Link-Layer and ICMP

EE122 Fall 2012

Scott Shenker

http://inst.eecs.berkeley.edu/~ee122/

Materials with thanks to Jennifer Rexford, Ion Stoica, Vern Paxson
and other colleagues at Princeton and UC Berkeley

UCB Startup Fair

October 23rd
Pauley Ballroom
12pm - 4pm

upload your resume at **ucbstartupfair.com**

sponsored by: stack**overflow** careers

Project 1

- Distribution of grades:
  - 50% perfect score of 240
  - 90% above 140
- Giving people a second chance:
  - Fix your project, get it running
  - We’ll figure out the penalties later
- Constraints:
  - Fix must cause *multiple* test cases to go from fail to pass
  - Regrades get maximum score of 200
  - Contact Anand/Colin for details……

Midterm

- Average score 104 (out of 119)
- Standard deviation 11
- 50th percentile 107
- 90th percentile 115

Question-by-question….

1. True/False: about 10% got full credit
   Peak about 18 out of 20
2. Multiple choice: about 8% got full credit
   Peak about 18 out of 20
   IP checksum only looks at header
   UDP header does not include addresses
3. TCP Basics: about 50% got full credit
4. Seq. of Messages: about 80% got full credit
5. The Real World: about 90% got full credit

Question-by-question….

6. Timer values: about 50% got full credit
7. Addressing: about 70% got full credit
8. Learning switch: about 60% got full credit
9. DNS: about 70% got full credit
10. Sliding window: about 70% got full credit

Humorous answers….

- Who is the unsung hero?
  - “me”
  - “Al Gore” (surprisingly popular answer)

- What letter caused the first demo to fail?
  - “love letter”
  - “S for shenker”

- In what year?
  - “1776”,
  - “122 BCE”
  - “12000 BC (there was time travel involved)”

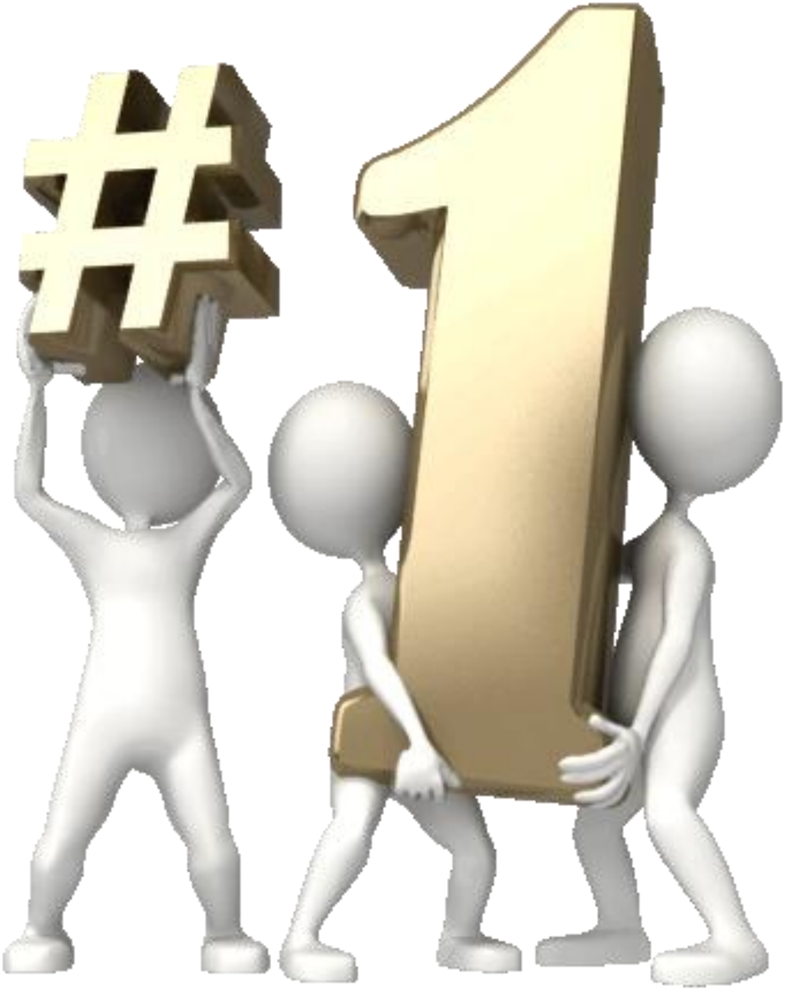
Suggested Bonus Questions…

- What is Scott Shenker's brother's profession?
- How many citations does Shenker have?
- Draw a giraffe

The Bet….

- 8 people with a score of 119 on questions 1-10
  - I geared the review to the test ***way*** too closely
  - But I felt like I had not sufficiently covered the "putting it all together" questions in lecture/sections
- 5 of the 119 scores got three bonus questions right
  - *The fact you knew the answers to the bonus questions is a very bad sign….*
- **0 of the 119 scores got four bonus questions**

Victory is Mine!



But in the spirit of fair play….

- I won’t collect on the bet
- I will donate my stake of $220 to the EE122 review sessions refreshment fund
- I’m taking bets for the final….(10:1 odds again)
- I was accused (in writing) for “cheating” on the bet
  - I won’t forget who you are…….
  - P. S. My TAs agree with you
  - **P. P. S. None of them will ever graduate**

Where Are We?

What Do We Know?

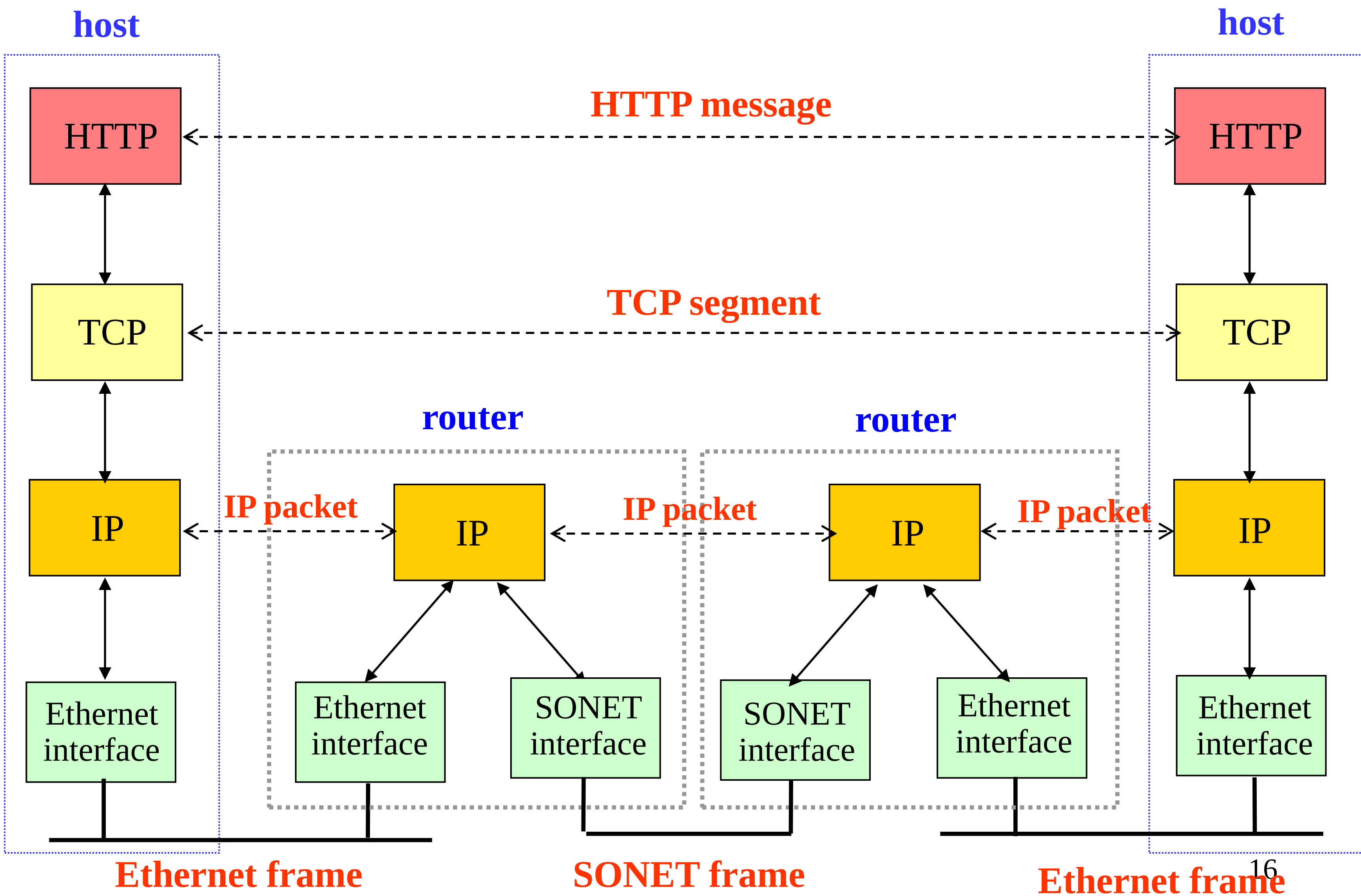
- How to route
  - L2 (learning switches)
  - L3 (DV, LS)
- How to get an IP address (DHCP)
- How to resolve names to IP addresses (DNS)
- How to forward packets (LPM)
- How to deliver packet reliably (TCP)
- How to access content (HTTP)
- ……

