EECS 489 Discussion 9

Announcements

- Project 3 due this week
- Project 4 to be released soon

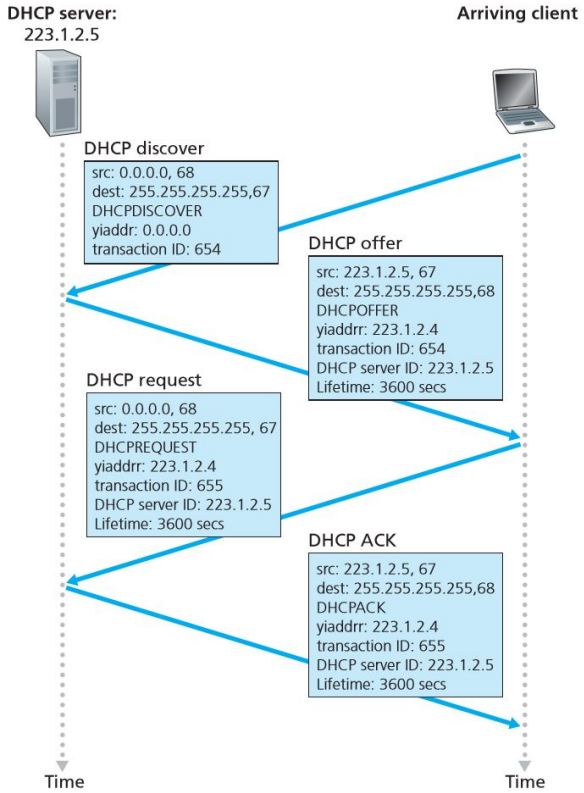
Agenda

- DHCP
- ARP
- IP Forwarding
- Calculating IP checksum
- ICMP
- Traceroute
- Project 4

DHCP

- Dynamic Host Configuration Protocol
- Used to answer “how do I get my IP address”
- Protocol steps:
  - DHCP server discovery: Broadcast to try and find DHCP server
  - DHCP server offer: Broadcast with available IP
  - DHCP request: respond to server offer
  - DHCP ACK

DHCP


DHCP server:
223.1.2.5
Arriving client
DHCP discover
src: 0.0.0.0, 68
dest: 255.255.255.255,67
DHCPDISCOVER
yiaddr: 0.0.0.0
transaction ID: 654
DHCP offer
src: 223.1.2.5, 67
dest: 255.255.255.255,68
DHCPOFFER
yiaddrr: 223.1.2.4
transaction ID: 654
DHCP server ID: 223.1.2.5
Lifetime: 3600 secs
DHCP request
src: 0.0.0.0, 68
dest: 255.255.255.255, 67
DHCPREQUEST
yiaddrr: 223.1.2.4
transaction ID: 655
DHCP server ID: 223.1.2.5
Lifetime: 3600 secs
DHCP ACK
src: 223.1.2.5, 67
dest: 255.255.255.255,68
DHCPACK
yiaddrr: 223.1.2.4
transaction ID: 655
DHCP server ID: 223.1.2.5
Lifetime: 3600 secs
Time
Time

ARP

- “Given a node’s IP, how do I find it’s MAC address?”
- Maintain ARP table: IP to MAC table
- Suppose A wants to learn B’s MAC address
  - A broadcasts ARP query packet (contains B’s IP address)
    - All machines on LAN get ARP query
    - Query packet contains A’s MAC and IP address
  - B generates ARP reply
    - Frame sent to A’s MAC address (unicast)

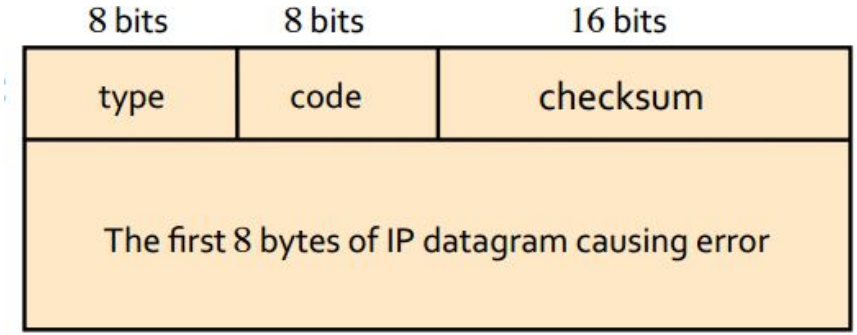
ARP Spoofing

“What stops someone from claiming an IP address they don’t have”

- Not really a fully solved problem
- Static ARP table
- Have ARP table coordinate more with the DHCP servers

