CS 640 Introduction to Computer Networks

Lab 3 Overview

Due: Friday November 04, 2022

Lab 3 Overview

For this lab, you will modify your virtual router to:

1. Generate Internet Control Messaging Protocol (ICMP) messages when error conditions occur.
2. Populate the ARP cache by generating and consuming Address Resolution Protocol (ARP) messages.
3. Build a routing table using distance vector routing. With these changes, your virtual router will no longer depend on a static ARP cache or static route table, and it will be pingable and traceable.

Sources

ARP: P&D Chapter 3.3.6, rfc6747

ICMP: P&D Chapter 3.3.8, rfc792

RIP: P&D Chapter 3.4 2, rfc2453

Lab requirements and implementation details

- Lab3 description, FAQ, Lab3 slide

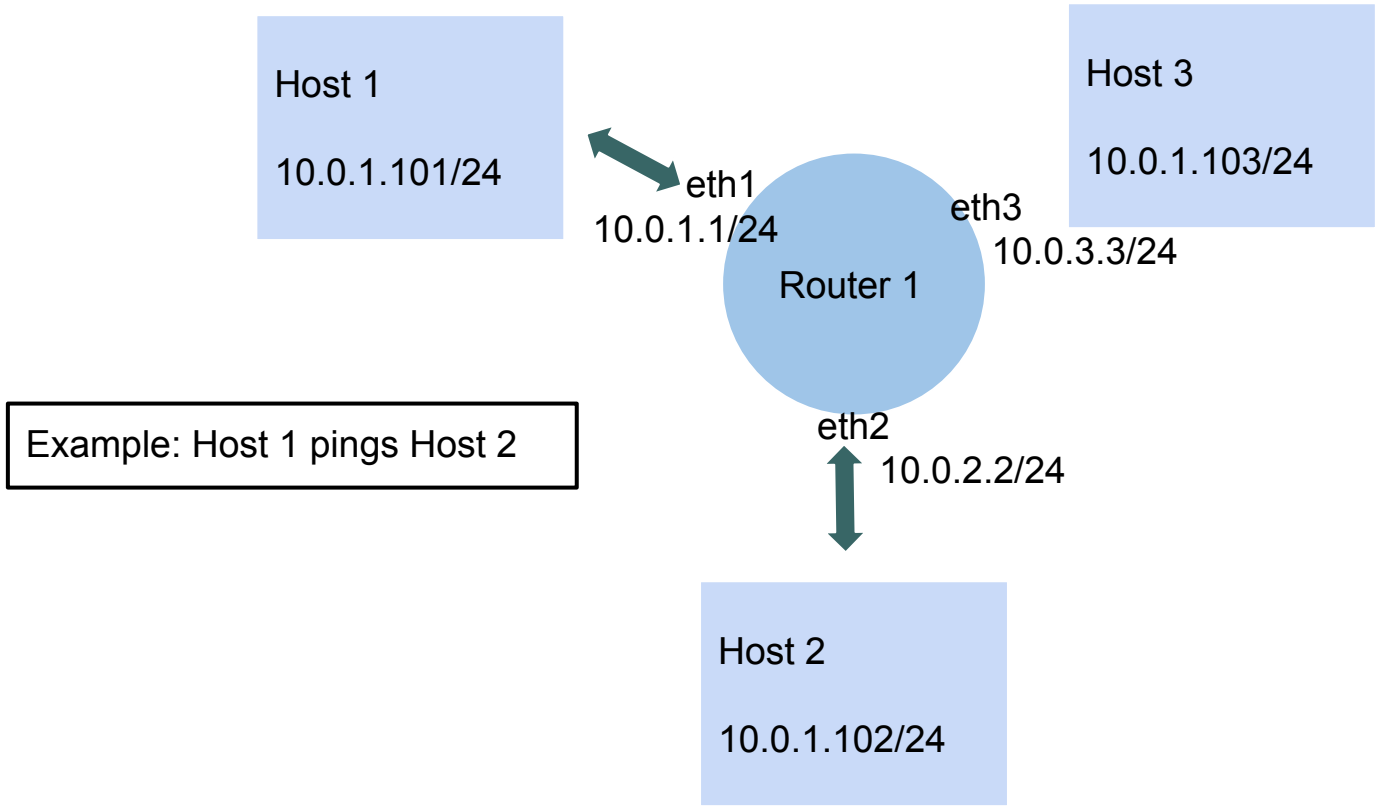
Learning Outcomes

After completing this lab, you should be able to:

- Write code that constructs and deconstructs packets containing multiple layers of protocols
- Explain how the Address Resolution Protocol (ARP) and distance vector (DV) routing, and ICMP work