Missing Pieces (covered today)

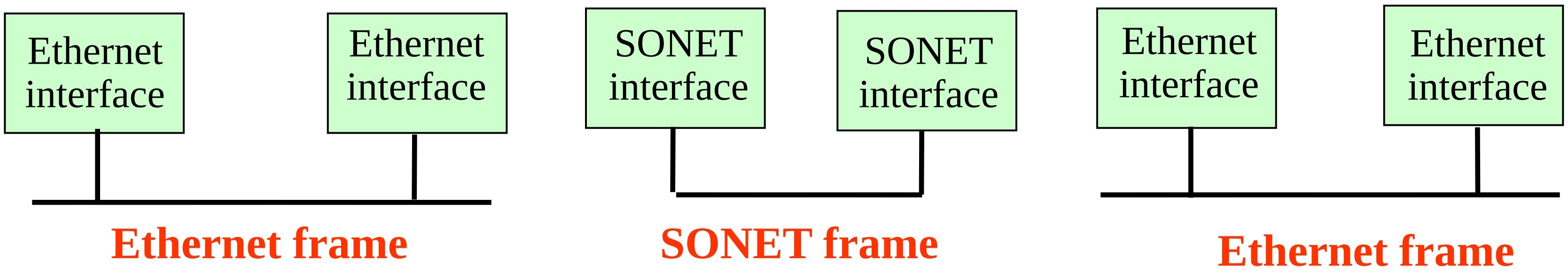
- Basics of link-layer (L2) networks
  - Will do details of ethernet later
- Using link-layer networks to reach destination
  - L2 involved at first/last hops (and in between)
- How do I find out about network problems?
  - Loops, MTU limitations, etc.

Background on Link-Layer

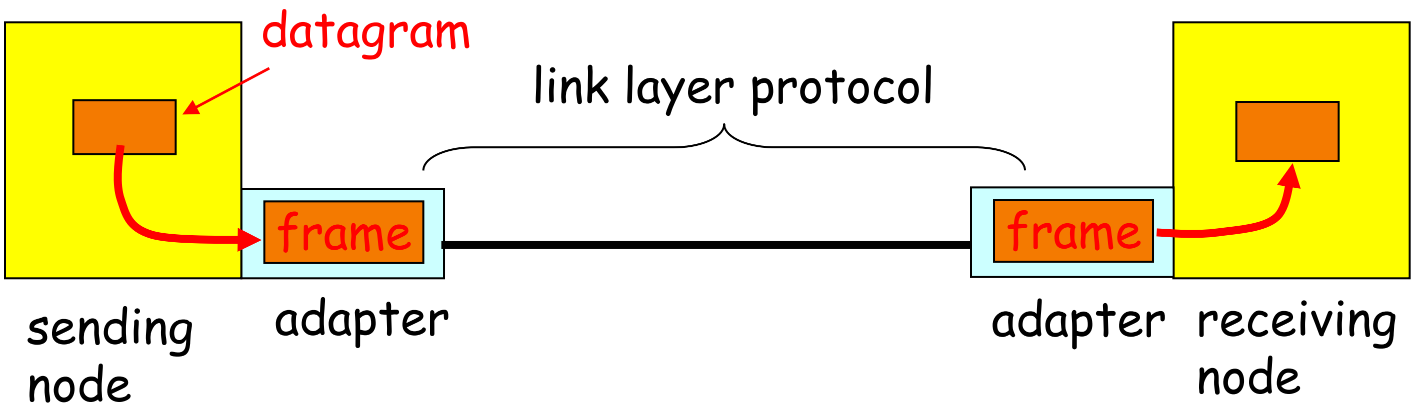
Message, Segment, Packet, and Frame



Focus on Link-Layer



Adapter-to-Adapter Communication



- Link layer implemented in adapter (*network interface card*; **NIC**)
  - Ethernet card, 802.11 card
- Sending side:
  - Encapsulates datagram in a frame
  - Determines local addressing, adds error checking, controls transmission
- Receiving side
  - Recognizes arrival, looks for errors, possibly acknowledges
  - Extracts datagram and passes to receiving node

Link-Layer Services

- Encoding
  - Representing the 0s and 1s
- Framing
  - Encapsulating packet into frame, adding header, trailer
  - Using MAC addresses rather than IP addresses
- Error detection
  - Errors caused by signal attenuation, noise
  - Receiver detects presence, may ask for repeat
- Resolving contention
  - Deciding who gets to transmit when multiple senders want to use a shared media
- Flow control (pacing between sender & receiver)

MAC Address vs. IP Address

- MAC addresses (used in link-layer)
  - Hard-coded in read-only memory when adapter is built
  - Like a social security number
  - Flat name space of 48 bits (e.g., 00-0E-9B-6E-49-76)
  - Portable, and can stay the same as the host moves
  - Used to get packet between interfaces on same network
- IP addresses
  - Configured, or learned dynamically
  - Like a postal mailing address
  - Hierarchical name space of 32 bits (e.g., 12.178.66.9)
  - Not portable, and depends on where the host is attached
  - Used to get a packet to destination IP subnet

Broadcast at Link-Level

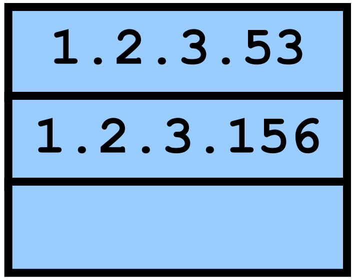
- Use broadcast address: ff:ff:ff:ff:ff:ff
- If have return MAC address, use that in response
- Unless want everyone to know result

