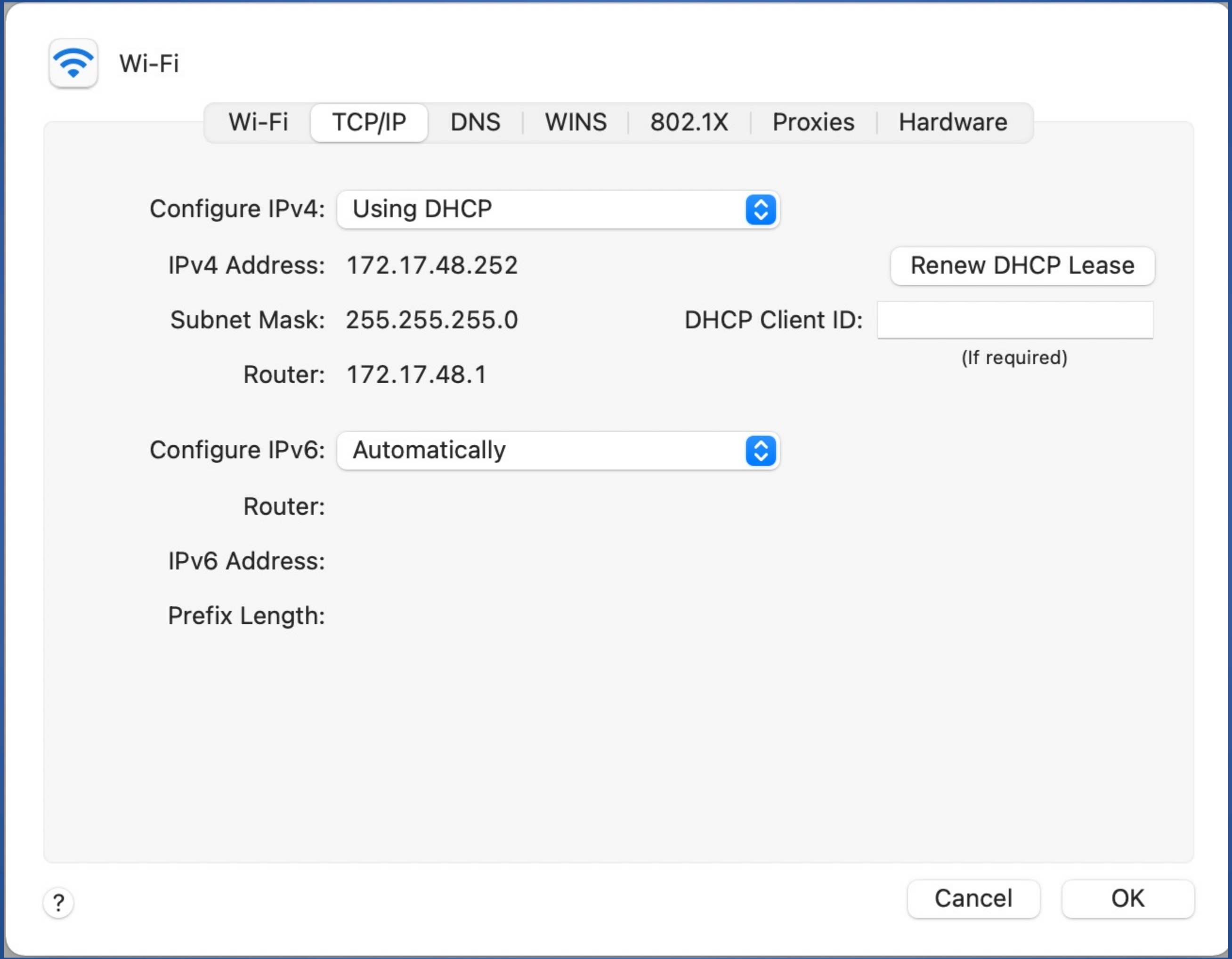
A network can designate IP addresses for its own hosts within its address range

How? Every address has two parts:

- Network part: identifies the network (eg. "Brown") to the Internet
- Host part: identifies individual hosts within Brown



But how big should the network be?


Wi-Fi
Wi-Fi
TCP/IP
DNS
WINS
802.1X
Proxies
Hardware
Configure IPv4:
Using DHCP
IPv4 Address:
172.17.48.252
Renew DHCP Lease
Subnet Mask:
255.255.255.0
DHCP Client ID:
(If required)
Router:
172.17.48.1
Configure IPv6:
Automatically
Router:
IPv6 Address:
Prefix Length:
?
Cancel
OK

Components of an IP

172.17.48.252

IPv4 Address: 172.17.48.252

Subnet Mask: 255.255.255.0

Router: 172.17.48.1

`Addr: 172.17.48.252` `10101100 00010001 00110000 11111100`

Key point: networks can be of different sizes!

Components of an IP

IPv4 Address:	172.17.48.252
Subnet Mask:	255.255.255.0
Router:	172.17.48.1

172.17.48.252

```
Addr: 172.17.48.252    10101100 00010001 00110000 11111100

Mask: 255.255.255.0    11111111 11111111 11111111 00000000
```

Key point: networks can be of different sizes!
=>The "subnet mask" defines what part of is the network part

Components of an IP

172.17.48.252

IPv4 Address:	172.17.48.252
Subnet Mask:	255.255.255.0
Router:	172.17.48.1

```
Addr: 172.17.48.252    10101100 00010001 00110000 11111100

Mask: 255.255.255.0    11111111 11111111 11111111 00000000
```

24 bits

8 bits = 2^8 = 256 hosts

Key point: networks can be of different sizes!
=>The "subnet mask" defines what part of is the network part

Written in "CIDR notation" or "prefix notation" as /(number of 1 bits in mask),
eg. 172.17.48.0/24

Common Prefix Sizes

Prefix	IPs	Number of hosts	Note
1.2.3.4.0/24	1.2.3.*	2^8 = 256	Common for local networks (LANs) Old term: "Class C"
1.2.0.0/16	1.2.*.*	2^16 = 65536	Old term: "Class B" Large (or older) organizations
1.0.0.0/8	1.*.*.*	2^24 = ~16M	Old term: "Class A"
1.2.3.100/30	1.2.3.1-1.2.3.3	4	A smaller prefix

Special/private IP ranges

Prefix	Note
127.0.0.0/8	Localhost (for networks on same system), usually 127.0.0.1
192.168.0.0/16	Private: often used for home networks
10.0.0.0.8	Private: often used for larger organizations (eg. Brown)
172.16.0.0/12	Private: larger space for organizations, systems (eg. Docker)

- Used for LANs, private networks
- More on this later

IP Address Space and ICANN

- Hosts on the internet must have unique IP addresses
- Internet Corporation for Assigned Names and Numbers
  - International nonprofit organization
  - Incorporated in the US
  - Allocates IP address space
  - Manages top-level domains
- Historical bias in favor of US corporations and nonprofit organizations

003/8	May 94	General Electric
009/8	Aug 92	IBM
012/8	Jun 95	AT&T Bell Labs
013/8	Sep 91	Xerox Corporation
015/8	Jul 94	Hewlett-Packard
017/8	Jul 92	Apple Computer
018/8	Jan 94	MIT
019/8	May 95	Ford Motor
040/8	Jun 94	Eli Lily
043/8	Jan 91	Japan Inet
044/8	Jul 92	Amateur Radio Digital
047/8	Jan 91	Bell-Northern Res.
048/8	May 95	Prudential Securities
054/8	Mar 92	Merck
055/8	Apr 95	Boeing
056/8	Jun 94	U.S. Postal Service

Viewing Network Configuration

MAC address

IPv4 address

```
deemer@ceres ~ % ip addr
2: enp7s0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu default ...
    link/ether c8:f7:50:55:9e:29 brd ff:ff:ff:ff:ff:ff
    inet 172.17.48.25/24 scope global enp0s31f6
       valid_lft forever preferred_lft forever
    inet6 fe80::caf7:50ff:fe55:9e29/64 scope link
       valid_lft forever preferred_lft forever
```

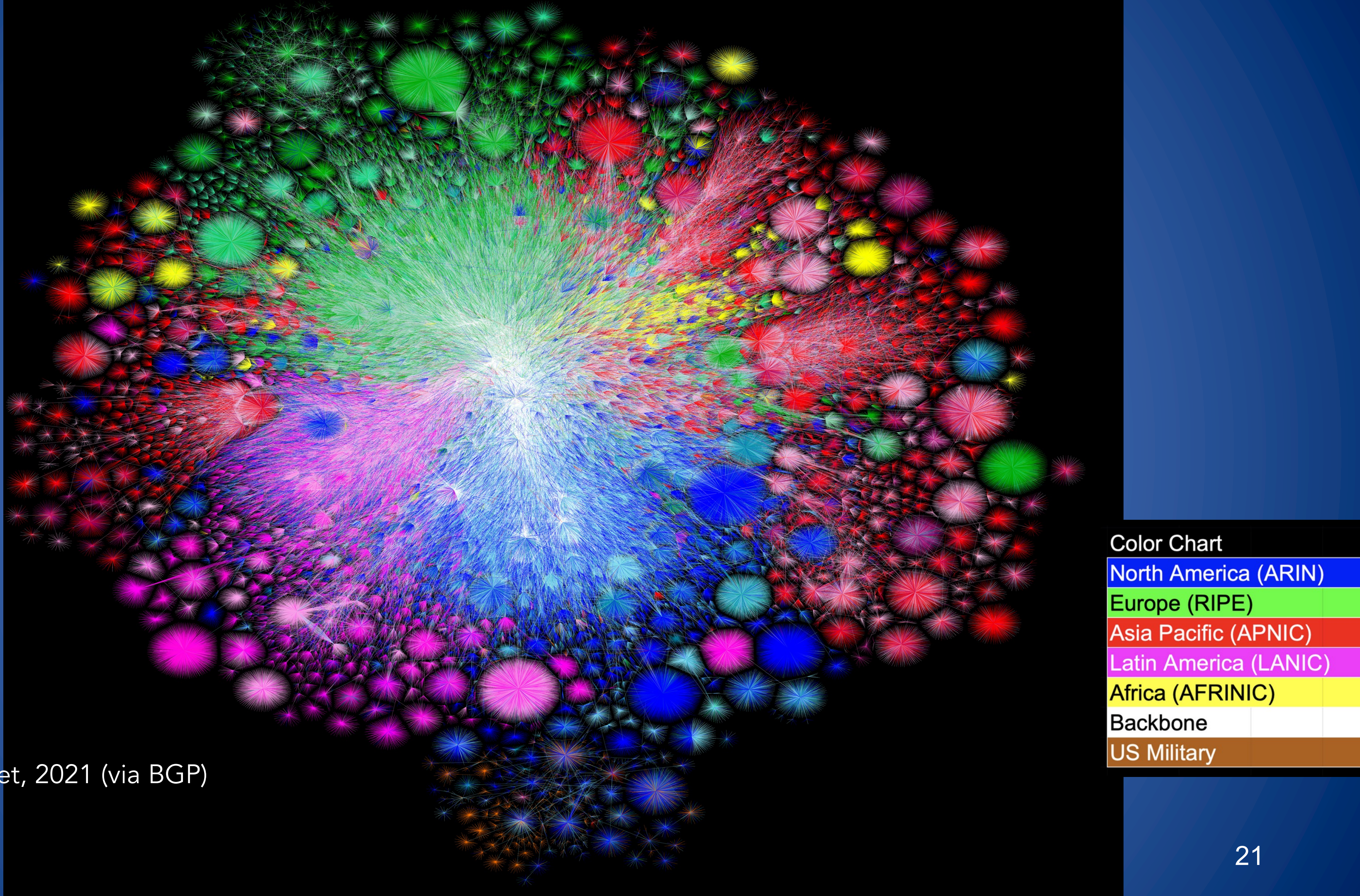
```
deemer@ceres ~ % ip route
127.0.0.0/8 via 127.0.0.1 dev lo
172.17.48.0/24 dev enp7s0  proto kernel
default via 172.17.48.1 dev eth0 src 172.17.44.22
```

