Exercises: Basics of Networking – III

Experiential Learning Workshop

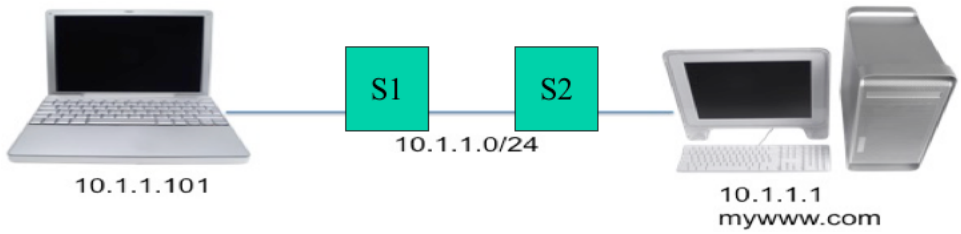
1 General Guidelines

1. Make a team of two or three unless stated otherwise.
2. For each exercise, use wireshark capture to verify contents
3. Ensure to use proper capture filter and don't capture irrelevant traffic
4. Where appropriate or applicable, use `wget` or `nc` to access the web server.
5. The default client for accessing web server is assumed to be browser, preferably firefox. You can use Chrome or any other browse as well.
6. The webserver in the example below is taken as 'myweb.com'. Please use your hostname or corresponding IP address instead in your exercise.
7. To kill any program in the linux terminal, please press **`Ctrl-C`** and not **`Ctrl-Z`**. The latter will suspend the program and not stop it.

Note: Appendix provides instructions on installing any package if not already installed.

2 TCP Streaming, Reliability and UDP message boundary

The setup for this exercise requires two switches to be used as follows:



Note:

i. The switch S1 can be your lab network switch, and thus connect 2nd machine via a switch which in turn is connected to wall jack (lab network switch).

ii. We will use following simple python programs. You don't really need to know the python language syntax (reading of the program itself should explain what is being done).

   a. `udp_client.py`
   b. `udp_server.py`
   c. `tcp_client.py`
   d. `tcp_server.py`

i. Use the option -h to understand the usage syntax. By default, server listens on all the IP address and port number 9999 and receive in the buffer size of 10 bytes and reads data from socket every 1 second and displays the same. The client program by default connects to server on port 9999, but requires server IP to be specified, the client uses a default buffer size of 100 bytes and sends 10 messages at interval of 5 seconds each. The options for these programs are

   a. –s <server IP address>
   b. –p <server port number>
   c. –b <buffer size>

d. –c <count of packets/messages to be sent>. The first message contains sequence of A… characters, 2nd message contains sequence of B… characters, and so on.
e. –d delay (in sending by client or receiving by server)

2.1 UDP message Orientation

1. On H2, run the server `./udp_server.py` with buffer size of 20.
2. On H1, run the client `./udp_client.py` sending the message to server with buffer size of 50 bytes.
3. Note down what is displayed by the server. Explain why the server does not receive full packet. Change the buffer size of server to study further.
4. Analyze wireshark capture on what is being transmitted by client.

2.2 UDP Packet Loss

1. Repeat the above exercise with client sending 10 packets at interval of 5 seconds.
2. Break the link between switch S1 and S2 at 12th seconds and restore this link at 26th second. To break the link, just remove the wire between 2 switches or between switch and the wall jack.
3. Using wireshark, analyze the packet sent by client as well as those received by server. Explain which packets are received and which are lost.

2.3 TCP Stream based transmission

1. Repeat the above exercise but with tcp client and servers.
2. Analyze if the server receives full data.
3. Explain and understand the difference between TCP and UDP behavior.

2.4 TCP Packet Loss and Recovery

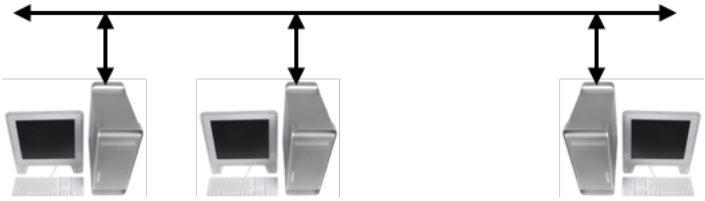
1. Repeat the above exercise.
2. Analyze if the server receives full data i.e. data transmitted during the time when link between S1 and S2 is broken/disconnected.
3. Analyze how many TCP segments were transmitted vs how many received.
4. Analyze if retransmitted segment contained any new data?
5. Analyze the time difference between retransmitted segments.
6. Explain and understand the difference between TCP and UDP behavior.