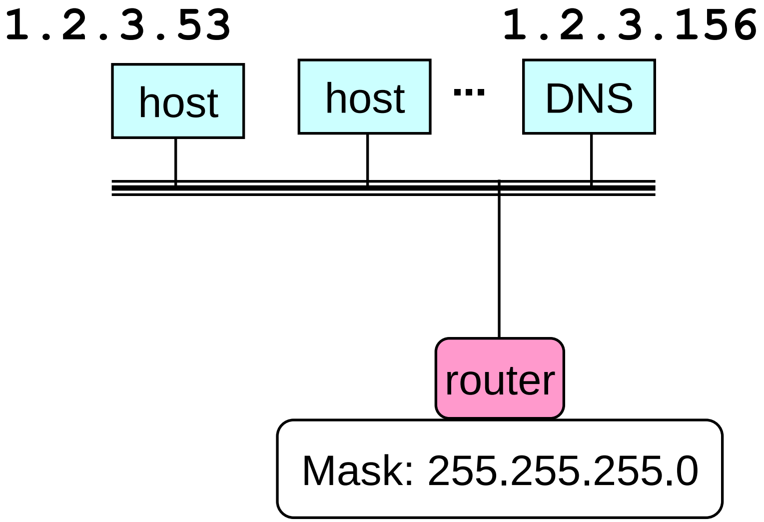
Broadcast at IP Level

- Can't broadcast to all IP hosts
- But application might want to send “local” broadcast
- Uses IP broadcast address 255.255.255.255
- Link-layer then uses link-layer broadcast

Sending Packets Over Link-Layer

IP packet





- Adapters only understand MAC addresses
  - Translate the destination IP address to MAC address
  - Encapsulate the IP packet inside a link-level frame

Steps in Sending a Packet

What do hosts need to know?

And how do they find out?

Steps in reaching a Host

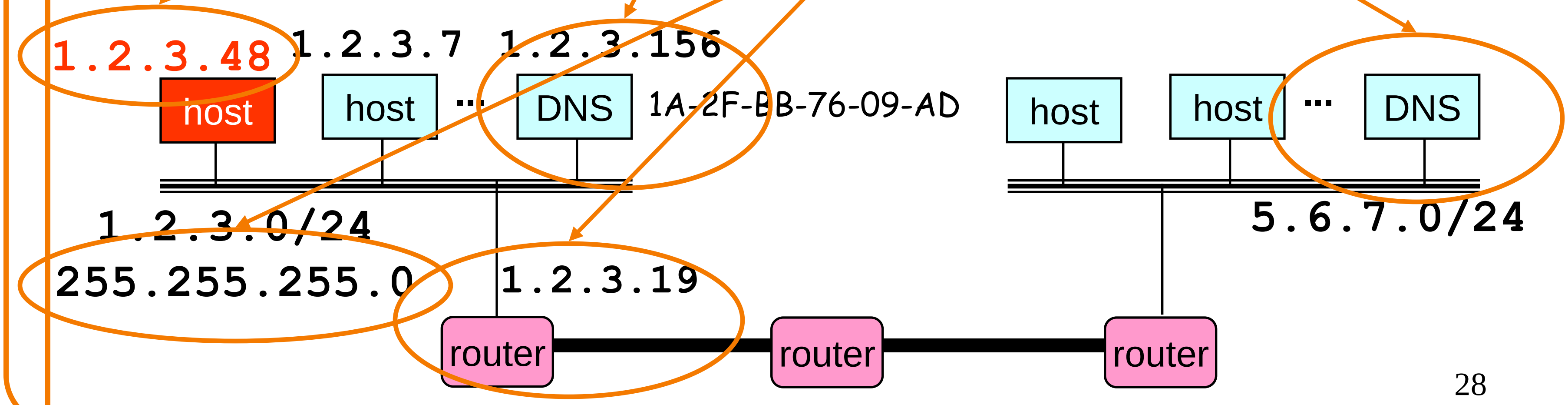
- First look up IP address
- Need to know where local DNS server is
  - **DHCP**
- Also needs to know its own IP address
  - **DHCP**

Sending a Packet

- On same subnet:
  - Use MAC address of destination.
  - ***How do hosts know?***
- On some other subnet:
  - Use MAC address of first-hop router.
  - ***How do they know?***
- And how can a host tell whether destination is on same or other subnet?
  - Use the netmask
  - **DHCP**

DHCP Refresher

- Dynamic Host Configuration Protocol (DHCP)
  - End host learns how to send packets
  - Learn IP address, DNS servers, "gateway", what's local
- Have already described DHCP operation
  - Sequence of broadcasts, no configuration needed



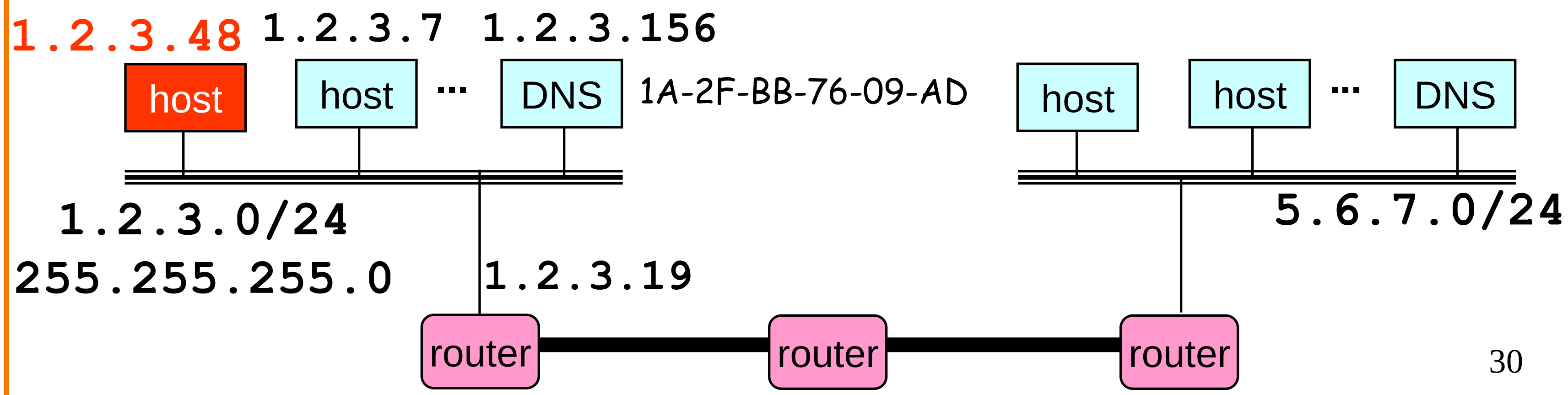
DHCP Supplies Basic Information

- IP address
- Mask
- Gateway router
- DNS server

- **Now what?**

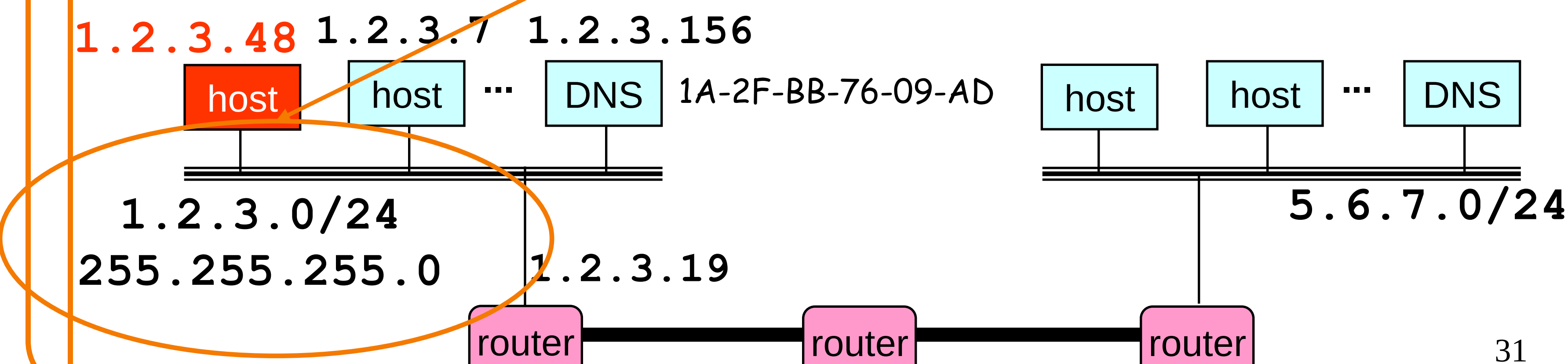
Sending A Packet: Which Destination?