Simple ARP Example

Start the Router 1 with static Route Table and ARP Cache



Simple ARP Example

Host1 pings Host2:

1. Host 1 sends an ARP request to the Router to get the MAC Addr of eth1
   1.1. eth1 - One of the interfaces of the router connected to Host1
2. Router sends ARP reply (10.0.1.1 is-at 1a:37:ca:0d:50:74)
3. Host 1 sends an ICMP echo request to Host 2 (10.0.1.101 > 10.0.2.102)
   3.1. With MAC Addr of eth1
4. Request received at Router, it forwards the request to Host 2 from eth2
5. Host 2 sends an ARP request to the Router to get MAC Addr of eth2
6. Router sends ARP reply (10.0.2.2 is-at 9e:2e:97:14:9c:45)
7. Now, Host 2 sends an ICMP echo reply to Host1 (10.0.2.102 > 10.0.1.101)
   7.1. With MAC Addr of eth2
8. Reply received at Router, it forwards it to Host1 - Ping completed

ICMP

1. **ICMP echo request/reply** example in the single_rt topology: h1 ping -c 1 10.0.1.1

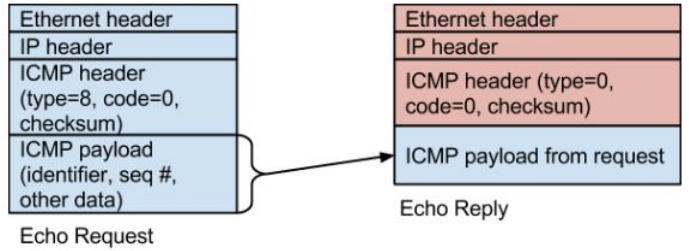
   When the router receives an ICMP request **destined for one of its interfaces**, send an ICMP reply

   00:00:00:00:00:01 > 1a:37:ca:0d:50:74, ethertype IPv4 (0x0800)

   10.0.1.101 > 10.0.1.1: ICMP echo request, id 19016, seq 1, length 64

   1a:37:ca:0d:50:74 > 00:00:00:00:00:01, ethertype IPv4 (0x0800)

   10.0.1.1 > 10.0.1.101: ICMP echo reply, id 19016, seq 1, length 64



2. **ICMP destination net unreachable** example in the single_rt topology: h1 ping -c 1 10.1.1.1

   This message must be sent if there is **no matching entry in the route table** when forwarding an IP packet

   00:00:00:00:00:01 > 1a:37:ca:0d:50:74, ethertype IPv4 (0x0800),(proto ICMP (1))

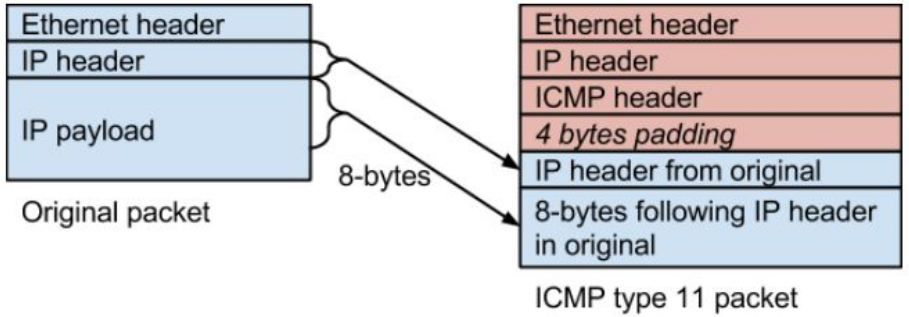
   10.0.1.101 > 10.1.1.1: ICMP echo request, id 21718, seq 1

   1a:37:ca:0d:50:74 > 00:00:00:00:00:01, ethertype IPv4 (0x0800), (proto ICMP (1))

   10.0.1.1 > 10.0.1.101: ICMP net 10.1.1.1 unreachable

   (tos 0x0, ttl 63, id 61570, offset 0, flags [DF], proto ICMP (1))

   10.0.1.101 > 10.1.1.1: ICMP echo request, id 21718, seq 1



ICMP

3. ICMP destination host unreachable

This message should be sent if the **MAC address associated with an IP address cannot be resolved using ARP**

example in the single_rt topology:

h1 ping -c 1 10.0.2.3

eth1:

00:00:00:00:00:01 > 86:95:d3:01:6f:e8, ethertype IPv4 (0x0800),(proto ICMP (1))