Gateway IP address

Brown's IP Space

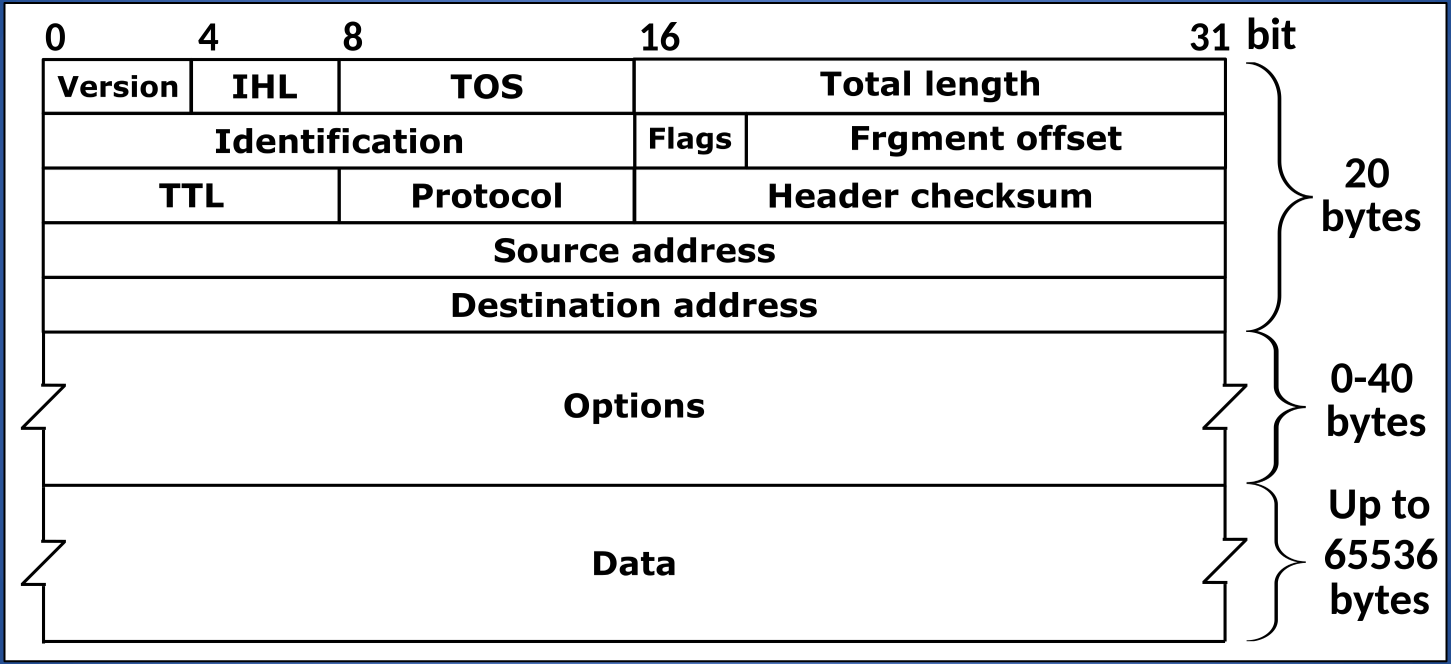
- Brown separates the network connecting dorms and the network connecting offices and academic buildings
  - Class B network 138.16.0.0/16 (64K addresses)
  - Class B network 128.148.0.0/16 (64K addresses)
- CS department
  - Several class C (/24) networks, each with 254 addresses
  - Tstaff supported machines: 128.148.31.0/24, 128.148.33.0/24, 128.148.38.0/24
  - Unsupported machines:128.148.36.0/24

How do we move packets *between* networks?



Map of the Internet, 2021 (via BGP)
OPTE project

The IPv4 Header

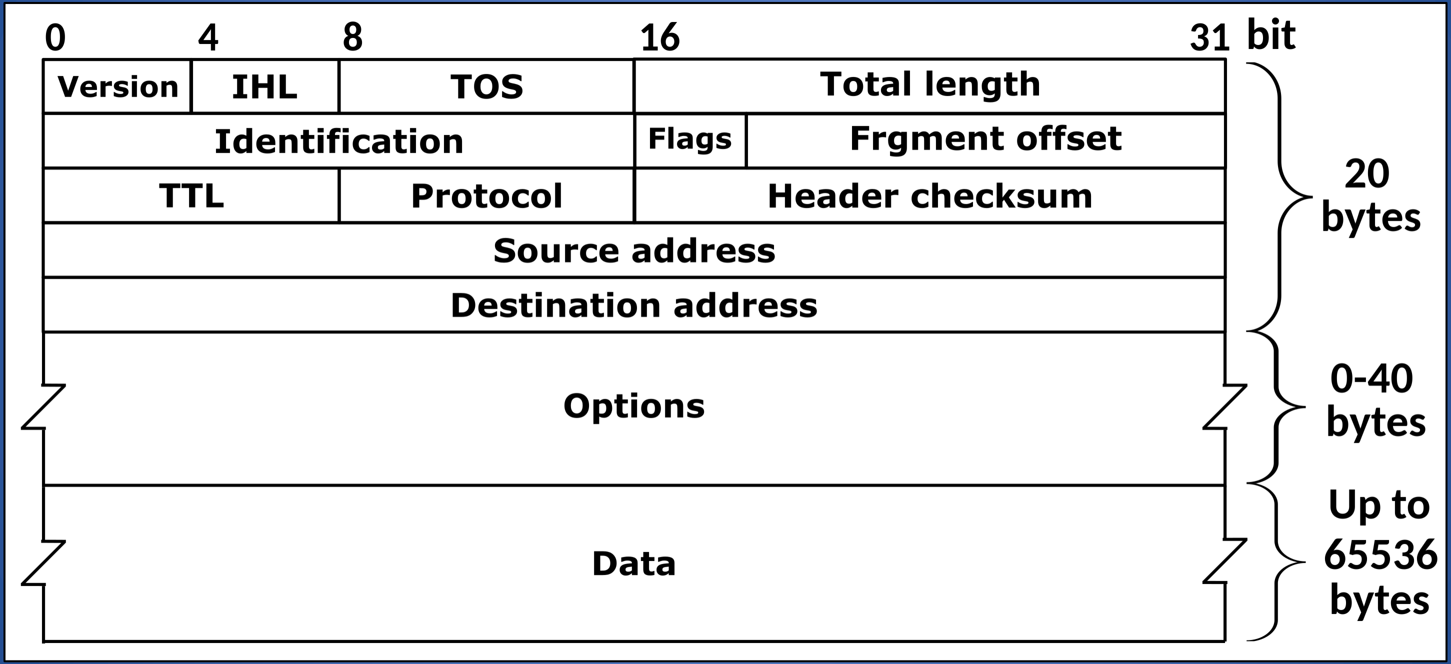


Defined by RFC 791
RFC (Request for Comment): defines network standard

IP Routing

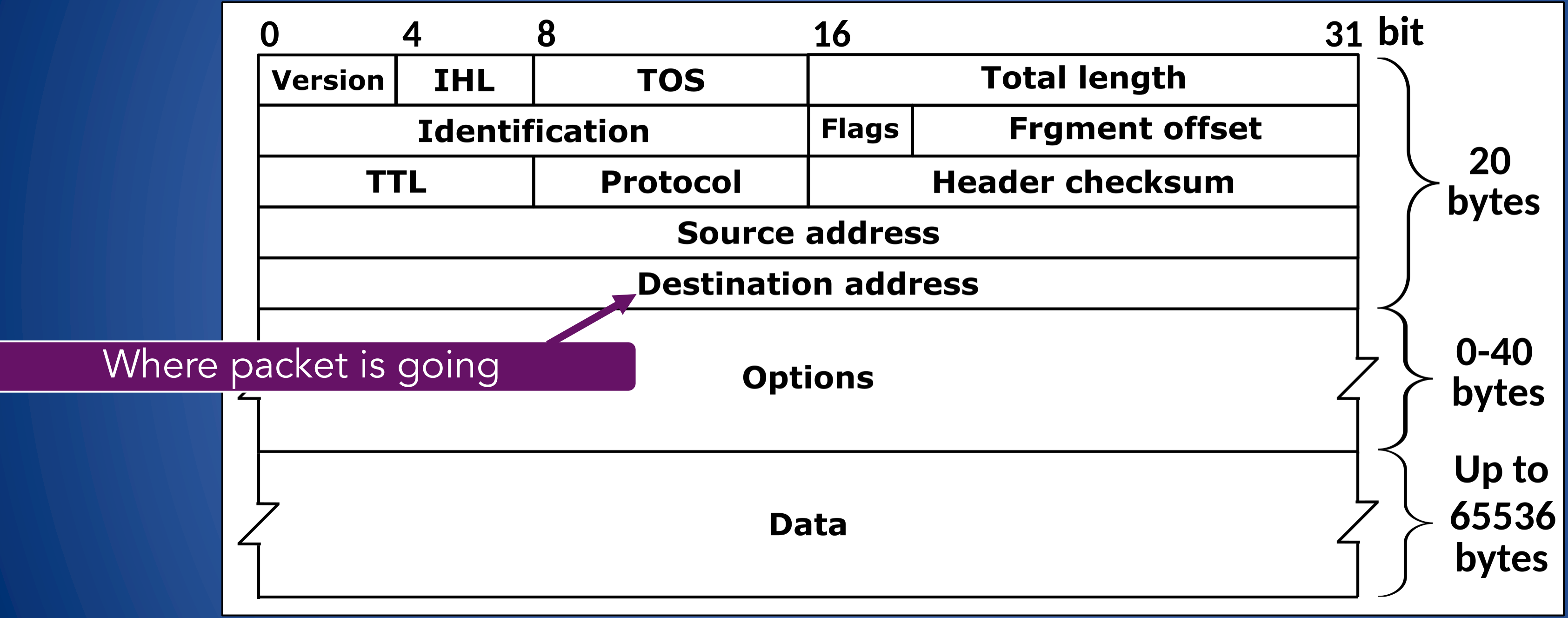
- A router connects two or more networks
  - Maintains tables to forward packets to the appropriate network
  - Forwarding decisions based solely on the destination address
  - Hosts (regular systems) can be routers too!
- Routing table
  - Maps ranges of addresses to LANs or other gateway routers