- If destination is on the local network
  - Need to address it directly (MAC address)
- If destination is not local ("remote")
  - Need to figure out the first "hop" on the local network
  - Need MAC address of first hop router



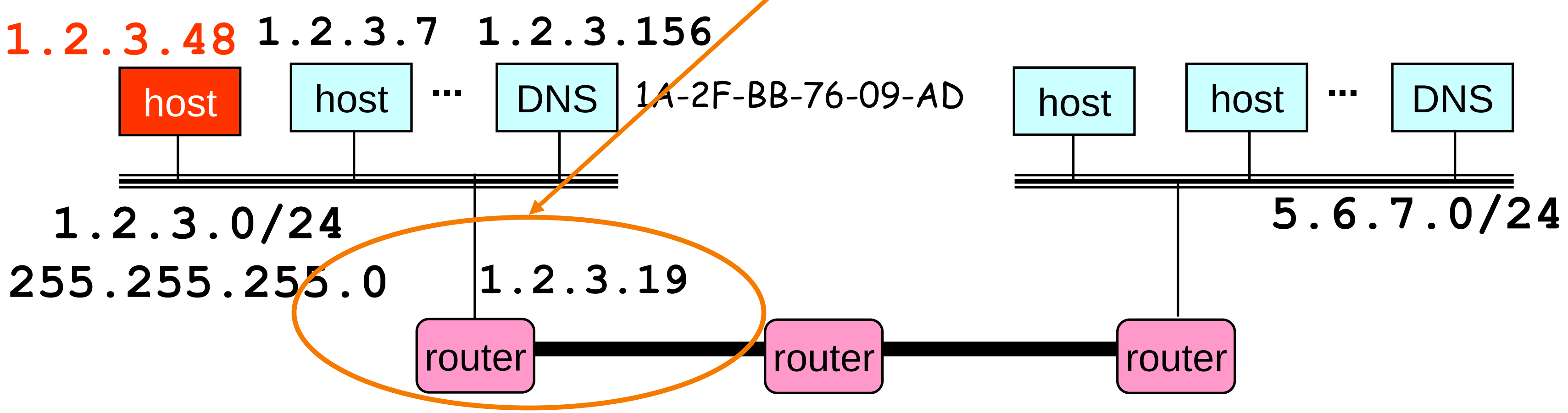
Determining if Address is Local

- Use the netmask
  - E.g., mask destination IP address w/ **255.255.255.0**
- Is it same value as our own masked address?
  - o Yes = local
  - o No = remote



In Both Cases, Need to Send Locally

- If it's remote, look up first hop in (very small) local routing table (in case there are multiple first hops)
  - E.g., by default, route via **1.2.3.19**
  - Now do the local case but for **1.2.3.19** rather than ultimate destination IP address



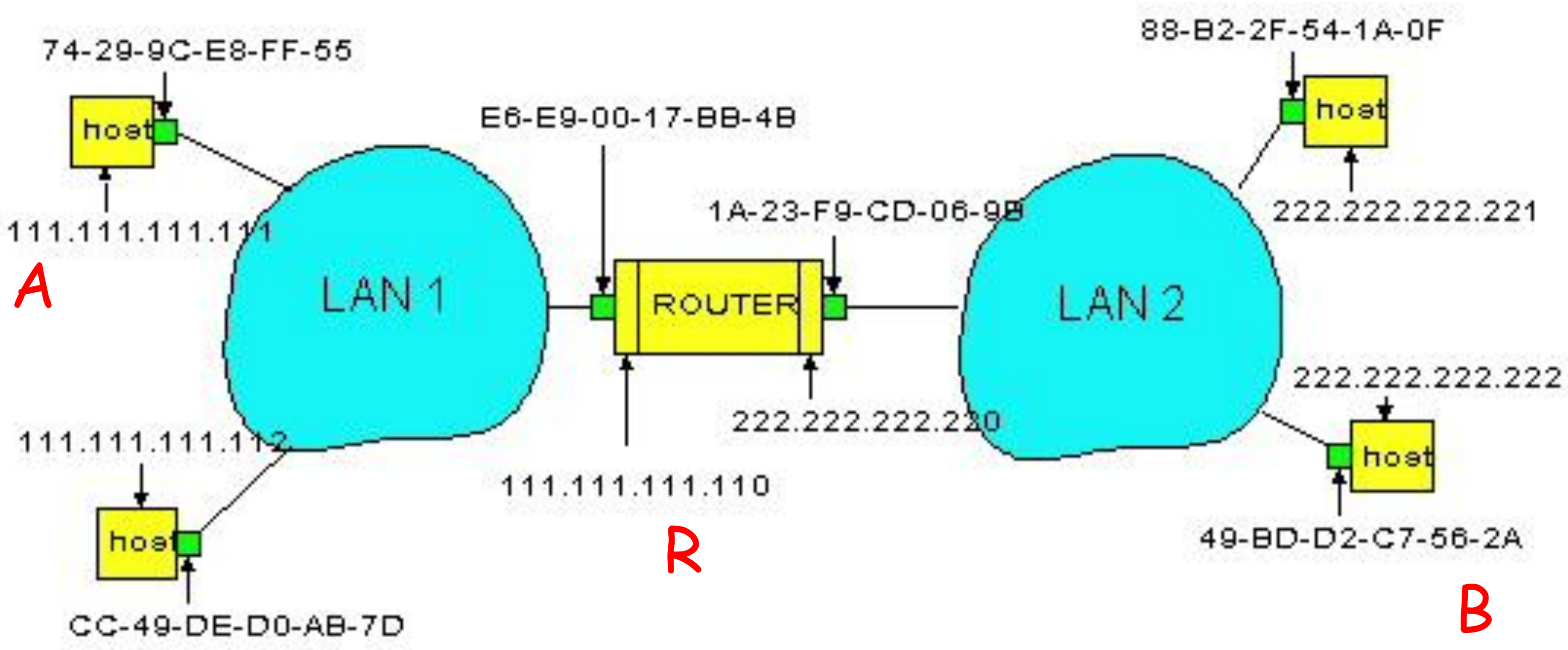
- For the local case, need to determine the destination's MAC address

Address Resolution Protocol

- Every node maintains an ARP table
  - <IP address, MAC address> pair
- Consult the table when sending a packet
  - Map destination IP address to destination MAC address
  - Encapsulate and transmit the data packet
- But: what if IP address not in the table?
  - Sender broadcasts: "**Who has IP address 1.2.3.156**?"
  - Receiver responds: "**MAC address 58-23-D7-FA-20-B0**"
  - Sender caches result in its ARP table