ICMP (Internet Control Message Protocol)

- Used by hosts and routers to communicate network-level information
  - Error reporting: unreachable host, network
  - Echo request/reply (used by ping)
  - ICMP error message does not trigger another ICMP message
- ICMP runs on network-layer, but above IP
- ICMP message format



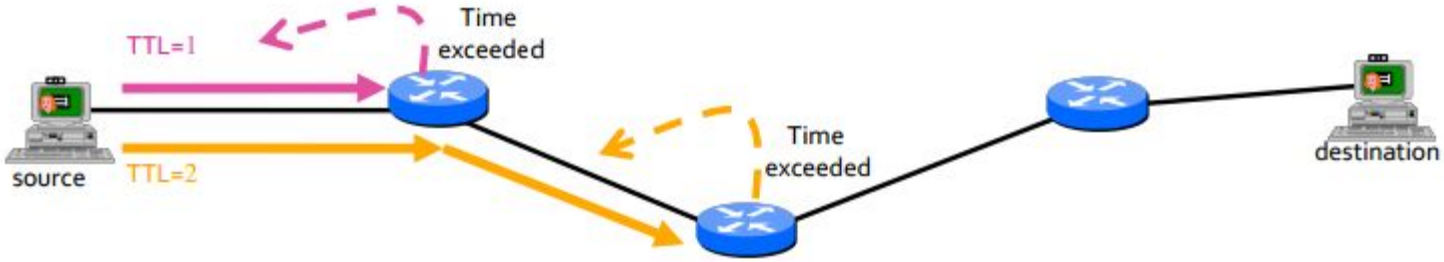
ICMP

Type	Code	Description
0	0	echo reply (ping)
3	0	dest network unreachable
3	1	dest host unreachable
3	2	dest protocol unreachable
3	3	dest port unreachable
3	4	frag needed but DF set
3	6	dest network unknown
3	7	dest host unknown
8	0	echo request (ping)
9	0	route advertisement
10	0	router discovery
11	0	TTL expired
12	0	bad IP header

Simplified Traceroute

- How to discover the routers along a path
- Source sends a series of UDP packets to destination
  - First 3 packets have TTL set to 1
  - Next 3 packets have TTL set to 2, etc.
  - Packets are all sent to an unused port number
- When packet's TTL expired:
  - Router discards packet
  - Send to source an ICMP message
  - Message include IP address of router
  - Used to get RTT estimate

Simplified Traceroute



Calculating IP Checksum

- Treat data as a sequence of 16-bit integers and computes the 1’s complement of their sum
- When adding numbers, a carryout from the most significant bit is added back to the result
- Suppose we have the following IP header
  - 4500 003c 1c46 4000 4006 b1e6 ac10 0a63 ac10 0a0c
  - b1e6 is the checksum

IP Checksum example

Given: `4500 003c 1c46 4000 4006 b1e6 ac10 0a63 ac10 0a0c`

Compute sum:

```
4500 + 003c = 453c

453c + 1c46 = 6182

6182 + 4000 = A182

A182 + 4006 = E188

E188 + AC10 = 18D98 + 1 = 8D99
```

IP Checksum example

Given: `4500 003c 1c46 4000 4006 b1e6 ac10 0a63 ac10 0a0c`

Compute sum:

```
8D99 + 0A63 = 97FC

97FC + AC10 = 1440C + 1 = 440D

440D + 0A0C = 4E19

~(4E19) = B1E6 = Checksum!
```

Source

Checksum: Disadvantage

With 16-bit checksum, not all corrupted packets will be detected

Data Item In Binary	Checksum Value	Data Item In Binary	Checksum Value
00001	1	00011	3
00010	2	00000	0
00011	3	00001	1
00001	1	00011	3
totals	7		7