The IPv4 Header

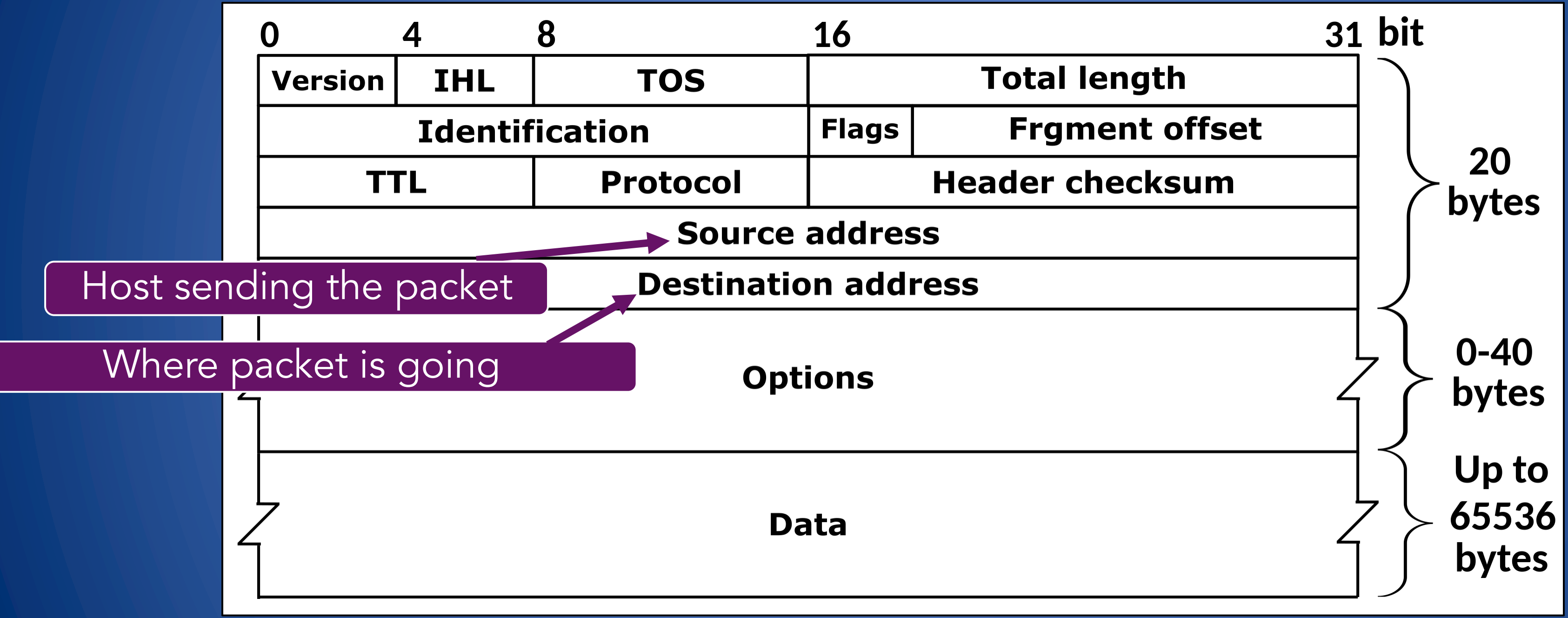


This header is at the start of every packet sent on the Internet

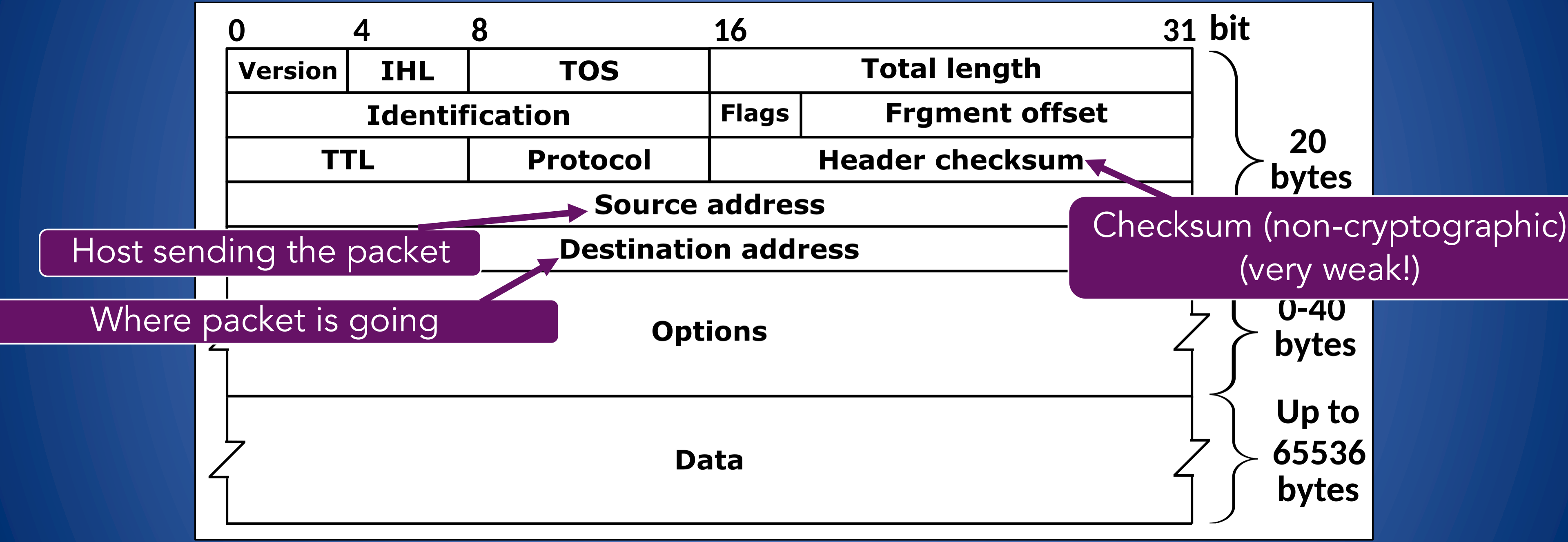
The IPv4 Header



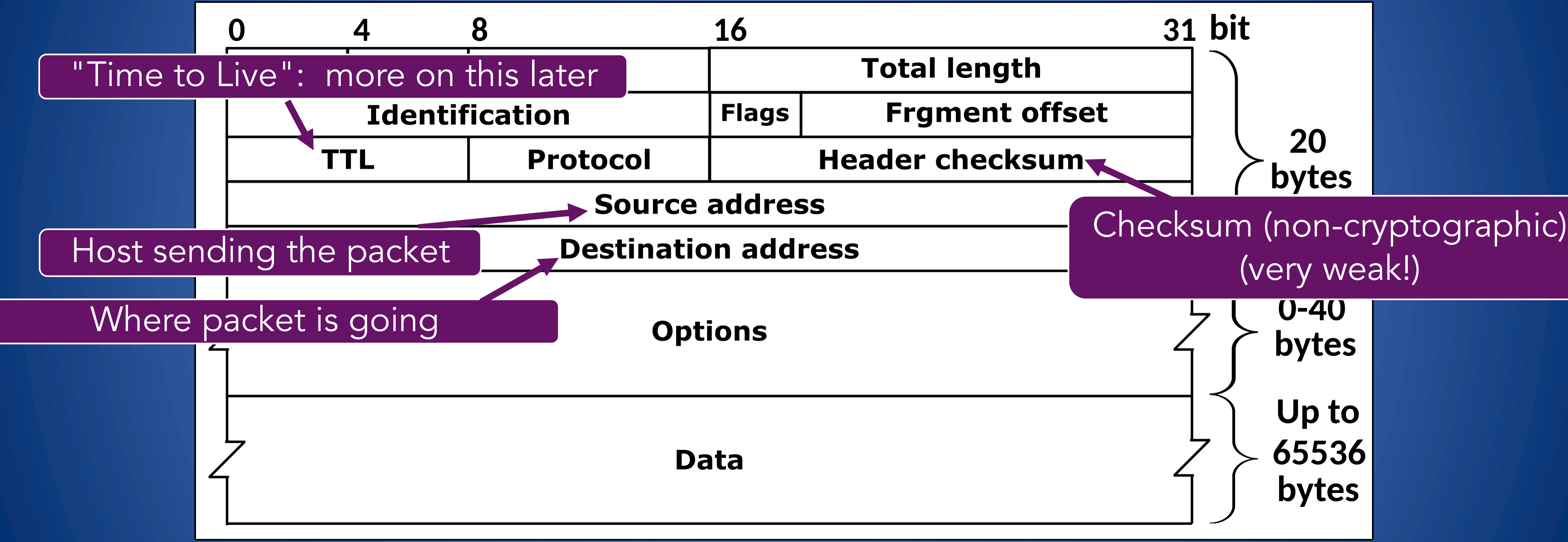
The IPv4 Header



The IPv4 Header



The IPv4 Header

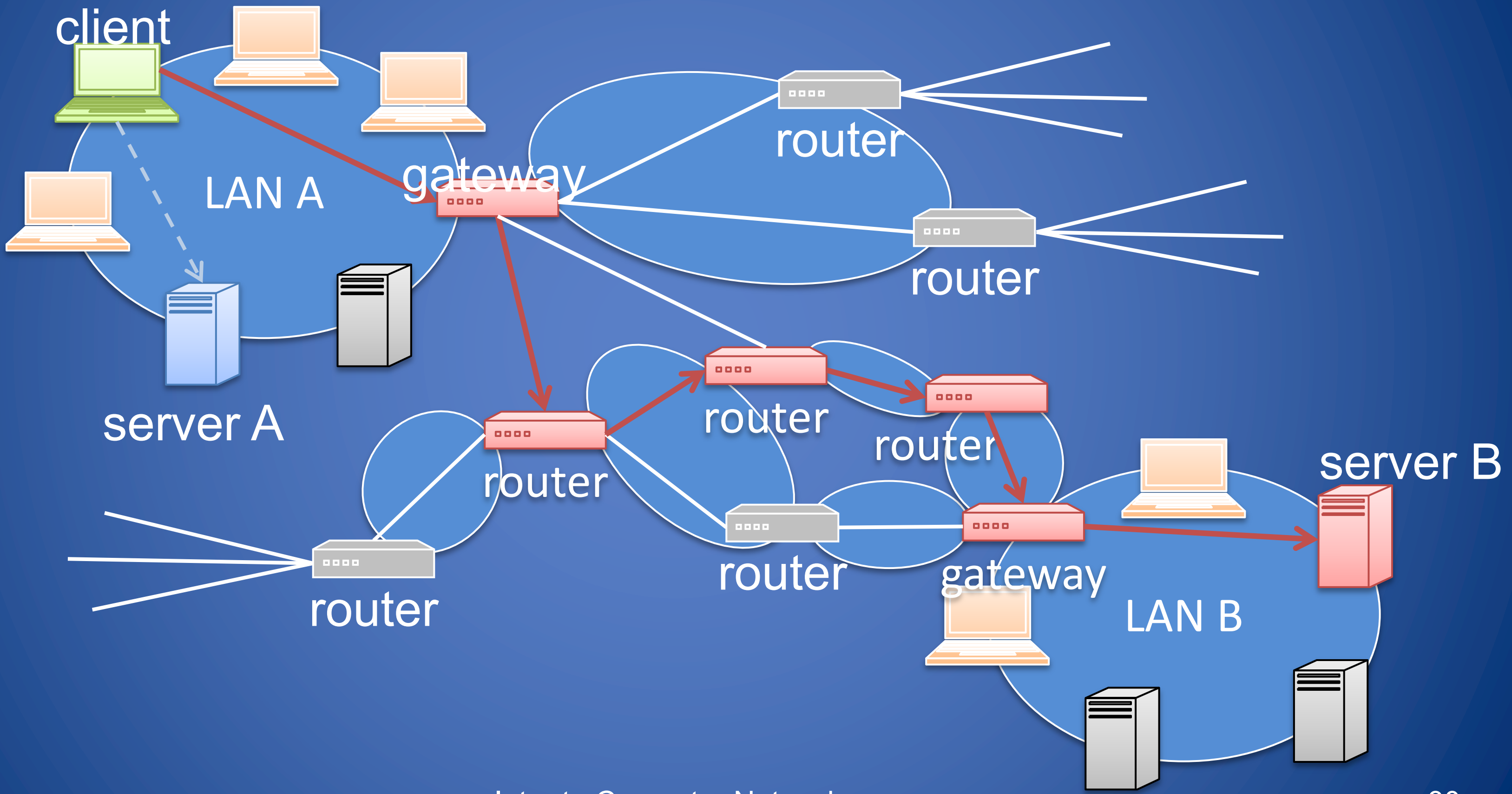


Example routing table

```
deemer@ceres ~ % ip route
127.0.0.0/8 via 127.0.0.1 dev lo
172.17.48.0/24 dev enp7s0  proto kernel
default via 172.17.48.1 dev eth0 src 172.17.44.22
```

- "Default": where to send packets when they go to a network you don't know about
- Also known as "next hop"

Routing Examples



Clicker Question (2)

Which layer best describes the operation of a router?

A. Application
B. Link
C. Transport
D. Network

Clicker Question(2) - Answer

Which layer best describes the operation of a router?