Example: A Sending a Packet to B

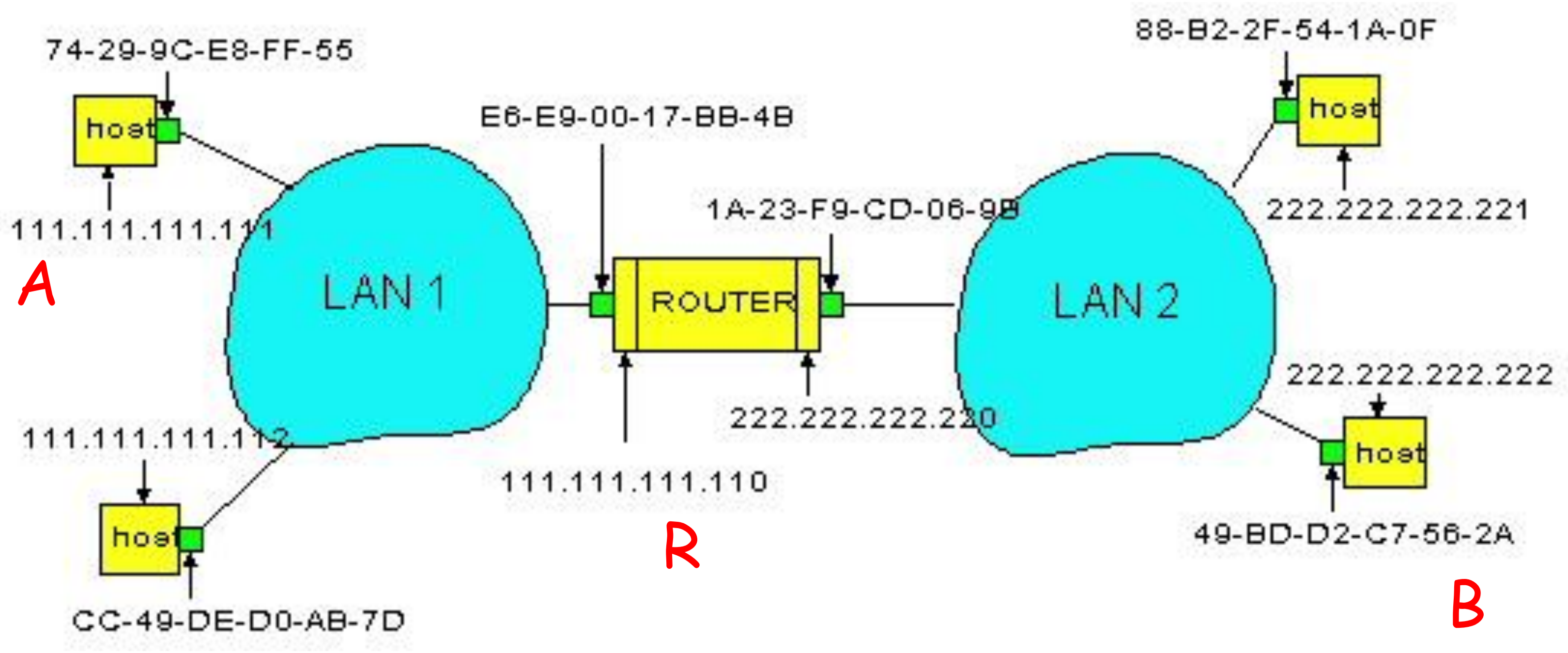
How does host A send an IP packet to host B?



Take a few minutes, break into groups, figure out how this would work…..

Example: A Sending a Packet to B

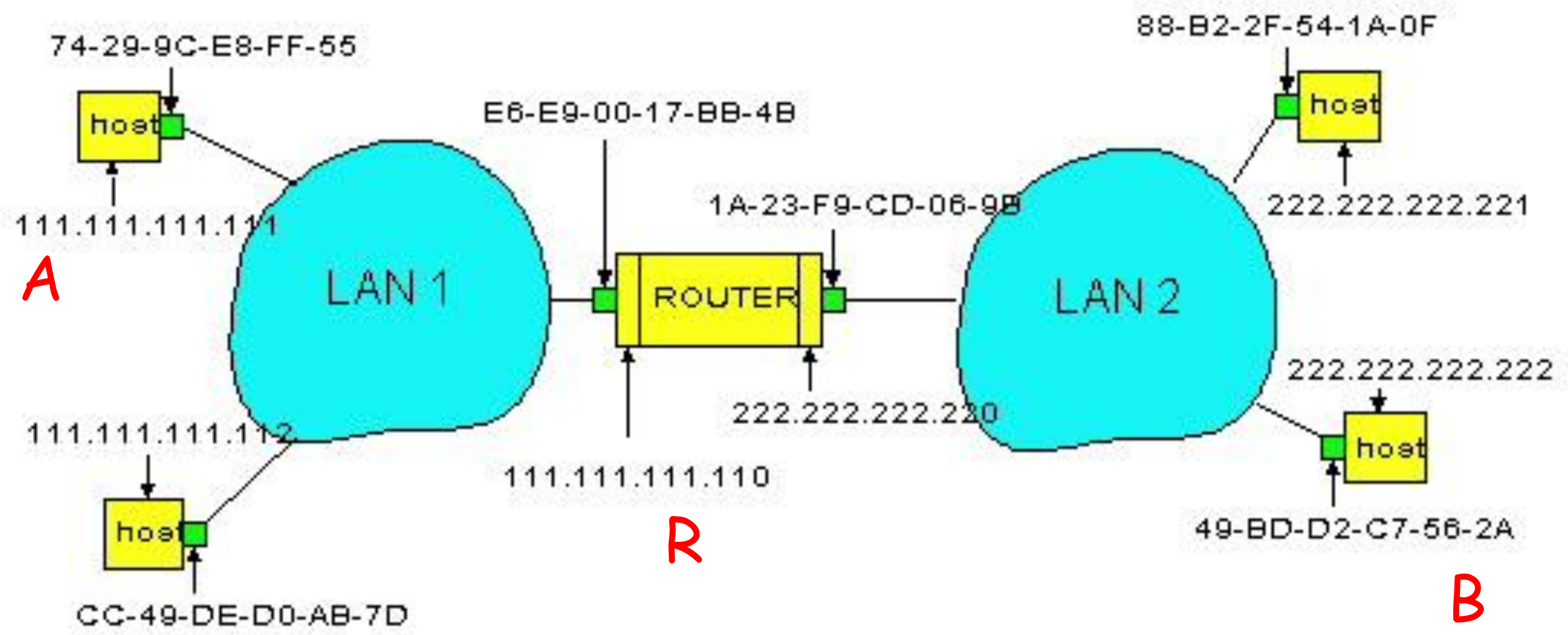
How does host A send an IP packet to host B?