2.5 Exercise Expectations

1. Ability to understand UDP message delivery being boundary oriented.
2. Ability to understand TCP based message streaming i.e. no message boundaries.
3. Ability to under packet loss recovery in TCP whereas there is no recovery in UDP packet loss.

3 Understanding ARP

3.1 Study ARP table maintenance



1. Setup three machines A, B, and C as shown in (4.1 **ICMP Redirect**). In this exercise we do not need 2nd and 3rd network i.e,. 10.x.2.0/24 and 10.x.3.0/24 and thus these addresses need not be assigned. Use of separate small switch is not recommended for this exercise

2. Study the existing ARP table and identify which machines are these.
   a. `arp -an`

3. Ping one live machine (i.e. machine which is reachable) and one unreachable machine. Study the ARP table. What does the entry show for unreachable machine.

4. Ping the broadcast address of your network. This requires sudo privilege.
   a. `sudo ping -b -c2  10.x.1.255`

5. Analyze in wireshark on number of ICMP response received as well as number of entries added in ARP Table.

3.2 Study Gratuitous ARP

1. Use the same setup as above. Note down MAC Address of 3 machines (A, B & C).

2. Install `arping` on the machine A (and may be others)
   a. `sudo apt install arping`

3. On A, add the IP address of A to that of C (replace eth0 with applicable interface name)
   a. `sudo ip addr add 10.x.1.201/24 dev eth0`

4. Issue arping with this new address to B
   a. `sudo arping -s 10.x.1.201 -c 2 10.x.1.101`

5. Study the ARP table at B. It should show updated MAC address of A for IP Address of C.

3.3 Exercise Expectations

1. Able to understand the use of ARP and Gratuituous ARP.

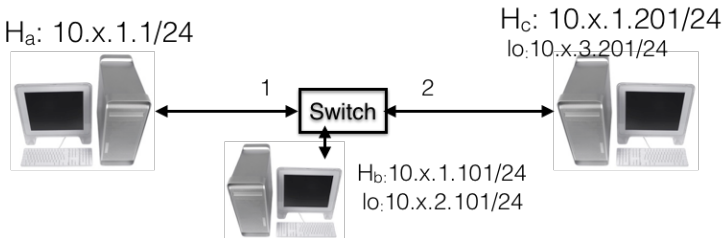
4 ICMP Errors

Note:

- The IP addresses specified in the below example are for illustrative purposes only. Use the IP Address assigned to your machines in the network.
- For this exercise you need to make a group of 3.

4.1 ICMP Redirect

1. Connect 3 m/cs in a network as shown in diagram. Assign the address manually to these systems Use the appropriate mask. Use value of x as per your team number e.g. 1 for team 1, 2 for team 2 and so on. The exercise can work without the use of small 4/8 port switch separately i.e. this exercise can be carried out in regular lab network.



2. On B and C, assign addresses from a different network on the loopback (`lo`) interface?

   a. `sudo ip addr add 10.x.2.101/24 dev lo` on B
   b. `sudo ip addr add 10.x.3.201/24 dev lo` on C

3. Disable acceptance of ICMP redirect on A to ensure that its routing table is not updated as per ICMP redirect. It is assumed that ethernet interface on the machine is `eth0` in the examples below. Replace it with appropriate values.

   a. `sudo sysctl -w net.ipv4.conf.all.accept_redirects=0`
   b. `sudo sysctl -w net.ipv4.conf.default.accept_redirects=0`
   c. `sudo sysctl -w net.ipv4.conf.eth0.accept_redirects=0`

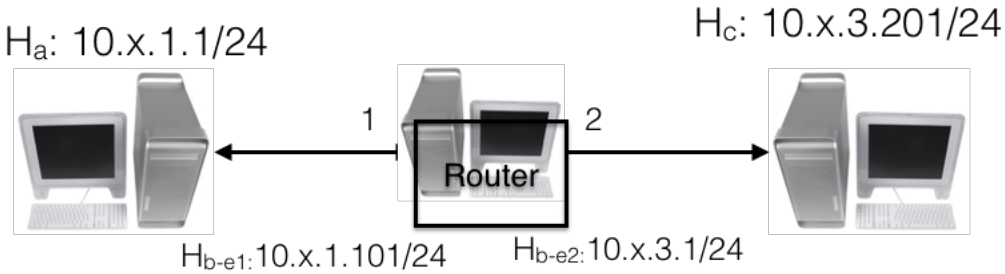
4. Add mis-redirected (incorrect) routing entries on A.