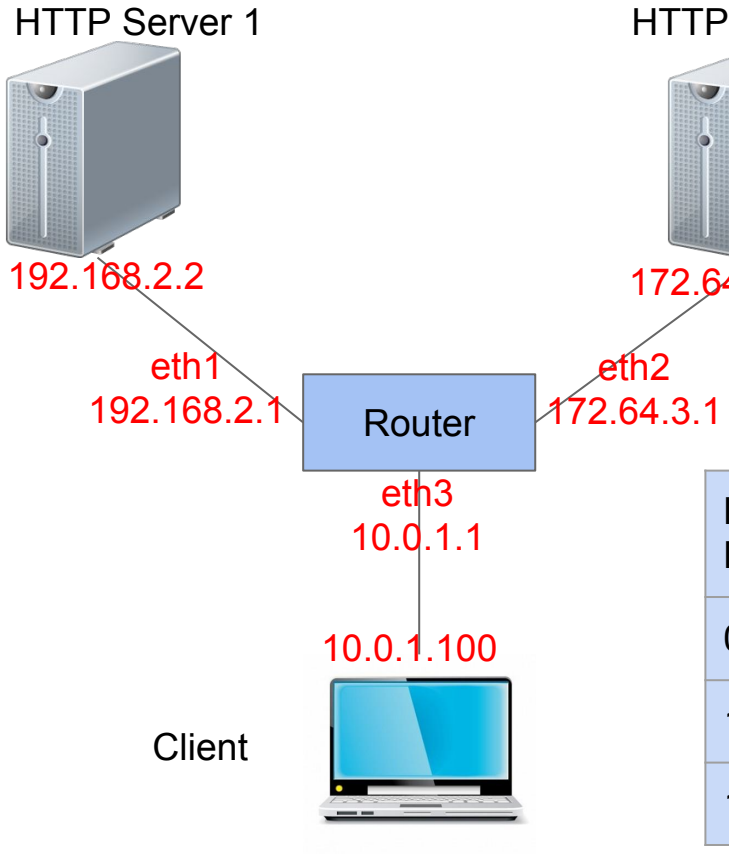
Project 4 Overview

- You are going to write a “simple router”
- Given a static network topology
- Given a static routing table
- You are responsible for writing the logic to handle Ethernet frames
  - Forward them
  - Generate ICMP message
  - Drop it
  - And more...

Project 4 Overview

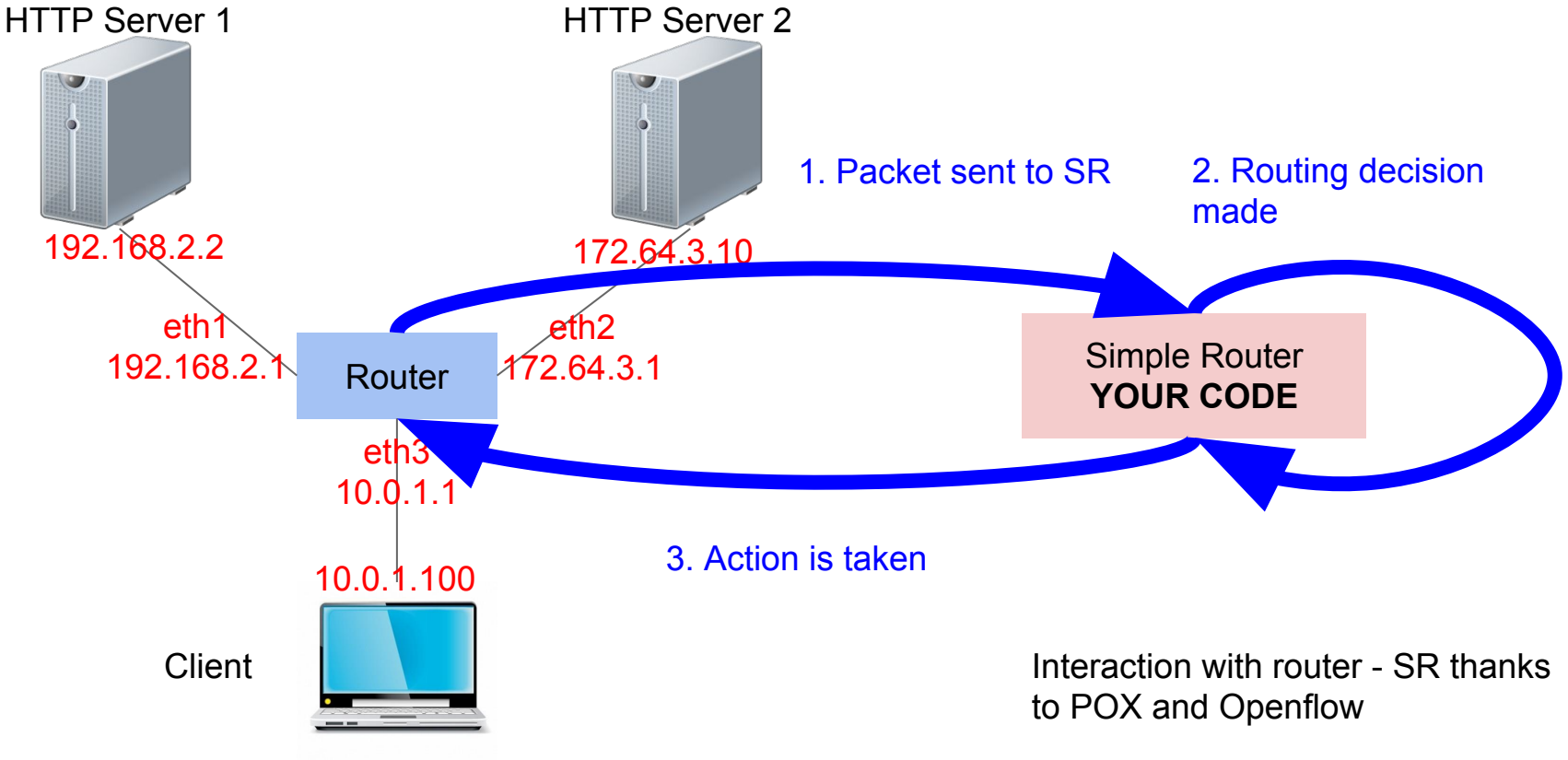
- Network topology emulated with mininet: your router connects 2 servers and a client
- Your router will handle real traffic