1. A sends packet to R.
2. R sends packet to B.

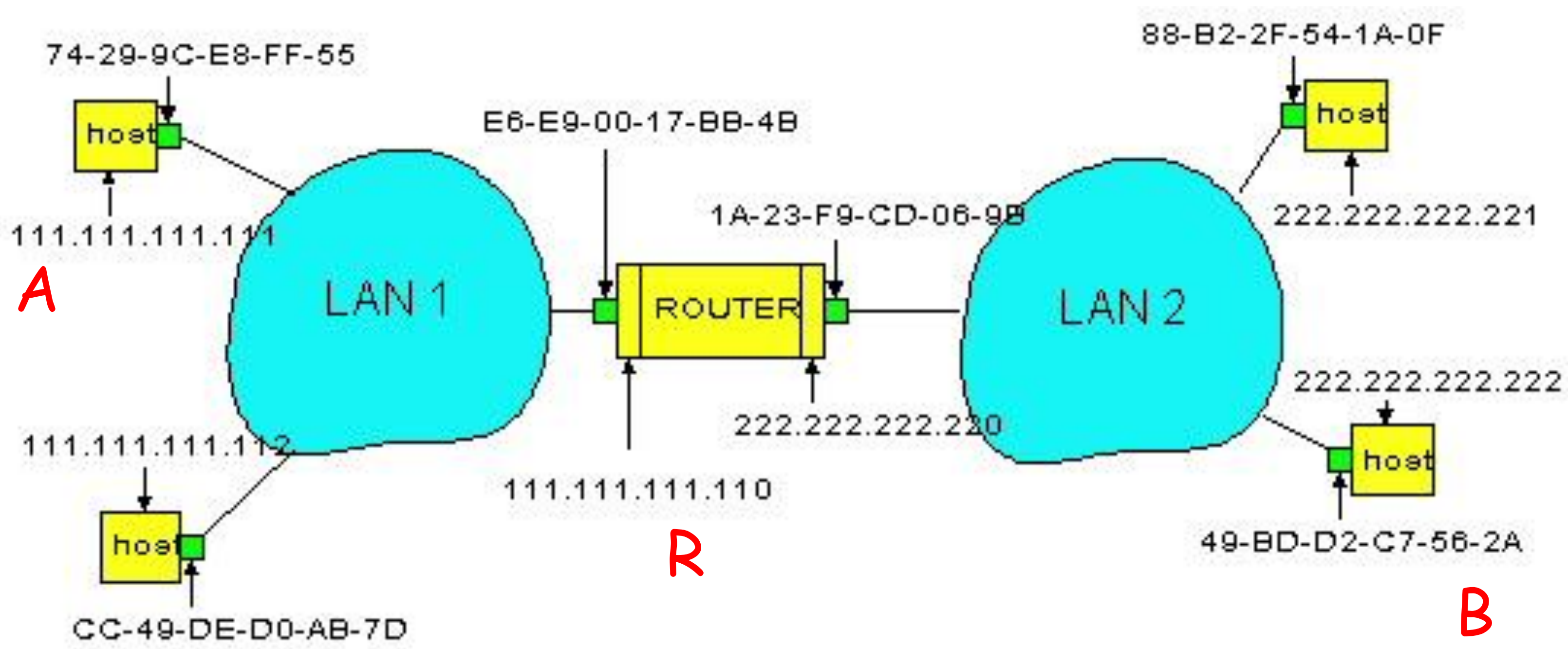
Host A Decides to Send Through R

- Host A constructs an IP packet to send to B
  - Source 111.111.111.111, destination 222.222.222.222
- Host A has a gateway router R
  - Used to reach destinations outside of 111.111.111.0/24
  - Address 111.111.111.110 for R learned via DHCP



Host A Sends Packet Through R

- Host A learns the MAC address of R's interface
  - ARP request: broadcast request for 111.111.111.110
  - ARP response: R responds with E6-E9-00-17-BB-4B
- Host A encapsulates the packet and sends to R

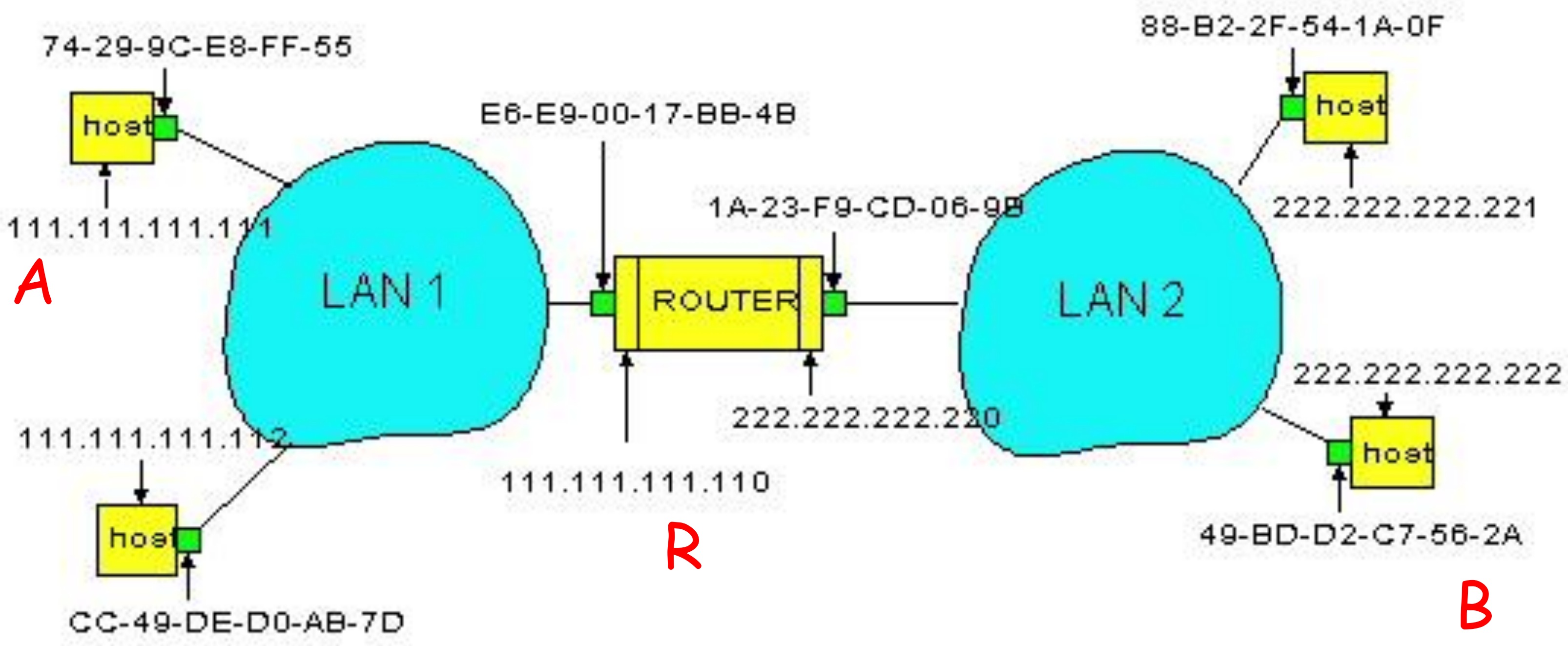


Two points:

- Routing table points to this port
- Destination address is within mask of port's address (i.e., local)