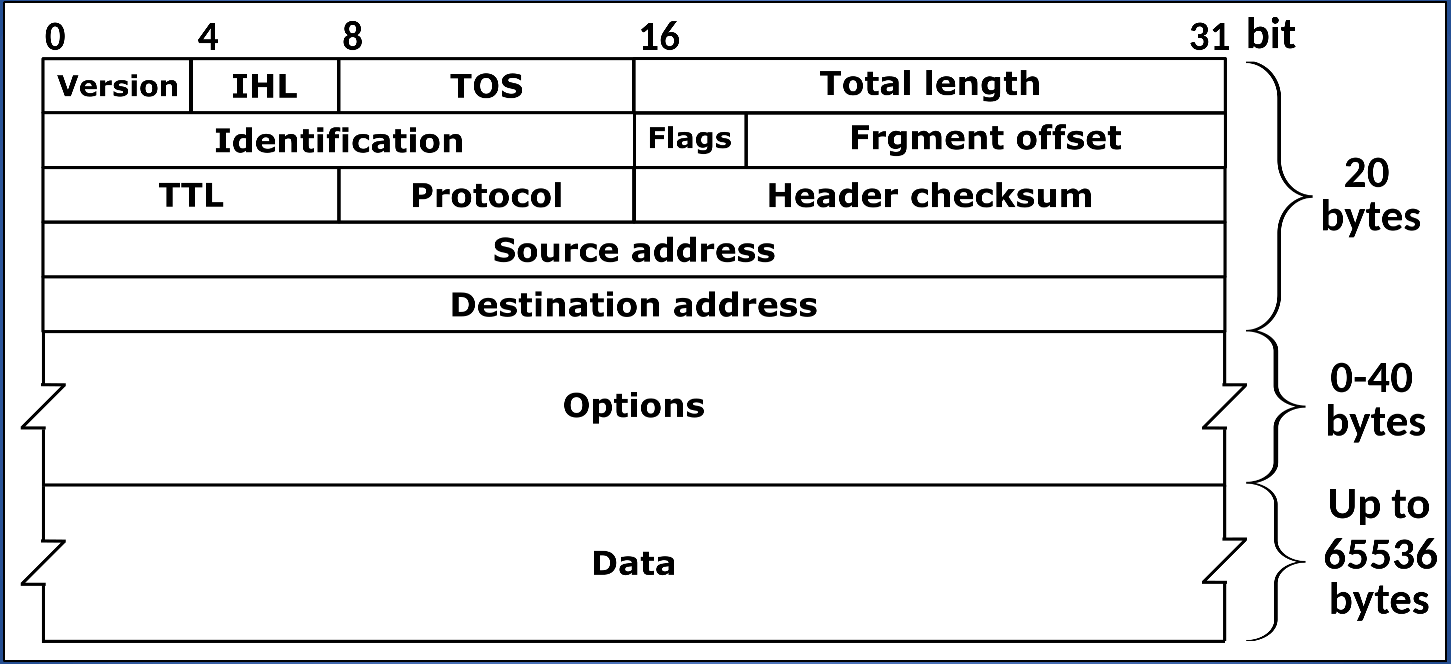
A. Application

B. Link

C. Transport

D. Network

The IPv4 Header



Defined by RFC 791
RFC (Request for Comment): defines network standard

A Simple Internet Protocol

- Internet Control Message Protocol (ICMP)
  - Used for network testing and debugging
  - Network-layer protocol: simple messages about IP forwarding/routing

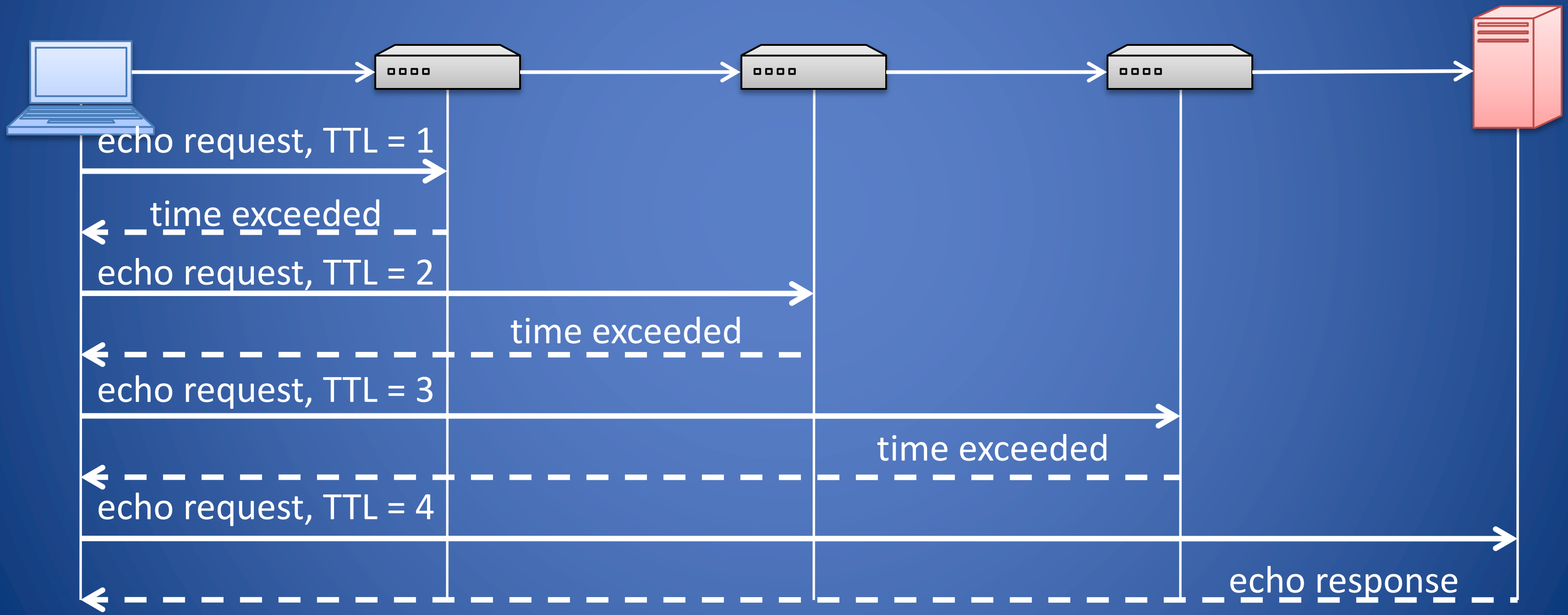
- Tools based on ICMP
  - Ping: send a message to an IP, get a response back
  - Traceroute: sends series ICMP packets with increasing TTL value to discover routes

TTL: Time to Live

- When TTL reaches 0, router may send back an error
  - "ICMP TTL exceeded" message
- If it does, we can identify a path used by a packet!

=> Traceroute takes advantage of this

Traceroute



echo request, TTL = 1
time exceeded
echo request, TTL = 2
time exceeded
echo request, TTL = 3
time exceeded
echo request, TTL = 4
echo response

Traceroute example

```
[deemer@Warsprite ~]$ traceroute -q 1 google.com
traceroute to google.com (142.251.40.174), 30 hops max, 60 byte packets
 1  router1-nac.linode.com (207.99.1.13)  0.621 ms
 2  if-0-1-0-0-0.gw1.cjj1.us.linode.com (173.255.239.26)  0.499 ms
 3  72.14.222.136 (72.14.222.136)  0.949 ms
 4  72.14.222.136 (72.14.222.136)  0.919 ms
 5  108.170.248.65 (108.170.248.65)  1.842 ms
 6  lga25s81-in-f14.1e100.net (142.251.40.174)  1.812 ms
```

Traceroute example

```
[deemer@Warsprite ~]$ traceroute -q 1 amazon.co.uk
traceroute to amazon.co.uk (178.236.7.220), 30 hops max, 60 byte packets
 1  router2-nac.linode.com (207.99.1.14)  0.577 ms
 2  if-11-1-0-1-0.gw2.cjj1.us.linode.com (173.255.239.16)  0.461 ms
 3  ix-et-2-0-2-0.tcore3.njy-newark.as6453.net (66.198.70.104)  1.025 ms
 4  be3294.ccr41.jfk02.atlas.cogentco.com (154.54.47.217)  2.938 ms
 5  be2317.ccr41.lon13.atlas.cogentco.com (154.54.30.186)  69.725 ms
 6  be2350.rcr21.b023101-0.lon13.atlas.cogentco.com (130.117.51.138)  69.947 ms
 7  a100-row.demarc.cogentco.com (149.11.173.122)  71.639 ms
 8  150.222.15.28 (150.222.15.28)  78.217 ms
 9  150.222.15.21 (150.222.15.21)  84.383 ms
10  *
11  150.222.15.4 (150.222.15.4)  74.529 ms
                                              . . .
30  178.236.14.162 (178.236.14.162)  83.659 ms
```

Practicing Ping and Traceroute

- Linux/Unix/MacOS
  - ip addr (Linux) / ifconfig (MacOS)
  - ping www.brown.edu
  - traceroute www.brown.edu
- Windows
  - ipconfig /all
  - tracert www.brown.edu

How do you get an IP address?

Obtaining Host IP Addresses - DHCP

- Networks are free to assign addresses within block to hosts
- Tedious and error-prone: e.g., laptop going from CIT to library to coffee shop
- Idea: client asks network for IP on connection

Obtaining Host IP Addresses - DHCP

- Networks are free to assign addresses within block to hosts
- Tedious and error-prone: e.g., laptop going from CIT to library to coffee shop
- Idea: client asks network for IP on connection

=> But how? How to send packets with no IP address?

Broadcast traffic

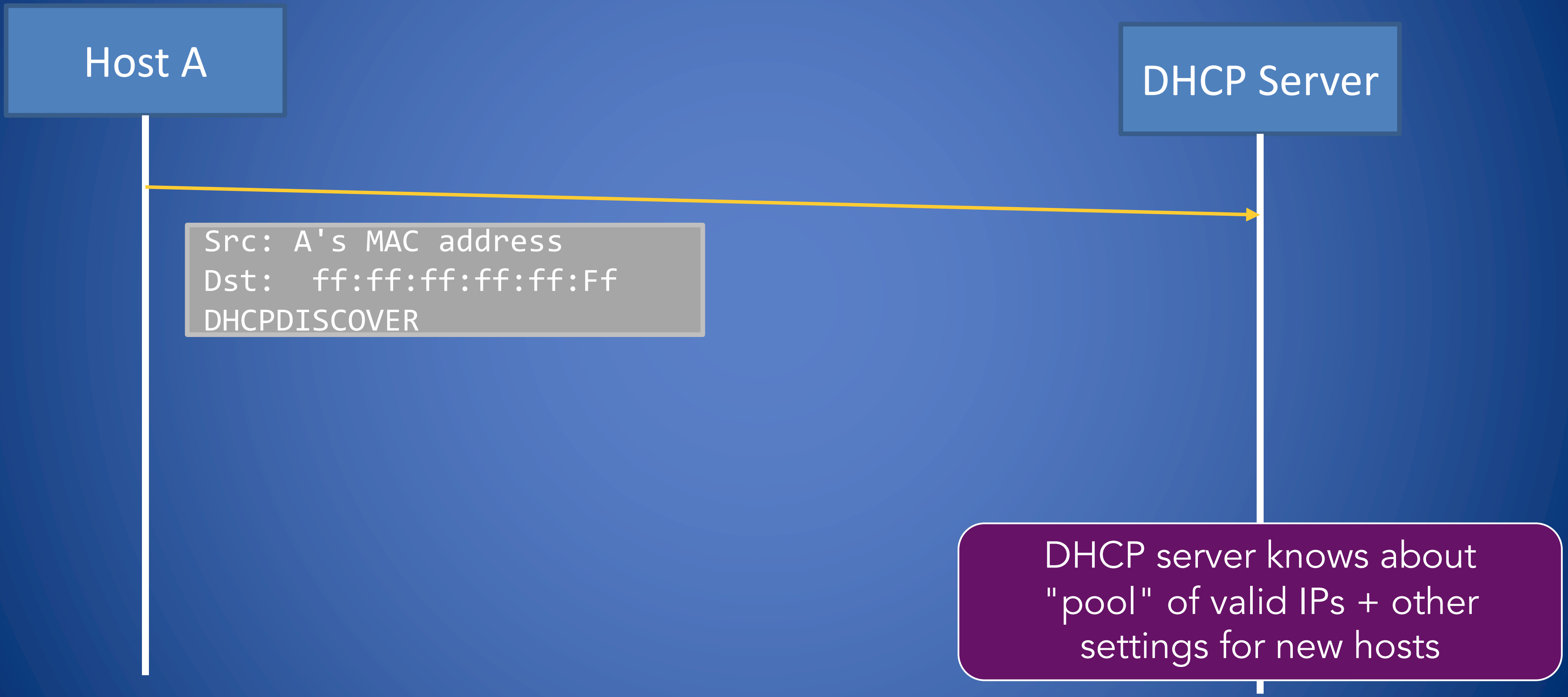
Special MAC address: ff:ff:ff:ff:ff:ff

- Forwarded to all hosts on network!
- Used for link-layer protocols, particularly for finding IP addresses (DHCP, ARP)