10.0.1.101 > 10.0.2.3: ICMP echo request, id 26640, seq 1, length 64

--------drop packet in the queue--------

86:95:d3:01:6f:e8 > 00:00:00:00:00:01, ethertype IPv4 (0x0800),(proto ICMP (1))

10.0.1.1 > 10.0.1.101: ICMP host 10.0.2.3 unreachable, length 36

(tos 0x0, ttl 63, id 10948, offset 0, flags [DF], proto ICMP (1), length 84)

10.0.1.101 > 10.0.2.3: ICMP echo request, id 26640, seq 1, length 64

eth2: Received three ARP requests:

76:15:8a:37:a1:52 > ff:ff:ff:ff:ff:ff Request who-has 10.0.2.3 tell 10.0.2.2

ICMP

4. ICMP time exceeded

Your router should decrement the TTL field by 1 and checks if the field (after decrement) equals 0, and generate an ICMP time exceeded message prior to dropping the packet whose TTL field is 0.

h1 traceroute -n 10.0.2.102

Use traceroute to test: traceroute send packets with ttl 1,2,...,n (upperlimit)

10.0.1.101.33909 > 10.0.2.102.33434 ttl 1

10.0.1.101.38099 > 10.0.2.102.33436 ttl 2

...

86:95:d3:01:6f:e8 (eth1's mac addr) > 00:00:00:00:00:01, ethertype IPv4,(proto ICMP (1))

10.0.1.1 > 10.0.1.101: ICMP time exceeded in-transit, length 36

10.0.1.101.33909 > 10.0.2.102.33434: UDP, length 32

ICMP

5. ICMP destination port unreachable

This message should be sent if your router receives a TCP or UDP packet destined for one of its interfaces

in single_rt: h1 wget 10.0.1.1

Routing Information Protocol

1. Your router should send a RIP request out of all of the router's interfaces when RIP is initialized.

2. Your router should send an unsolicited RIP response out all of the router's interfaces every 10 seconds thereafter.

3. Time out route table entries for which an update has not been received for more than 30 seconds.

Example when you listen to r1-eth3 for topology: pair_rt: we start r1 and then r2

R1 - eth3 - R2

tcpdump: **listening on r1-eth3**

66:f4:c9:7c:b6:5e > ff:ff:ff:ff:ff:ff

10.0.100.1.520 > 224.0.0.9.520

RIPv2, **Request**

66:f4:c9:7c:b6:5e > ff:ff:ff:ff:ff:ff, ethertype IPv4 (0x0800), length 106: (tos 0x0, ttl 15, id 0, offset 0, flags [none], proto UDP (17), length 92)

10.0.100.1.520 > 224.0.0.9.520

RIPv2, **Response**, length: 64, routes: 3 or less

AFI IPv4, 10.0.100.0/24, metric: 0, next-hop: self

AFI IPv4, 10.0.2.0/24, metric: 0, next-hop: self

AFI IPv4, 10.0.1.0/24, metric: 0, next-hop: self

d2:00:28:92:fc:e2 > ff:ff:ff:ff:ff:ff, ethertype IPv4 (0x0800)

10.0.100.2.520 > 224.0.0.9.520

RIPv2, **Request**, length: 4, routes: 0 or less

d2:00:28:92:fc:e2 > ff:ff:ff:ff:ff:ff, ethertype IPv4 (0x0800)

10.0.100.2.520 > 224.0.0.9.520

RIPv2, **Response**, length: 64, routes: 3 or less

AFI IPv4, 10.0.100.0/24, metric: 0, next-hop: self

AFI IPv4, 10.0.4.0/24, metric: 0, next-hop: self

AFI IPv4, 10.0.3.0/24, metric: 0, next-hop: self

RIP

66:f4:c9:7c:b6:5e > d2:00:28:92:fc:e2, RIP solicited response from 5e to e2: when sending a RIP response for a specific RIP request, the destination IP address and destination Ethernet address should be the IP address and MAC address of the router interface that sent the request.