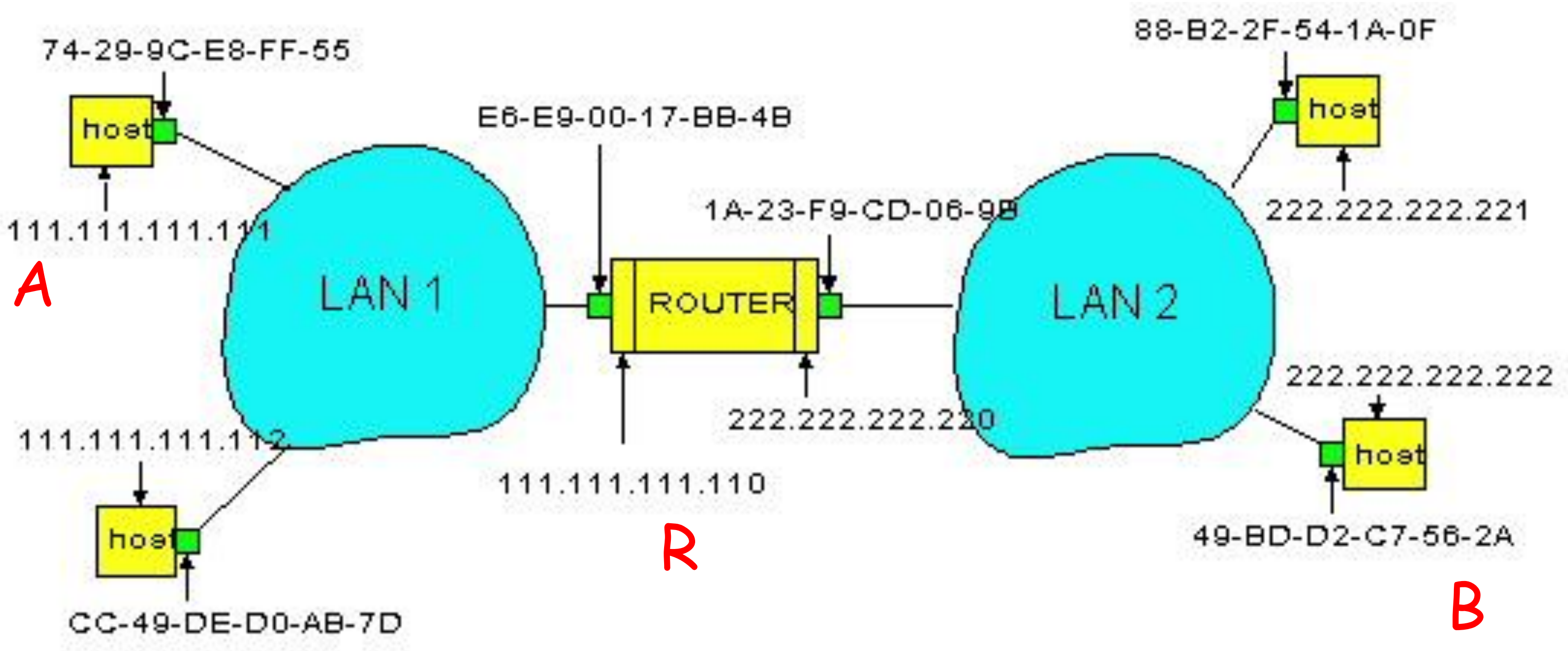
.222

- Router R cons… its forwarding table
  - Packet matches 222.222.222.0/24 via other adapter



R Sends Packet to B

- Router R's learns the MAC address of host B
  - ARP request: broadcast request for 222.222.222.222
  - ARP response: B responds with 49-BD-D2-C7-56-2A
- Router R encapsulates the packet and sends to B



Security Analysis of ARP

- Impersonation
  - Any node that hears request can answer …
  - … and can say whatever they want

- Actual legit receiver never sees a problem
  - Because even though later packets carry its IP address, its NIC doesn't capture them since not its MAC address

Key Ideas in Both ARP and DHCP

- Broadcasting: Can use broadcast to make contact
  - Scalable because of limited size

- Caching: remember the past for a while
  - Store the information you learn to reduce overhead
  - Remember your own address & other host's addresses

- Soft state: eventually forget the past
  - Associate a time-to-live field with the information
  - … and either refresh or discard the information
  - Key for robustness in the face of unpredictable change

Why Not Use DNS-Like Tables?

- When host arrives:
  - Assign it an IP address that will last as long it is present
  - Add an entry into a table in DNS-server that maps MAC to IP addresses (i.e., no need for ARP!)

- **Think about it for a few minutes, talk in groups**

Two Different Issues

- Setting up the database:
  - Names: explicit creation, tied to “static” addresses
    - DNS need only handle occasional updates
  - Hosts: come and go without explicitly informing network
    - Must do MAC-IP mapping on demand
  - But could leverage DHCP
    - DHCP knows when a host arrives
    - And DHCP messages already contain MAC addresses
- Using the MAC address:
  - So if I get MAC address when I look up address in DNS, how can I use that information?
  - The database must live in each router and host for it to save any time….but it does cut down on broadcasting

5 Minute Break

Network Control Messages
(and how to use them for discovery)

What Errors Might A Router See?

- Dead-end: No route to destination
- Sign of a loop: TTL expires
- Can’t physically forward: packet too big
  - And has DF flag set
- Can’t keep up with traffic: buffer overflowing
- Header corruption or ill-formed packets
- ….

Which should network tell host about?

- No route to destination?
  - Host can’t detect or fix routing failure.
- TTL expires?
  - Host can’t detect or fix routing loop.
- ...
  - ...erence

This assumes we want to bind the meaning of packet drops to congestion

- Buffer overflowing?
  - Transport congestion control can detect/deal with this
- Header corruption or ill-formed packets?
  - Host can’t fix corruption, but can fix formatting errors

Router Response to Problems?

- Router doesn't really need to respond
  - Best effort means never having to say you're sorry
  - So, IP could conceivably just silently drop packets
- Network is already trying its best
  - Routing is already trying to avoid loops/dead-ends
  - Network can't reduce packet size (in DF packets)
  - Network can't reduce load, nor fix format problems
- What more can/should it do?

Error Reporting Helps Diagnosis

- Silent failures are **really hard to diagnose**
- IP includes feedback mechanism for network problems, so they don't go undetected
- Internet Control Message Protocol (ICMP)
- The Internet "print" statement
- Runs on IP, but viewed as *integral* part of IP

Internet Control Message Protocol

- Triggered when IP packet encounters a problem
  - E.g., **Time Exceeded** or **Destination Unreachable**
- ICMP packet sent back to the source IP address
  - Includes the error information (e.g., type and code)
  - IP header plus 8+ byte *excerpt* from original packet
- Source host receives the ICMP packet
  - Inspects *excerpt* (e.g., protocol/ports) to identify socket
- Exception: not sent if problem packet is ICMP
  - And just for fragment 0 of a group of fragments

Types of Control Messages

- **Need Fragmentation**
  - IP packet too large for link layer, DF set
- **TTL Expired**
  - Decremented at each hop; generated if $\Rightarrow$ 0
- **Unreachable**
  - Subtypes: network / host / port
    - (who generates Port Unreachable?)
- **Source Quench**
  - Old-style signal asking sender to slow down
- **Redirect**
  - Tells source to use a different local router

Using ICMP

- ICMP intended to tell host about network problems
  - **Diagnosis**
  - Won’t say more about this….
- Can exploit ICMP to elicit network information
  - **Discovery**
  - Will focus on this….

Discovering Network Path Properties

- *PMTU Discovery*: What is largest packet that go through the network w/o needing fragmentation?
  - Most efficient size to use
  - (Plus fragmentation can amplify loss)

- *Traceroute:*
  - What is the series of routers that a packet traverses as it travels through the network?