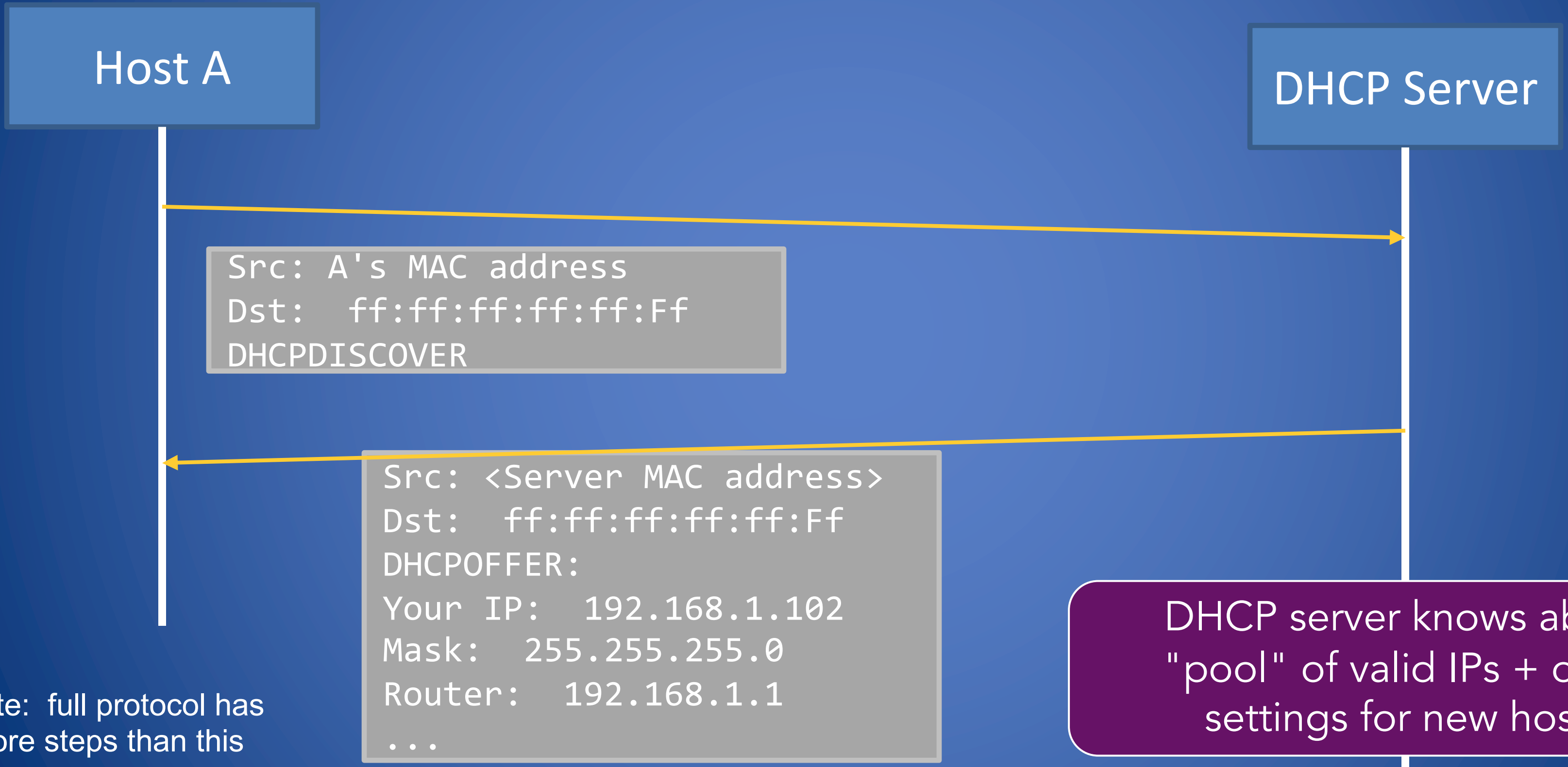
Each IP subnet also has a broadcast address, usually last IP (eg. 192.168.1.255)

Start of DHCP

Start of DHCP



Start of DHCP



DHCP server knows about "pool" of valid IPs + other settings for new hosts

Note: full protocol has more steps than this

Problems with DHCP?

- What happens if a random host decides to be a DHCP server?

Problems with DHCP?

- What happens if a random host decides to be a DHCP server?

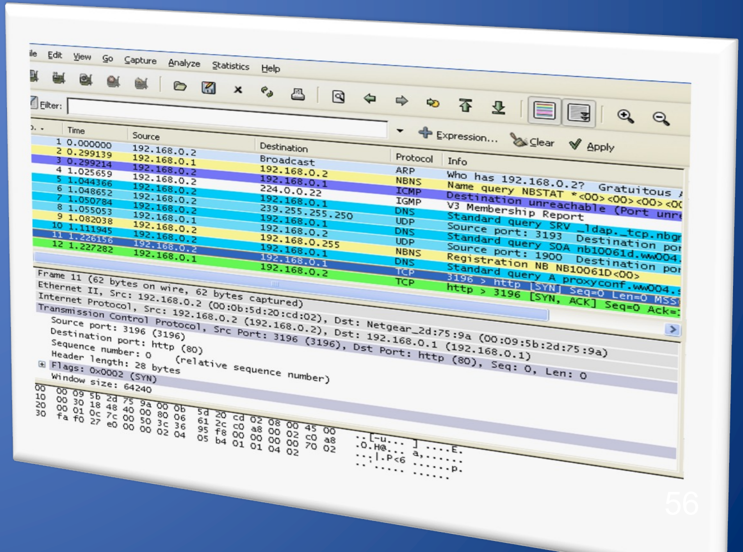
⇒Race condition! If an attacker can make an offer more quickly than the server, can assign a host's IP settings

  - Would be detected by the real DHCP server, though (why?)

Monitoring

Network Monitoring

- Understanding the traffic through a network to evaluate performance and detect anomalous conditions
- Tools collect raw network traffic, annotate packets according to their protocol and provide filters:
  - TCPDump: command line tool displaying textual information
  - Wireshark: graphical tool that colors packets



Network Traffic Analysis: TCPDump

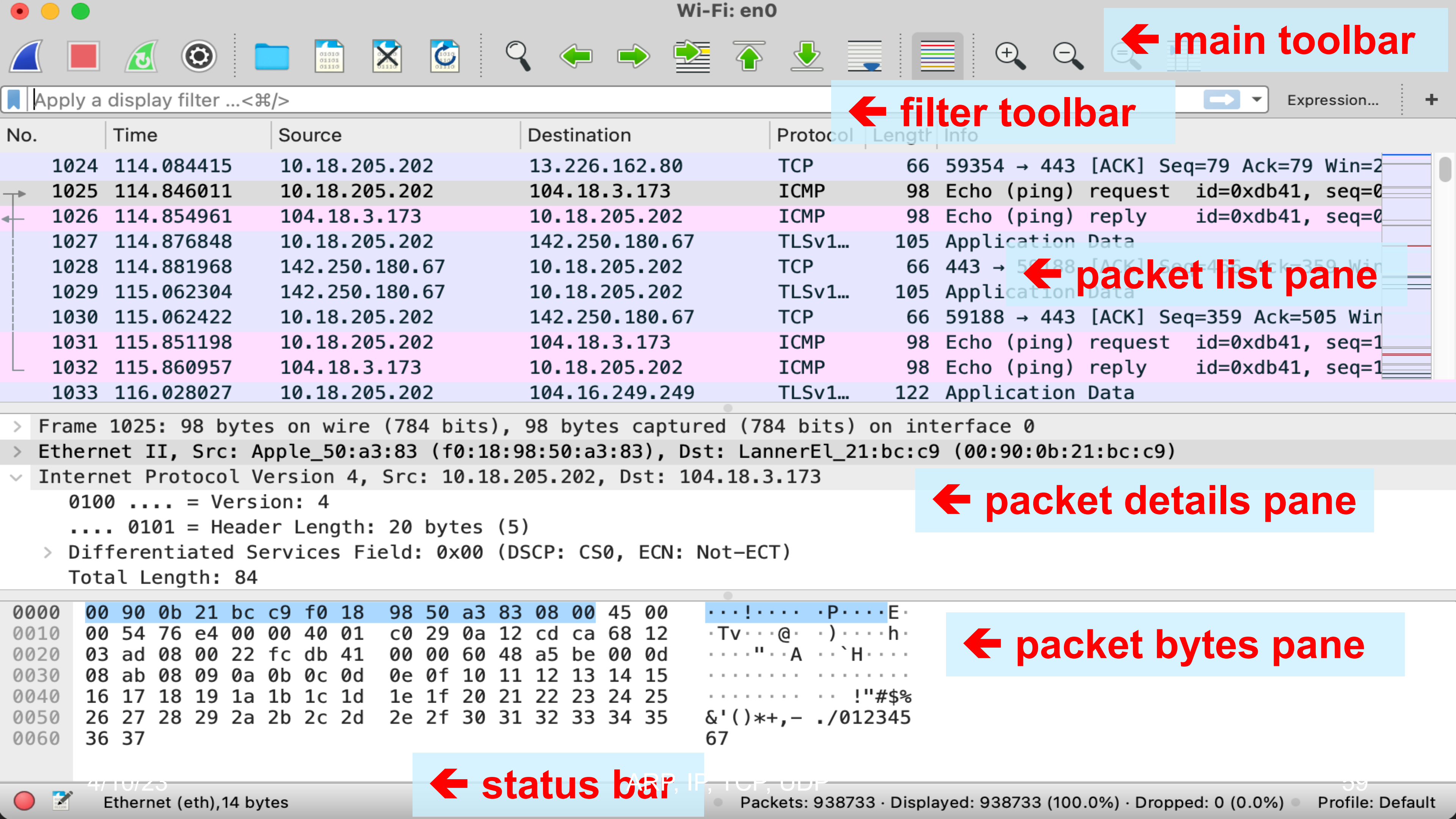
```
bernardo — -zsh — 85×34
bernardo@Bibi-MacBook-Pro-4394 ~ % sudo tcpdump
tcpdump: data link type PKTAP
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on pktap, link-type PKTAP (Apple DLT_PKTAP), capture size 262144 bytes
12:33:59.277326 IP 162.125.80.17.https > 10.18.205.202.59855: Flags [P.], seq 3443913
180:3443913211, ack 1152850195, win 1339, options [nop,nop,TS val 2762333655 ecr 1950
44380], length 31
12:33:59.277431 IP 162.125.80.17.https > 10.18.205.202.59855: Flags [F.], seq 31, ack
 1, win 1339, options [nop,nop,TS val 2762333731 ecr 195044380], length 0
12:33:59.277440 IP 10.18.205.202.59855 > 162.125.80.17.https: Flags [R], seq 11528501
95, win 0, length 0
12:33:59.278341 IP 10.18.205.202.62016 > parigi2.istat.it.domain: 34305+ PTR? 17.80.1
25.162.in-addr.arpa. (44)
12:33:59.430641 IP parigi2.istat.it.domain > 10.18.205.202.62016: 34305 NXDomain 0/1/
0 (94)
12:33:59.640328 IP 10.18.205.202.59857 > pr1stfldb.istat.it.https: Flags [P.], seq 22
64808671:2264808948, ack 3074131225, win 4096, length 277
12:33:59.645411 IP pr1stfldb.istat.it.https > 10.18.205.202.59857: Flags [.], ack 277
, win 511, length 0
12:33:59.646925 IP pr1stfldb.istat.it.https > 10.18.205.202.59857: Flags [P.], seq 1:
342, ack 277, win 513, length 341
12:33:59.646950 IP 10.18.205.202.59857 > pr1stfldb.istat.it.https: Flags [.], ack 342
, win 4090, length 0
```

Wireshark

Packet "sniffer": can capture of raw data from the network for analysis

- Lots of plugins to examine (or "dissect") almost any protocol
  - Assuming it isn't encrypted...
- Can show any packets that appear on your network interface, not just those tagged with your IP!
  - Called "promiscuous mode"
- Requires admin privileges to capture traffic