Project 4 Environment



Destination Network	Gateway	Network Mask	Outgoing Iface
0.0.0.0	10.0.1.100	0.0.0.0	eth3
192.168.2.2	192.168.2.2	255.255.255.255	eth1
172.64.3.10	172.64.3.10	255.255.255.255	eth2

Project 4 Environment



What Does My Code Need To Do

- Route Ethernet frames
- Handle ARP requests and replies
- Handle traceroutes
- Handle TCP/UDP packets sent to one of the router’s interfaces
- Respond to ICMP echo requests
- Maintain an ARP cache
- And more (see spec)

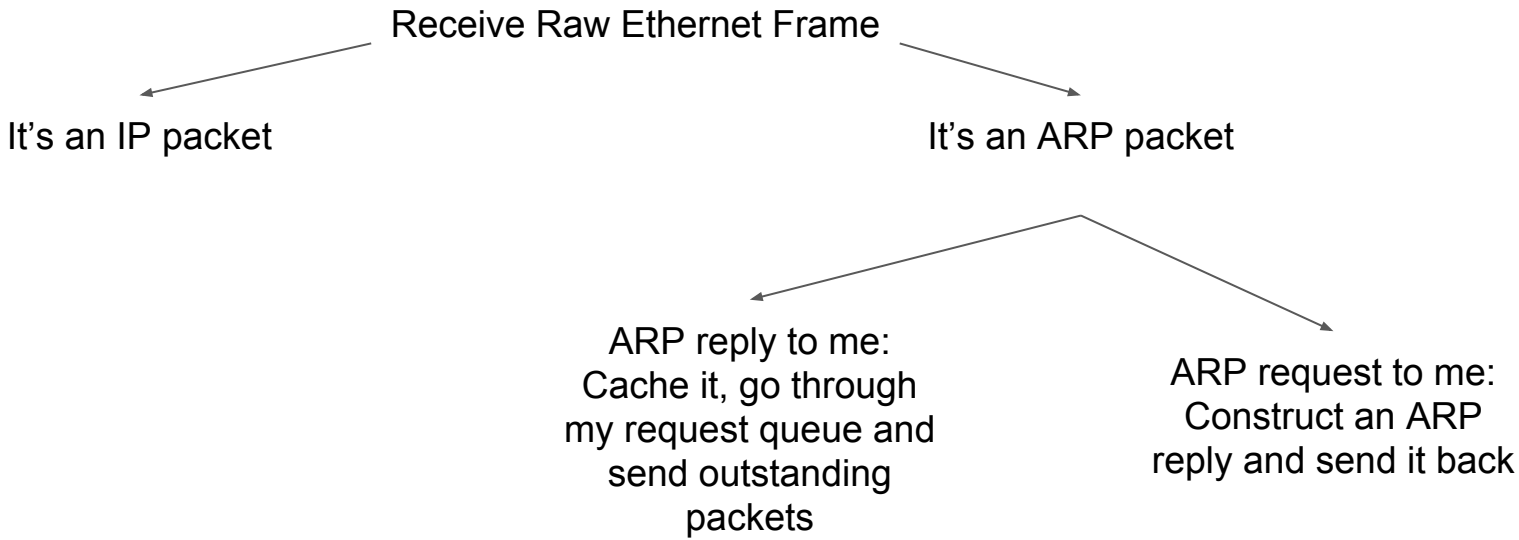
Project 4 FAQ

Can I modify the source code? Yes

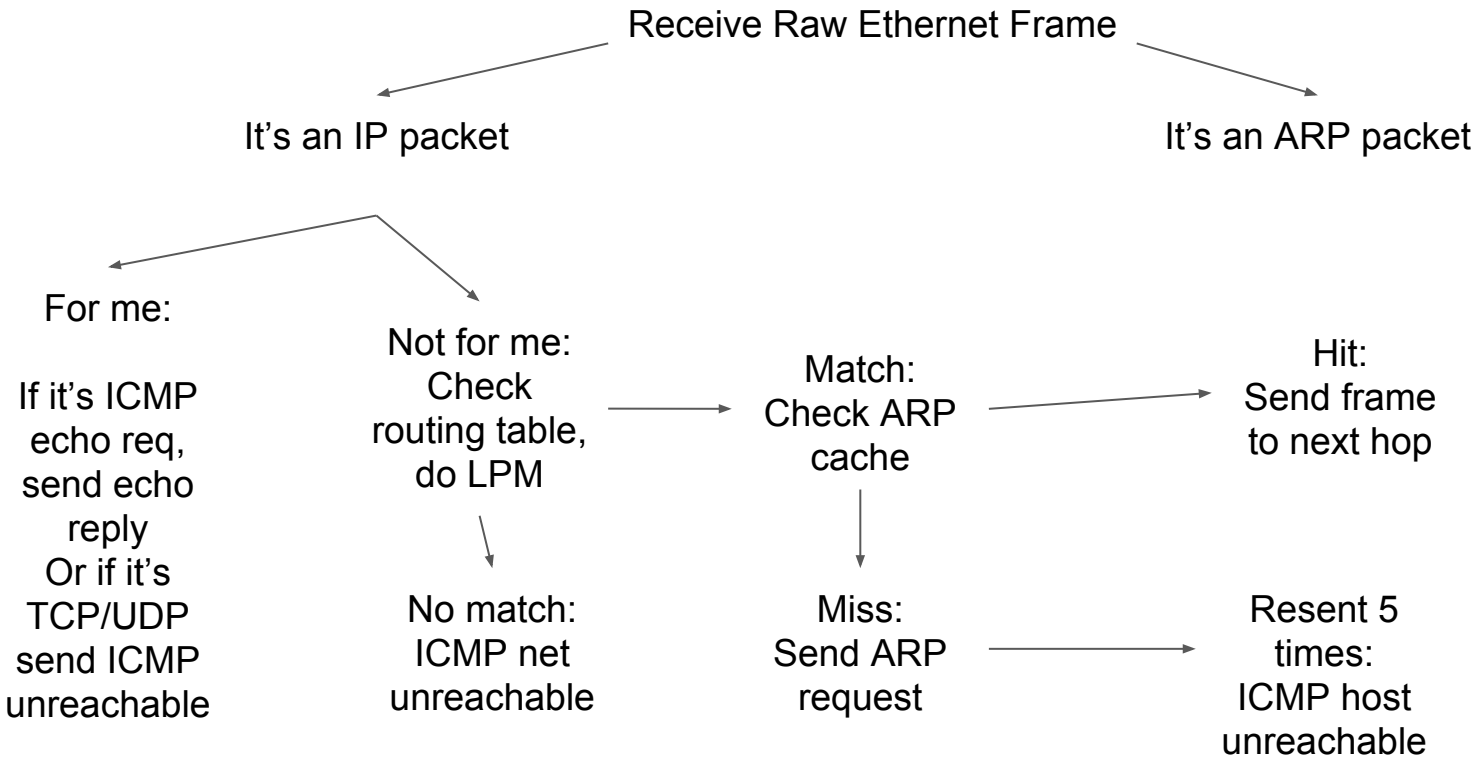
Can I use C++? Yes, but might not be as helpful as you think

Does my program have to EXACTLY match the reference implementation? No, not necessarily. As long as you still do ARP caching and the end commands still work

Rough Flow Chart



Rough Flow Chart



Helpful Tips

- Use wireshark (run -l <logfile.pcap>) with your program to log packets
- Compare wireshark output with reference output if your packets are getting rejected
- Don’t forget to use htonl, htons, ntohs, ntohl
- We have debug functions setup in sr_utils.c
  - Print_hdrs, print_addr_ip_int
- Test your program with things like ping, traceroute, wget

Working router demo

Acknowledgements

http://web.eecs.umich.edu/~sugih/courses/eecs489/lectures/06-IPv4+CIDR+ICMP.pdf

http://web.eecs.umich.edu/~sugih/courses/eecs489/lectures/21-Physical+ARP+DHCP.pdf

Stanford's CS144 Staff