Show a normal arp exchange biology essay

Science, Biology

Part 2 Submission

Chapter # 4, 5, 6, 7

TCP/IPQuestion NumberMarks ObtainedLab 4. 5aLab 4. 5bExercise 5. 2Lab 6. 5Lab 7. 2Lab 7. 3

Name: Harpreet Singh

B00592696

Part 2 Submission

Lab 4. 5a: Clear the ARP cache and using tcpdump show a normal ARP exchange. Do not forget to annotate your submission with comments showing the purpose of the commands (e. g., to clear the ARP cache, listen to message/packet exchange) and the resulting exchange of information. Note that the -e switch with tcpdump will show the hardware addresses.

Purpose of the Lab:

The purpose of the lab is to show a normal ARP exchange.

Lab Setup:

1. Start tcpdump on hostC1.[student@hostC1 ~]$ sudo tcpdump -n - e -vvv arp or icmpComment :- The above command is used to observe the ARP and ICMP packets on the host C1.- n means not to convert the host address into names.-e means to print the link level header-vvv means to print the output more verbosely ie more descriptively2. Clear the ARP cache on hostC1. [student@hostC1 ~]$ sudo arp -ad10. 1. 3. 1 (10. 1. 3. 1) deletedComment :- The above command is used to delete the arp cache memory . 3. On hostC1 send a single ping request to hostC2.[student@hostC1 ~]$ ping -c1

hostC2PING hostC2. test. ca (10. 1. 3. 2): 56 data bytes64 bytes from 10. 1. 3. 2: icmp_seq= 0 ttl= 64 time= 0. 548 ms--- hostC2. test. ca ping statistics ---1 packets transmitted, 1 packets received, 0. 0% packet lossround-trip min/avg/max/stddev = 0. 548/0. 548/0. 548/0. 000 msComment :- Here we are using " ping " command to find out the connectivity to " hostC2"-c means exit after receiving count packet4. Observe the tcpdump output. (Observation output is below)5. Observe the ARP cache on hostC1 after the ping. (Observation output is below)

Results and Discussion:

Results:

ARP cache before the experiment:

[student@hostC1 ~]$ arp -ahostC1. test. ca (10. 1. 3. 1) at 00: 00: 0a: 00: 03: 01 on e0 permanent [ethernet]routerC. test. ca (10. 1. 3. 254) at 00: 00: 0a: 00: 03: fe on e0 expires in 954 seconds [ethernet]Comment :- The above commands displays the entries in cache table prior of making use of the ping command . the table contains the entry of host C1 ie its MAC address associated with the IP address.

Ping output:

[student@hostC1 ~]$ ping -c1 10. 1. 3. 2PING 10. 1. 3. 2 (10. 1. 3. 2): 56 data bytes64 bytes from 10. 1. 3. 2: icmp_seq= 0 ttl= 64 time= 0. 276 ms--- 10. 1. 3. 2 ping statistics ---1 packets transmitted, 1 packets received, 0. 0% packet lossround-trip min/avg/max/stddev = 0. 276/0. 276/0. 276/0. 000 ms

---

[student@hostC1 ~]$ ping -c1 hostC2PING hostC2. test. ca (10. 1. 3. 2): 56 data bytes64 bytes from 10. 1. 3. 2: icmp_seq= 0 ttl= 64 time= 0. 548 ms---hostC2. test. ca ping statistics ---1 packets transmitted, 1 packets received, 0. 0% packet lossround-trip min/avg/max/stddev = 0. 548/0. 548/0. 548/0. 000 msComment :- the above command displays the output of " ping " command to test the reach ability of host C2 from C1. We tried the command with both the IP address and the Host name . The result displays the Round Trip Time of the packet .

ARP cache after the ping:

[student@hostC1 ~]$ arp -ahostC1. test. ca (10. 1. 3. 1) at 00: 00: 0a: 00: 03: 01 on e0 permanent [ethernet]hostC2. test. ca (10. 1. 3. 2) at 00: 00: 0a: 00: 03: 02 on e0 expires in 1100 seconds [ethernet]routerC. test. ca (10. 1. 3. 254) at 00: 00: 0a: 00: 03: fe on e0 expires in 1100 seconds [ethernet]Comment :- After making use of the ping command the result of arp-a shows the entries of hostC2 . ie its entry has been added and the arp cache table has been updated.