- Wireshark is a GUI app: for terminal only variants, see: tcpdump, tshark


Wi-Fi: en0
main toolbar
Apply a display filter …<⌘/>
Expression…
filter toolbar
No. Time Source Destination Protocol Length Info
1024 114.084415 10.18.205.202 13.226.162.80 TCP 66 59354 → 443 [ACK] Seq=79 Ack=79 Win=2
1025 114.846011 10.18.205.202 104.18.3.173 ICMP 98 Echo (ping) request id=0xdb41, seq=0
1026 114.854961 104.18.3.173 10.18.205.202 ICMP 98 Echo (ping) reply id=0xdb41, seq=0
1027 114.876848 10.18.205.202 142.250.180.67 TLSv1… 105 Application Data
1028 114.881968 142.250.180.67 10.18.205.202 TCP 66 443 →
1029 115.062304 142.250.180.67 10.18.205.202 TLSv1… 105 Appli
1030 115.062422 10.18.205.202 142.250.180.67 TCP 66 59188 → 443 [ACK] Seq=359 Ack=505 Win
1031 115.851198 10.18.205.202 104.18.3.173 ICMP 98 Echo (ping) request id=0xdb41, seq=1
1032 115.860957 104.18.3.173 10.18.205.202 ICMP 98 Echo (ping) reply id=0xdb41, seq=1
1033 116.028027 10.18.205.202 104.16.249.249 TLSv1… 122 Application Data
packet list pane
Frame 1025: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface 0
Ethernet II, Src: Apple_50:a3:83 (f0:18:98:50:a3:83), Dst: LannerEl_21:bc:c9 (00:90:0b:21:bc:c9)
Internet Protocol Version 4, Src: 10.18.205.202, Dst: 104.18.3.173
0100 .... = Version: 4
.... 0101 = Header Length: 20 bytes (5)
Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
Total Length: 84
packet details pane
0000 00 90 0b 21 bc c9 f0 18 98 50 a3 83 08 00 45 00
0010 00 54 76 e4 00 00 40 01 c0 29 0a 12 cd ca 68 12
0020 03 ad 08 00 22 fc db 41 00 00 60 48 a5 be 00 0d
0030 08 ab 08 09 0a 0b 0c 0d 0e 0f 10 11 12 13 14 15
0040 16 17 18 19 1a 1b 1c 1d 1e 1f 20 21 22 23 24 25
0050 26 27 28 29 2a 2b 2c 2d 2e 2f 30 31 32 33 34 35
0060 36 37
packet bytes pane
status bar
Ethernet (eth), 14 bytes
Packets: 938733 · Displayed: 938733 (100.0%) · Dropped: 0 (0.0%)
Profile: Default

Practice with Wireshark

- Checking a connection
  - Ping 127.0.0.1 (localhost)
  - Ping <your-ip-address> (ifconfig)
  - Ping www.brown.edu
- Traceroute www.brown.edu

ARP Protocol

IP and MAC Addresses

- Devices on a local area network have
  - IP addresses (network layer)
  - MAC addresses (data link layer)
- IP addresses are used for high level protocols
- MAC addresses are used for low level protocol
- Network administrator configures IP address and subnet on each machine
- How to translate IP Addresses into MAC addresses?

Address Resolution Protocol (ARP)

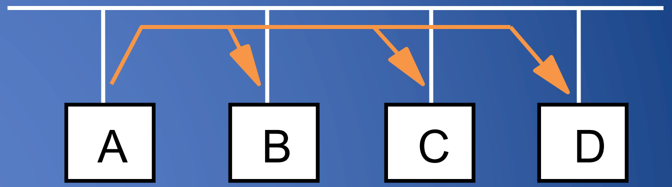
- Connects the network layer to the data link layer
- Maps IP addresses to MAC addresses
- Based on broadcast messages and local caching
- Does not support confidentiality, integrity, or authentication
- Defined as a part of RFC 826

ARP Messages

- ARP broadcasts in a frame a requests of type

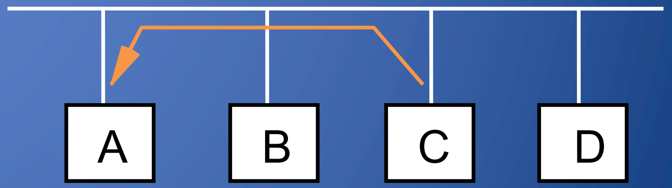
  who has <IP addressC >

  tell <IP addressA >



- Machine with <IP addressC> responds to requesting machine message

  <IP addressC > is at <MAC address>



- Requesting machine caches response

ARP Cache

- The Linux, Windows and OSX command arp - a displays the ARP table

Internet Address	Physical Address	Type
128.148.31.1	00-00-0c-07-ac-00	dynamic
128.148.31.15	00-0c-76-b2-d7-1d	dynamic
128.148.31.71	00-0c-76-b2-d0-d2	dynamic
128.148.31.75	00-0c-76-b2-d7-1d	dynamic
128.148.31.102	00-22-0c-a3-e4-00	dynamic

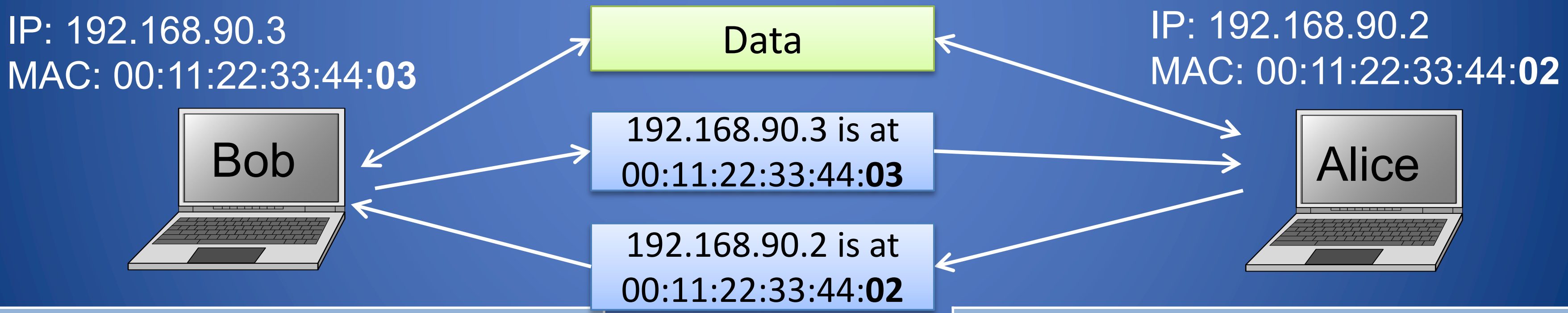
- Command arp –a –d flushes the ARP cache
- ARP cache entries are stored for a configurable amount of time

ARP Spoofing

- The ARP table is updated whenever an ARP response is received
- Requests are not tracked
- ARP announcements are not authenticated
- Machines trust each other
- A rogue machine can spoof other machines

ARP Normal Operation

- Normal operation
  - Alice communicates with Bob



ARP Cache	
192.168.90.2	00:11:22:33:44:**02**

ARP Cache	
192.168.90.3	00:11:22:33:44:**03**

Clicker Question (1)

After a great experience at CS166 TA hours, Bob decides to message Alice about how much he appreciates the CS166 staff. Eve would like to trick Bob into sending this network traffic to her (instead of Alice). Assuming Eve has access to everyone's MAC and IP, what ARP response could Eve send to Bob to accomplish this?

A. <Eve's IP> is at <Eve's MAC>
B. <Eve's IP> is at <Alice's MAC>
C. <Alice's IP> is at <Eve's MAC>
D. <Alice's IP> is at <Alice's MAC>

Clicker Question (1) Answer

After a great experience at CS166 TA hours, Bob decides to message Alice about how much he appreciates the CS166 staff. Eve would like to trick Bob into sending this network traffic to her (instead of Alice). Assuming Eve has access to everyone's MAC and IP, what ARP response could Eve send to Bob to accomplish this?