10.0.100.1.520 > 10.0.100.2.520

RIPv2, **Response**, length: 64, routes: 3 or less

AFI IPv4, 10.0.100.0/24, metric: 0, next-hop: self

AFI IPv4, 10.0.2.0/24, metric: 0, next-hop: self

AFI IPv4, 10.0.1.0/24, metric: 0, next-hop: self

66:f4:c9:7c:b6:5e > ff:ff:ff:ff:ff:ff

10.0.100.1.520 > 224.0.0.9.520

RIPv2, **Response**, length: 104, routes: 5 or less

AFI IPv4, 10.0.100.0/24, metric: 0, next-hop: self

AFI IPv4, 10.0.2.0/24, metric: 0, next-hop: self

AFI IPv4, 10.0.1.0/24, metric: 0, next-hop: self

AFI IPv4, 10.0.4.0/24, metric: 1, next-hop: 10.0.100.2

AFI IPv4, 10.0.3.0/24, metric: 1, next-hop: 10.0.100.2

d2:00:28:92:fc:e2 > ff:ff:ff:ff:ff:ff

10.0.100.2.520 > 224.0.0.9.520

RIPv2, **Response**, length: 104, routes: 5 or less

AFI IPv4, 10.0.100.0/24, metric: 0, next-hop: self

AFI IPv4, 10.0.4.0/24, metric: 0, next-hop: self

AFI IPv4, 10.0.3.0/24, metric: 0, next-hop: self

AFI IPv4, 10.0.2.0/24, metric: 1, next-hop: 10.0.100.1

AFI IPv4, 10.0.1.0/24, metric: 1, next-hop: 10.0.100.1

Implementation Details - Router.java

handlePacket()

- Check type: call *handleIpPacket()* or *handleArpPacket()*

handleIpPacket()

- Checks TTL, if TTL = 0
  - Generate ICMP - Time exceeded
- Check RIP - UDP Protocol and (Destination IP is one of the interfaces of the router or Destination IP is 224.0.0.9) and if PORT is UDP.RIP_PORT(520)
  - If yes, *handleRipPacket()*
- Check if packet is destined for one of the router interfaces
  - If Protocol is TCP or UDP
    - Generate ICMP - Destination port unreachable
  - Else If Protocol is ICMP
    - Generate ICMP - Echo Reply
- Else *forwardIpPacket()*

Implementation Details - Router.java

forwardIpPacket()

- Check If any entry match in the route table, if not:
  - Generate ICMP - Destination net unreachable
- Lookup the MAC address in the ARP Cache,
  - If no entry, send ARP request.
  - **Hint: Use Concurrent HashMap to store map between ip and thread**
- Else forward the packet

handleRipPacket()

- If type is request
  - Send a response back to source - all entries in the route table
- Else
  - Update the route table - insert if new entry or update existing route entry if cost is lower
    - Hint: need cost field in RouteEntry.java
  - If route entries updated - Send RIP packet to all interfaces on neighbours (broadcast)

Implementation Details - Router.java

handleArpPacket()

- if ARP request and target IP == interface IP
  - Create and send ARP reply
- Else if ARP reply
  - Update ARP Cache - insert <sender's mac, sender's ip>

Hint: Use concurrent data structures and classes (Thread class) for sending ARP Requests

Rubric: Submission

Late policy:

- Up to 30 minutes late – lose 0% points
- Upto 24 hours late — lose 10% of points
- Upto 48 hours late — lose 30% of points
- Upto 72 hours late — lose 60% of points
- Beyond 72 hours — lose 100% of points

Description	**Points**	**Explanation**
Format correct	1	Files submitted as specified in the lab description.
Documentation in code	2	Useful comments throughout code.
Code compiles	2	Code compiles with no help from TA.

Rubric: Part 2 ICMP

Description	Points	Test commands
Time to live Exceeded	3	ping ran with count and time set
Destination net unreachable	3	ping ran with count set to unreachable net
Destination host unreachable	3	ping ran with count set to unreachable host
Destination host reachable	2	ping ran with no packet loss
Destination port unreachable	3	nc ran with connection refused output
Echo reply implemented	3	Checked manually by TA
Same subnet is reachable	2	ping ran in same subnet with 0% loss
Other subnet is reachable	2	ping ran in different subnet with 0% loss
All valid IPs in linear topology are reachable	3	traceroute shows all IPs

Rubric: Part 3 ARP

Description	Points	Test commands
Generates ARP replies	3	tcpdump output shows replies
Generates ARP requests	3	tcpdump output shows requests
Receives ARP replies	3	tcpdump output shows replies
Packets dropped from queue after 3 requests + 1 second	2	Checked manually by TA
ARP request unsuccessful when pinging a non-exsistent IP address	2	tcpdump output does not show requests that should not be there.
Discovers and probes hosts on a computer network	2	arping output shows search

Rubric: Part 4 RIP

Description	Points	Test commands
Entries populated when router is started	2	tcpdump output shows RIP messages
RIP packets created as UDP with destination port 520	2	Checked manually by TA along with tcpdump with output showing UDP
RIP request sent to all router interfaces	3	tcpdump showing IP and mac addresses
Route table entries are updated overtime.	3	tcpdump and ping commands
Works on pair, triangle, and linear topologies	6	2 points for each topology