- *Ping:*
  - Simple RTT measurements

Ping: Echo and Reply

- ICMP includes simple "echo" functionality
  - Sending node sends an ICMP Echo Request message
  - Receiving node sends an ICMP Echo Reply

- Ping tool
  - Tests connectivity with a remote host
  - … by sending regularly spaced Echo Request
  - … and measuring delay until receiving replies

- If you have never used ping, do it tonight!
  - One of the few ways you actually "see" the network

Path MTU Discovery

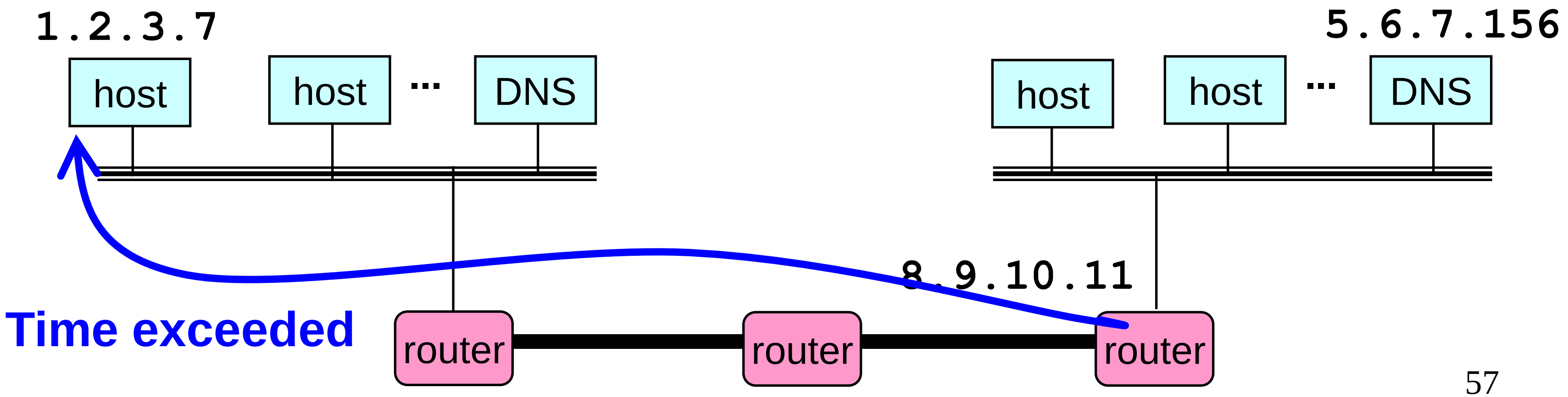
- MTU = Maximum Transmission Unit
  - Largest IP packet that a link supports
- Path MTU (PMTU) = minimum end-to-end MTU
  - Must keep datagrams no larger to avoid fragmentation
- How does the sender know the PMTU is?
- Strategy (RFC 1191):
  - **Try** a desired value
  - Set **DF** to prevent fragmentation
  - Upon receiving **Need Fragmentation** ICMP …
    - … oops, that didn't work, try a smaller value

Issues with Path MTU Discovery

- What set of values should the sender try?
  - Usual strategy: work through “likely suspects”
  - E.g., 4352 (FDDI), 1500 (Ethernet),
    1480 (IP-in-IP over Ethernet), 296 (some modems)
- What if the PMTU changes? (how could it?)
  - Sender will immediately see *reductions* in PMTU (how?)
  - Sender can periodically try larger values
- What if **Needs Fragmentation** ICMP is lost?
  - Retransmission will elicit another one
- How can **The Whole Thing Fail**?
  - “PMTU **Black Holes**”: routers that don’t send the ICMP

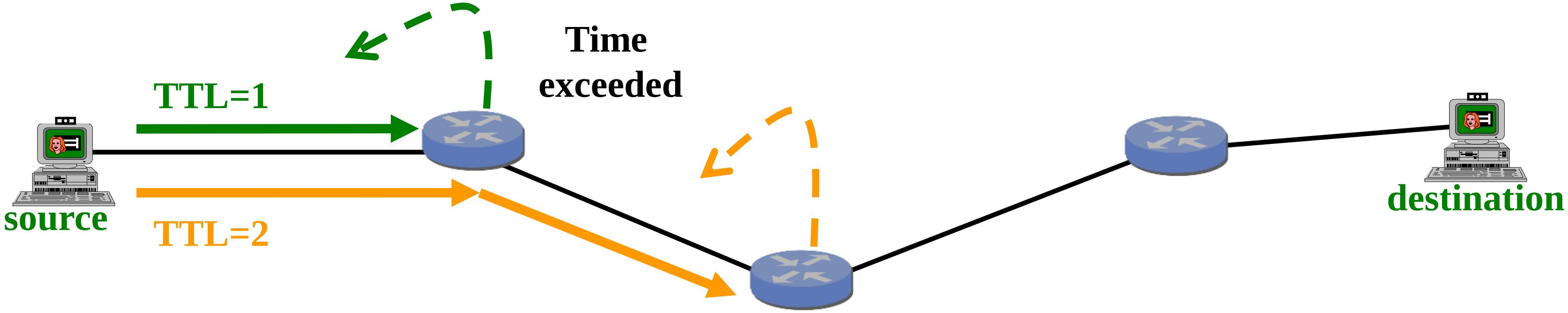
Discovering Routing via *Time Exceeded*

- Host sends an IP packet
  - Each router decrements the time-to-live field
- If **TTL** reaches 0
  - Router sends **Time Exceeded** ICMP back to the source
  - Message identifies router sending it
    - o Since ICMP is sent using IP, it's just the IP source address
    - o And can use PTR record to find name of router



Traceroute: Exploiting *Time Exceeded*

- Time-To-Live field in IP packet header
  - Source sends a packet with TTL ranging from ***1*** to ***n***
  - Each router along the path decrements the TTL
  - "TTL exceeded" sent when TTL reaches *0*
- *Traceroute* tool exploits this TTL behavior



Send packets with TTL=1, 2, …
and record source of *Time Exceeded* message

traceroute to www.whitehouse.gov (204.102.114.49),
30 hops max, 40 byte packets

traceroute to www.whitehouse.gov (204.102.114.49),
30 hops max, 40 byte packets

1 cory115-1-gw.EECS.Berkeley.EDU (128.32.48.1)
0.829 ms 0.660 ms 0.565 ms

traceroute to www.whitehouse.gov (204.102.114.49),
30 hops max, 40 byte packets

```
 1 cory115-1-gw.EECS.Berkeley.EDU (128.32.48.1)
   0.829 ms 0.660 ms 0.565 ms
 2 cory-cr-1-1-soda-cr-1-2.EECS.Berkeley.EDU (169.229.59.233)
   0.953 ms 0.857 ms 0.727 ms
```

```
traceroute to www.whitehouse.gov (204.102.114.49),
   30 hops max, 40 byte packets
 1 cory115-1-gw.EECS.Berkeley.EDU (128.32.48.1)
   0.829 ms 0.660 ms 0.565 ms
 2 cory-cr-1-1-soda-cr-1-2.EECS.Berkeley.EDU (169.229.59.233)
   0.953 ms 0.857 ms 0.727 ms
 3 soda-cr-1-1-soda-br-6-2.EECS.Berkeley.EDU (169.229.59.225)
   1.461 ms 1.260 ms 1.137 ms
 4 g3-8.inr-202-reccev.Berkeley.EDU (128.32.255.169)
   1.402 ms 1.298 ms *
 5 ge-1-3-0.inr-002-reccev.Berkeley.EDU (128.32.0.38)
   1.428 ms 1.889 ms 1.378 ms
 6 oak-dc2--ucb-ge.cenic.net (137.164.23.29)
   1.731 ms 1.643 ms 1.680 ms
 7 dc-oak-dc1--oak-dc2-p2p-2.cenic.net (137.164.22.194)
   3.045 ms 1.640 ms 1.630 ms
 8 * * *
 9 dc-lax-dc1--sac-dc1-pos.cenic.net (137.164.22.126)
   13.104 ms 13.163 ms 12.988 ms
10 137.164.22.21 (137.164.22.21)
   13.328 ms 42.981 ms 13.548 ms
11 dc-tus-dc1--lax-dc2-pos.cenic.net (137.164.22.43)
   18.775 ms 17.469 ms 21.652 ms
12 a204-102-114-49.deploy.akamaitechnologies.com (204.102.114.49)
   18.137 ms 14.905 ms 19.730 ms
```

Lost Reply

Router doesn't send ICMPs

No PTR record for address

Final Hop