A. <Eve's IP> is at <Eve's MAC>

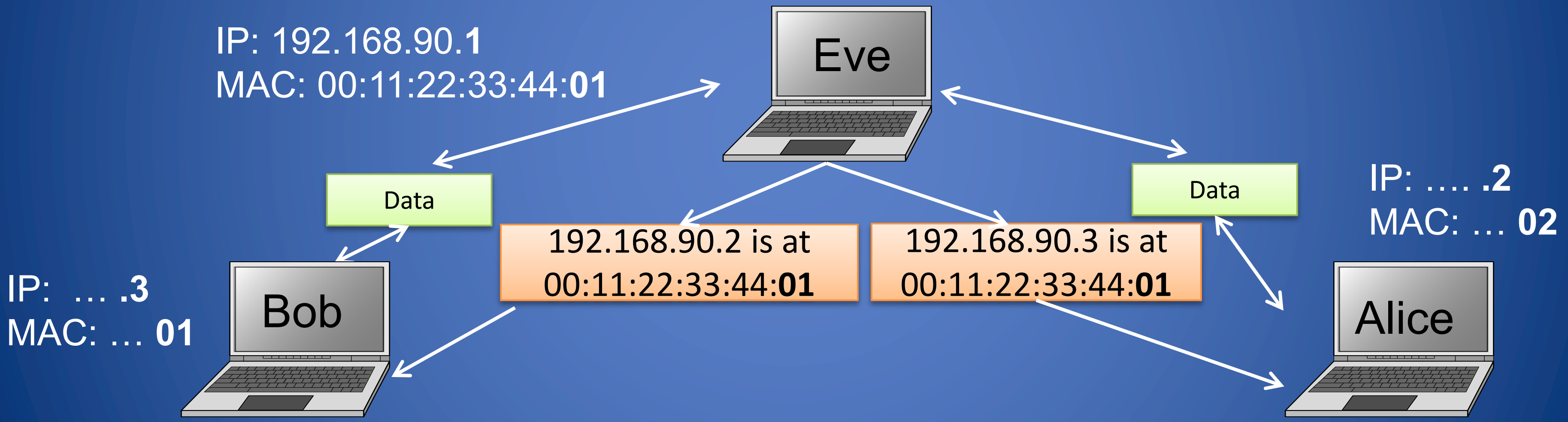
B. <Eve's IP> is at <Alice's MAC>

C. <Alice's IP> is at <Eve's MAC>

D. <Alice's IP> is at <Alice's MAC>

ARP Poisoning Attack

- Man-in-the-middle attack
  - ARP cache poisoning leads to eavesdropping



Poisoned ARP Cache	
192.168.90.2	00:11:22:33:44:**01**

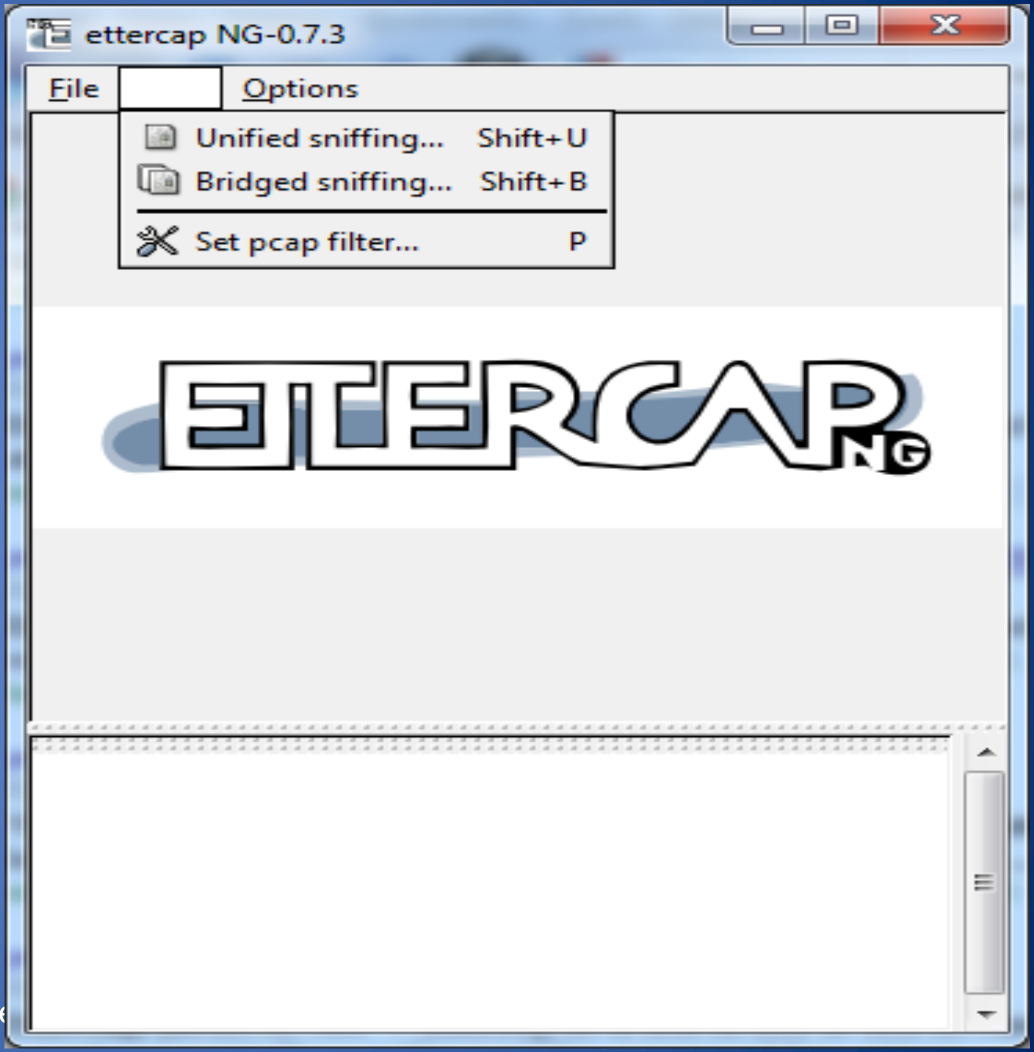
Poisoned ARP Cache	
192.168.90.3	00:11:22:33:44:**01**

ARP Poisoning & ARP Spoofing

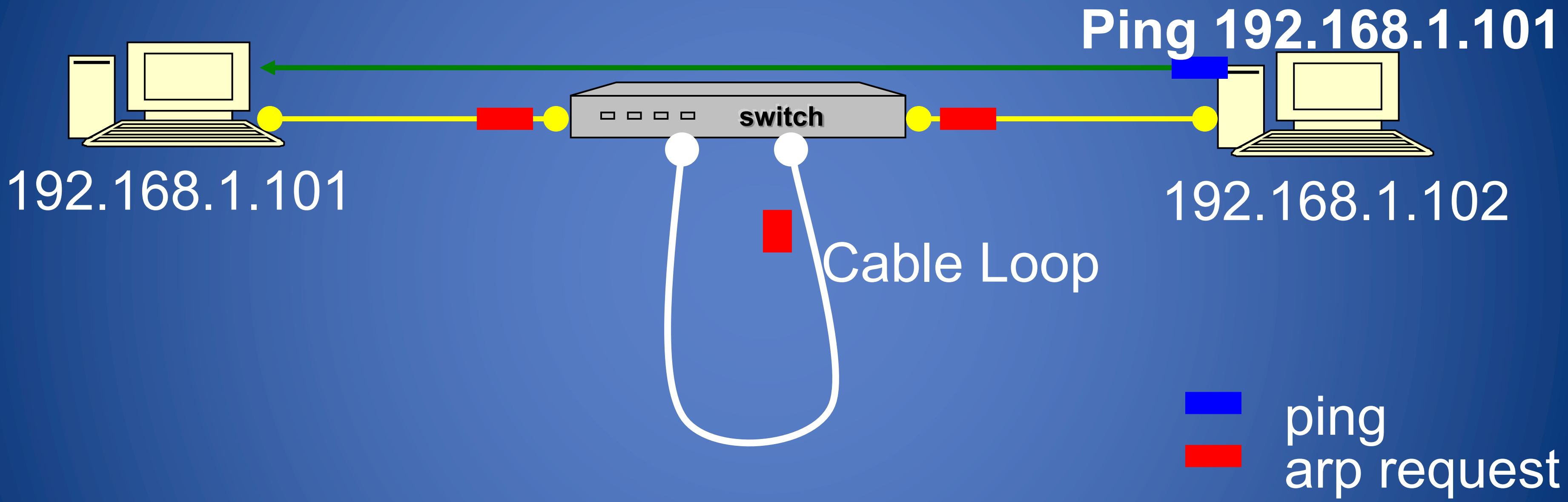
- Almost all ARP implementations are stateless
- An ARP cache updates every time that it receives an ARP reply
  - … even if it did not send any ARP request!
- Can "poison" ARP cache with gratuitous ARP replies
- Using static entries solves the problem but it is cumbersome to manage!

Ettercap

- Ettercap is a suite for man in the middle attacks on LAN
- In this demo we use:
  - Unified sniffing (promiscuous mode)
  - MiTM attack (arp poisoning)
  - Protocol dissection active and passive (telnet password retrieval)

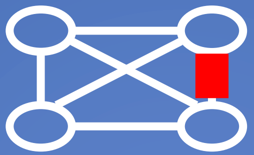


network DOS using ARP

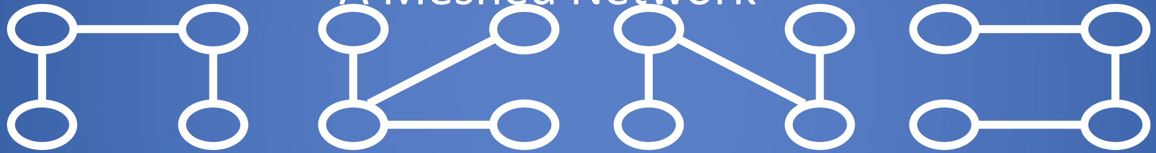


How can it be solved?

Spanning Tree Protocol (ISO 802.1D)



A Meshed Network



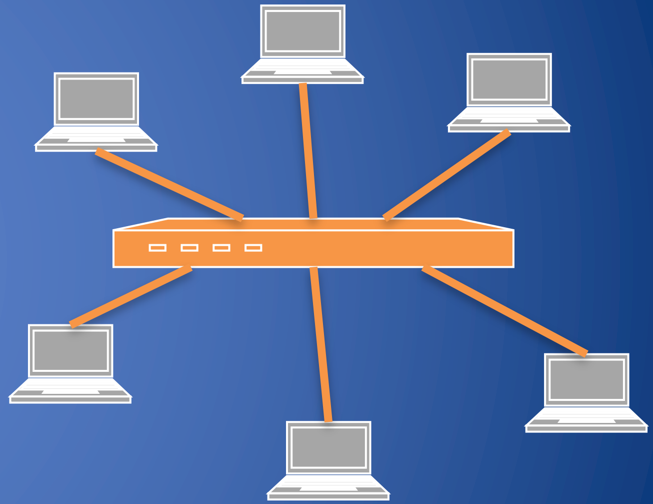
Four spanning trees of the Meshed Network

- Suppose you have a Meshed Network with bidirectional links that make loops/cycles…
- …then a spanning tree of the Meshed Network is the same network and no loops/cycles

Another attack on switches

Switching

- A switch connects devices on a local area network (LAN)
- Has multiple interfaces, or ports
- Operates on link-layer frames
- As devices connect, learns MAC addresses of some or all the devices on the network



Extra: Example

- Table initially empty

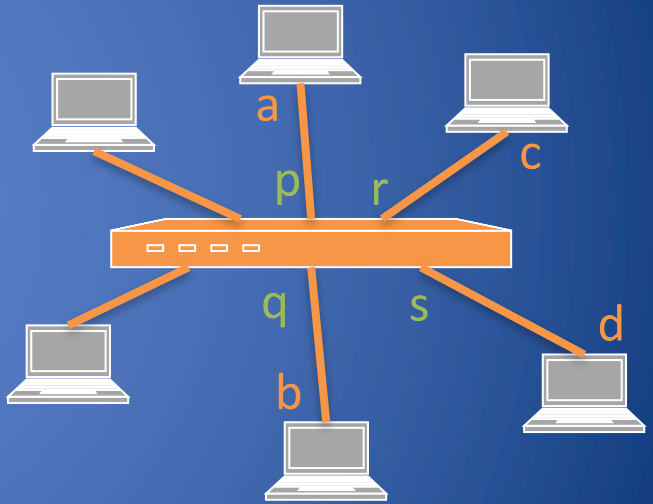
- Frame (a, b) broadcast;
  - entry (a, p) added to table

a	p

- Frame (c, a) forwarded on p
  - entry (c, r) added to table

a	p
c	r

- Frame (a, c) forwarded on r
  - table unchanged
- Frame (a, d) broadcast
  - table unchanged



Frame Processing

- Switch receives on port p frame with source s and destination d

  e = get(d)

  if e == null [device with address d not known]

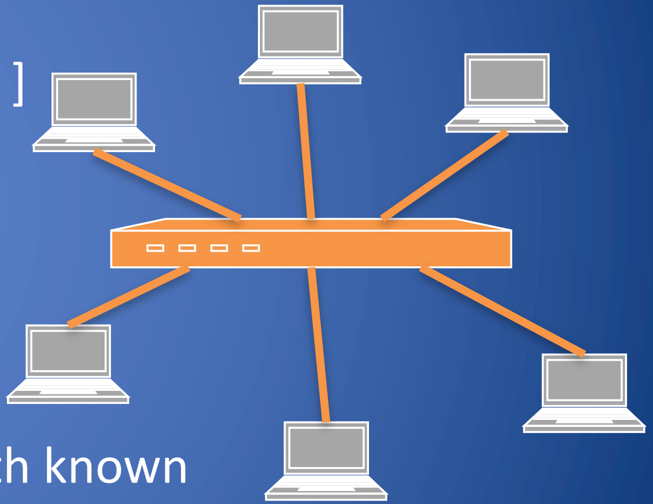
  broadcast frame on all ports but p

  else [device with address d known]

  forward frame on e.port

  put(s, p) [adds or updates table entry]

- For a network with a single switch, a frame with known destination address is directly delivered only to the recipient