   a. `sudo ip route add 10.x.2.0/24 via 10.x.1.201`
   b. `sudo ip route add 10.x.3.0/24 via 10.x.1.101`

5. Ping from A to addresses `10.x.2.101` and `10.x.3.201`

6. Analyze in wireshark the icmp redirect packets. Use the capture filter `icmp`.

4.2 Basic Routing



1. Connect 3 m/cs in a network as shown in diagram. Assign the address manually to these systems Use the appropriate mask. Use value of x as per your team number e.g. 1 for team 1, 2 for team 2 and so on. To get a second interface on B, use the USB to ethernet adaptor on B.

2. Define IP addresses for B. Assign the IP addresses via the network Setting UI or via the following command.
   a. `sudo ip addr add 10.x.1.101/24 dev eno1`
   b. `sudo ip addr add 10.x.3.1/24 dev enx...` (USB device)

3. Convert machine B into a router
   a. `sudo sysctl -w net.ipv4.ip_forward=1`

4. Define routing for network 10.1.3.0/24 on **Ha**.
   a. `sudo ip route add 10.x.3.0/24 via 10.x.1.101`

5. Define routing for network 10.1.1.0./24 on **Hc**
   a. `sudo ip route add 10.x.1.0/24 via 10.x.3.1`

6. Check reachability between **Ha** and **Hc** (use ping to check)

4.3 PMTU Discovery

1. Connect 3 m/cs in a network as in above diagram for basic routing. Assign the address manually to these systems Use the appropriate mask.

2. Change the MTU on Router (Hb) for link between Hb and Hc.
   a. `sudo ip link set mtu 1000 dev` <USB/ethernet interface)

3. Define appropriate routing on A to reach C
   a. `sudo ip route add 10.x.3.0/24 via 10.x.1.101`

4. Define appropriate routing on C to reach back A
   a. `sudo ip route add 10.x.1.0/24 via 10.x.3.1`

5. Send a ping packet bigger than 1000 bytes from A to C. Run following on A
   a. `ping -c1 -s 1200 -p 50515253 10.x.3.201`

6. Analyze wireshark capture to study ICMP error (fragmentation needed) and subsequent packet fragmentation (This will be studied later). How many

packets are finally generated from A. Analyze the ping response. Are there two responses?

4.4 Proxy ARP

1. Connect 3 m/cs in a network as shown in diagram below. Assign the address manually to these systems Use the mask of /22 to ensure that both Ha and Hc are in same network.

2. Define the subnet mask for router interface to be /24 to ensure that Ha is network-1 and Hb is in network-2.

3. Enable proxy ARP on Hb (router)
   a. `sudo sysctl -w net.ipv4.conf.all.proxy_arp=1`
   b. `sudo sysctl -w net.ipv4.ip_forward=1`

4. No need to define routing entries at A and C since they are part of same network. Clear up their ARP table (sudo arp -d <IP address) for each IP Address.

5. Check reachability and check ARP table. Analyze the ARP Response in wireshark. ARP Reply should come from Hb for both Ha and Hc though both Ha and Hc do not know about this router.

4.5 TTL Expiry

1. Connect 3 m/cs in a network as in above exercise of basic routing.
2. Send a ping packet A to C with TTL=1.
3. Analyze the response with ICMP errors. Analyze how TTL Expiry works.
4. Analyze the source IP address in the ICMP error packet.

4.6 IP fragmentation

1. Connect 3 m/cs in a network as in above exercise of PMTU discovery.
2. Change the MTU size of the link to a smaller value.
3. Send a large size UDP packet i.e packet size larger than MTU
   a. `./udp_client -b 1200 -s 10.x.3.101 -p 2222 -c 1`
4. Analyze the packet transmission in wireshark and see the fragmentation.

4.7 Exercise Expectations

1. Able to understand proxy ARP, Routing, and ICMP errors and source address of packet corresponding to ICMP errors.
2. Able to understand ICMP Redirect, PMTU Discovery and TTL expired.