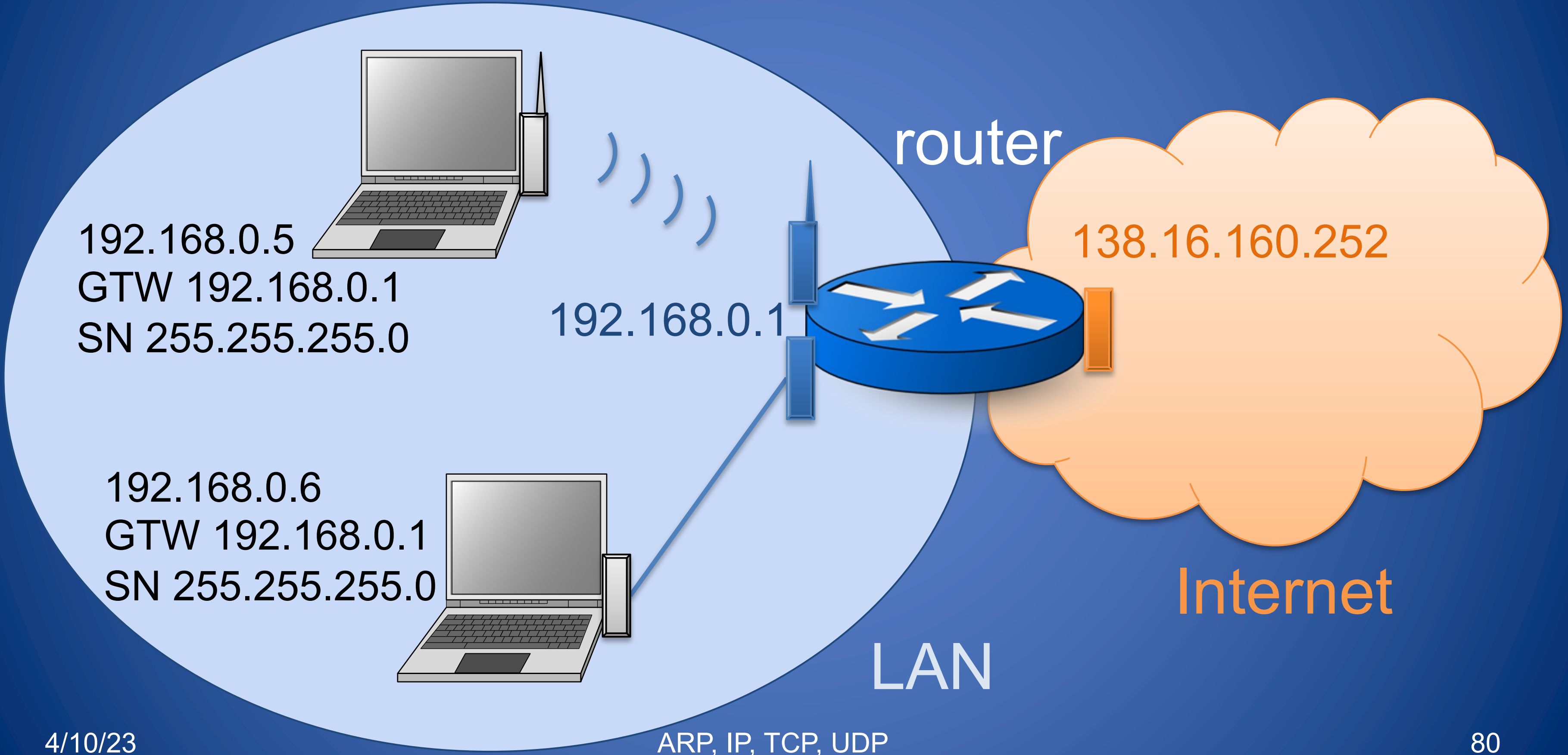


Attack on a learning switch

- Idea: flood the switch with many packets from different source MAC addresses

- If MAC table is full, switch just broadcasts all packets to all ports

From the LAN to the Internet



Break!

TCP and UDP Protocols

Transport Layer

- The transport layer supports one or more of the following features
  A. Reliable data transfer (resending of dropped packets)
  B. In-order delivery of segments of file or media stream
  C. Congestion control (request longer/shorter segments)
  D. Ability to distinguish multiple applications on same host via ports (16-bit numbers)
- The main transport layer protocols are
  – UDP (supports B, D)
  – TCP (supports A, B, C, D)

User Datagram Protocol (UDP)

- Stateless, unreliable transport-layer protocol
- Can distinguish multiple concurrent applications on a single host
- No delivery guarantees or acknowledgments
  - Efficient
  - Suitable for audio/video streaming and voice calls
  - Unsuitable for file transmission and text messaging

Transmission Control Protocol (TCP)

- Stateful protocol for reliable data transfer, in-order delivery of messages and ability to distinguish multiple applications on same host
  - HTTP and SSH are built on top of TCP
- TCP packages a data stream it into segments transported by IP
  - Order maintained by marking each packet with sequence number
  - Every time TCP receives a packet, it sends out an ACK to indicate successful receipt of the packet
- TCP generally checks data transmitted by comparing a checksum of the data with a checksum encoded in the packet

Ports

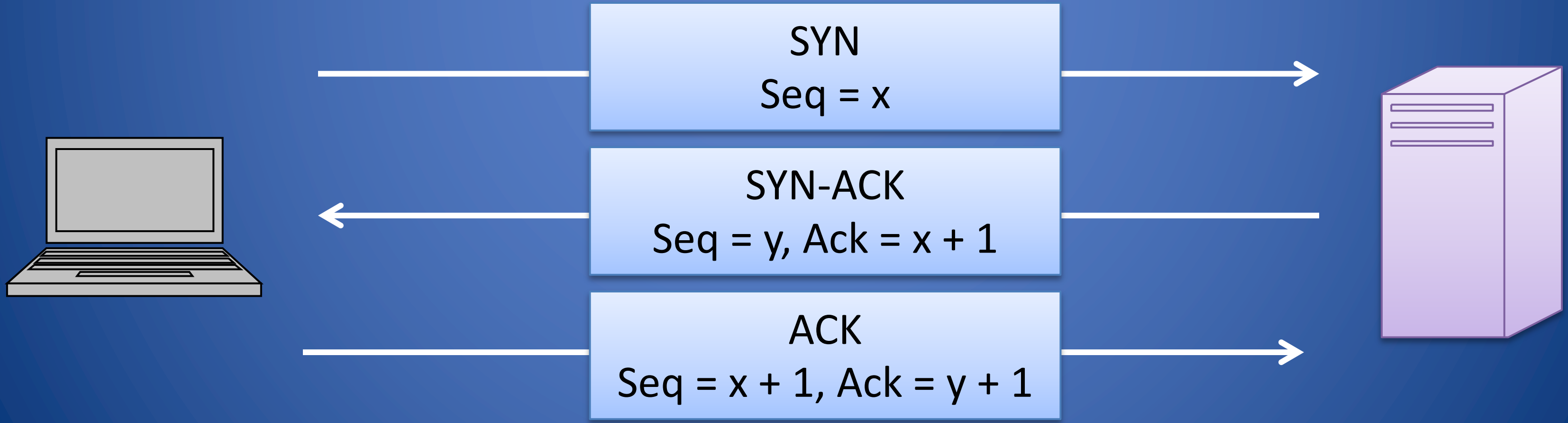
- TCP and UDP support concurrent applications on same server
- Ports are 16 bit numbers identifying where data is directed
- The TCP or UDP header includes source and a destination port
- Ports 0 through 1023 are reserved for client-to-server requests in known application protocols
  - E.g., HTTPS uses 443 and SSH uses 22
- Ports 1024 through 49151 are known as user ports, and typically provide the return channel for the server-to-client response

TCP Packet Format

Bit Offset	0-3	4-7	8-15	16-18	19-31
0	**Source Port**			**Destination Port**	
32	**Sequence Number**				
64	**Acknowledgment Number**				
96	Offset	Reserved	Flags	**Window Size**	
128	Checksum			Urgent Pointer	
160	Options				
>= 160	Payload				

Establishing TCP Connections

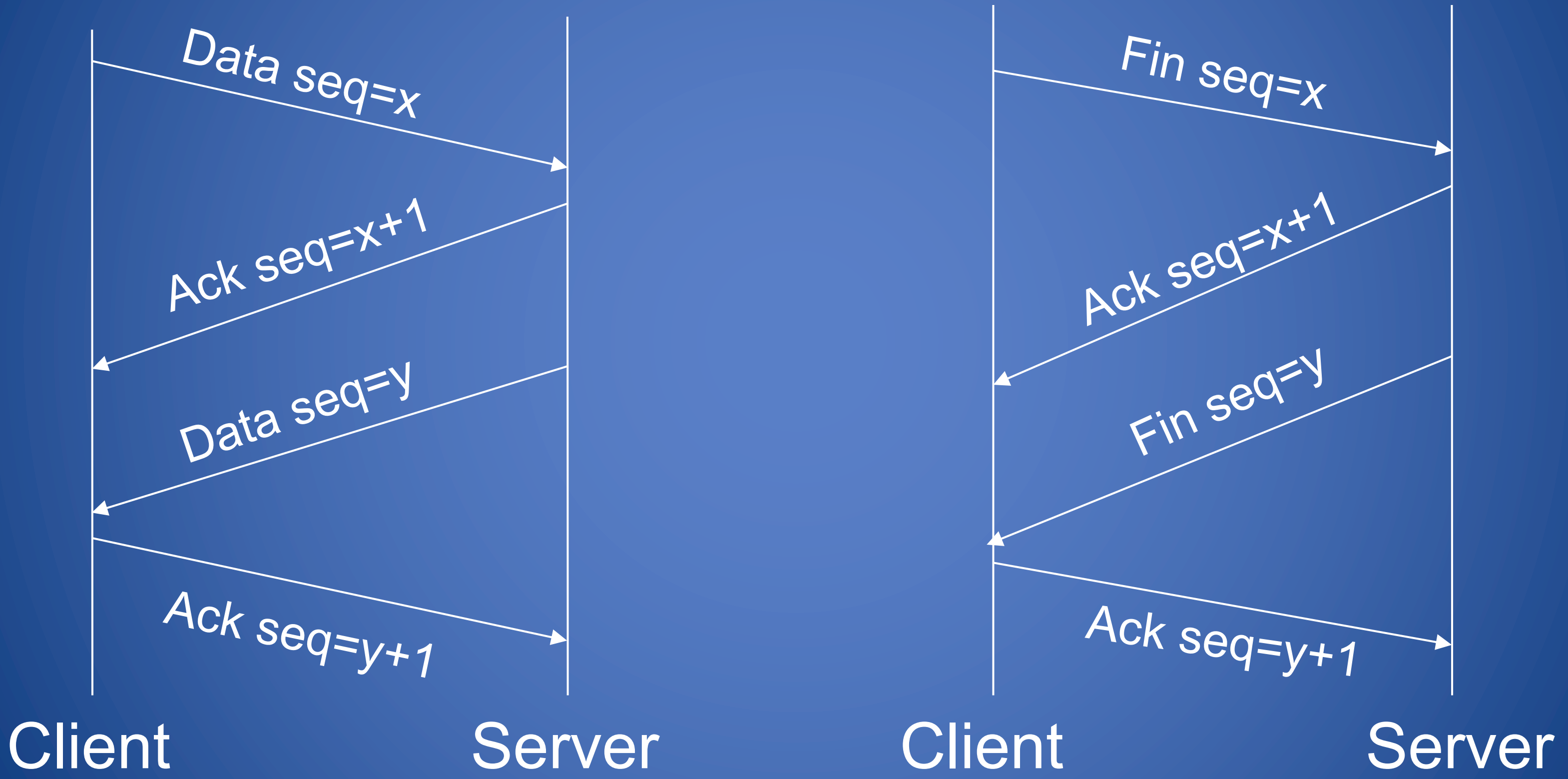
- TCP connections are established through a three-way handshake
- The server generally is a passive listener, waiting for a connection request
- The client requests a connection by sending out a SYN packet
- The server responds by sending a SYN/ACK packet, acknowledging the connection
- The client responds by sending an ACK to the server, thus establishing connection



TCP Data Transfer

- The three way handshake initializes sequence numbers for the request and response data streams
- The TCP header includes a 16 bit checksum of the payload and parts of the header, including source and destination
- Acknowledgment or lack thereof is used by TCP to keep track of network congestion and control flow
- TCP connections are cleanly terminated with a 4-way handshake
  - The client which wishes to terminate the connection sends a FIN message to the other client
  - The other client responds by sending an ACK
  - The other client sends a FIN
  - The original client now sends an ACK, and the connection is terminated

TCP Data Transfer and Teardown



Clicker Question (2)

Eve is once again up to no good. She decides to modify the payload of a TCP packet that Alice sends to Bob by randomly flipping a bit. Would Bob be able to detect this?

A. Yes, since most likely the checksum will not match
B. Yes, since the packet will be totally corrupted
C. No, since there are no security features in TCP
D. No, since it is computationally infeasible

Clicker Question (2) - Answer

Eve is once again up to no good. She decides to modify the payload of a TCP packet that Alice sends to Bob by randomly flipping a bit. Would Bob be able to detect this?

A. **Yes, since most likely the checksum will not match**
B. Yes, since the packet will be totally corrupted
C. No, since there are no security features in TCP
D. No, since it is computationally infeasible

Telnet Protocol (RFC 854)

- Telnet is a protocol that provides a general, bi-directional, not encrypted communication
- `telnet` is a generic TCP client
  - Allows a computer to connect to another one
  - Provides remote login capabilities to computers on the Internet
  - Sends whatever you type
  - Prints whatever comes back
  - Useful for testing TCP servers (ASCII based protocols)

What We Have Learned

- IP address space allocation
- ARP protocol
- ARP poisoning attack
- Transport layer protocols
  - TCP for reliable transmission
  - UDP when packet loss/corruption is tolerated
- Lack of built-in security for link, network, and transport layer protocols
  - Security enhanced protocols have been developed for these layers
  - Alternate solution is to provide security at application layer