tcpdump output:

tcpdump: listening on e0, link-type EN10MB (Ethernet), capture size 96 bytes(1)15: 00: 21. 415928 00: 00: 0a: 00: 03: 01 > ff: ff: ff: ff: ff: ff, ethertype ARP (0x0806), length 42: Ethernet (len 6), IPv4 (len 4), Request who-has 10. 1. 3. 2 tell 10. 1. 3. 1, length 28(2)15: 00: 21. 416143 00: 00: 0a: 00: 03: 02 > 00: 00: 0a: 00: 03: 01, ethertype ARP (0x0806), length 42: Ethernet (len 6), IPv4 (len 4), Reply 10. 1. 3. 2 is-at 00: 00: 0a: 00: 03: 02,

length 28(3)15: 00: 21. 416145 00: 00: 0a: 00: 03: 01 > 00: 00: 0a: 00: 03: 02, ethertype IPv4 (0x0800), length 98: (tos 0x0, ttl 64, id 17642, offset 0, flags [none], proto ICMP (1), length 84)10. 1. 3. 1 > 10. 1. 3. 2: ICMP echo request, id 31497, seq 0, length 64(4)15: 00: 21. 416175 00: 00: 0a: 00: 03: 02 > 00: 00: 0a: 00: 03: 01, ethertype IPv4 (0x0800), length 98: (tos 0x0, ttl 64, id 17640, offset 0, flags [none], proto ICMP (1), length 84)10. 1. 3. 2 > 10. 1. 3. 1: ICMP echo reply, id 31497, seq 0, length 64Comment :- The output of the tcpdump above how the 'ping ' operations work by making use of the " arp" request and reply(1) It is Ethernet broadcast which ask for MAC address for the " 10. 1. 3. 2" , " 0x0806" means it is a arp request or replylenght -42 consist of 14 byte Ethernet + 28 byte of ARP request/reply(2) It is a reply of the ARP request . It provides " 00: 00: 0a: 00: 03: 02" as the MAC address . This request is Unicast in nature.(3) It is a ICMP request made by using the ping command , length is 98 as it contains the 14 byte Ethernet + 20 byte of IP header + 64 byte of ICMP datagram (8byte ICMP header +56 byte of ICMP message).(4) It is a ICMP reply message to the request made in (3), we can observe how Source and Destination addresses are changed

Discussion:

This Lab helped us in observing how the " ping " operations works, starting from finding the MAC address of the destination address by using the ARP procedure and then performing the ICMP request and Reply operations . tcpdumps output shows the different length of different packet ie length 42 in ARP and length 98 in ICMP with the changes in the Sender and receiver addresses. The ping operations result in adding of MAC address host C2 in the arp cache tableConclusion: We have observed a normal ARP exchange

which results in a new ARP table entry. Lab 4. 5b: Clear the ARP cache and using tcpdump show ARP messages when trying to telnet to a non-existent host. Also observe the TCP timeout value.

Purpose of the Lab:

The purpose of the lab is to show ARP requests to a non-existent host and to observe the TCP timeout value.

Lab Setup:

Start tcpdump on hostC1. sudo tcpdump -n port discard or arp[student@hostC1 ~]$ sudo tcpdump -n -e -v arpComment :- The above command is used to capture " arp" packets .-n means not to convert the addresses to namesCapture only those packets which are going to the discarded port or have arp packets . Show the ARP cache on hostC1. [student@hostC1 ~]$ arp -ahostC1. test. ca (10. 1. 1. 1) at 00: 00: 0a: 00: 01: 01 on e0 permanent [ethernet]routerC. test. ca (10. 1. 1. 254) at 00: 00: 0a: 00: 01: fe on e0 expires in 372 seconds [ethernet]Comment :- The above commands displays the entries in cache table prior of making use of the telnet command . the table contains the entry of host C1 ie its MAC address associated with the IP address. On hostC1 telnet to 10. 1. 3. 5 (non-existent) [student@hostC1 ~]$ date; telnet 10. 1. 3. 5; dateComment :- The above command will provide the date and perform a telnet i. e. remote login operation from hostC1 to a non existent host at 10. 1. 3. 5Observe the tcpdump output. (Observation is provided below)

Results and Discussion:

Results:

ARP cache before the experiment:

[student@hostC1 ~]$ arp -ahostC1. test. ca (10. 1. 3. 1) at 00: 00: 0a: 00: 03: 01 on e0 permanent [ethernet]hostC2. test. ca (10. 1. 3. 2) at 00: 00: 0a: 00: 03: 02 on e0 expires in 718 seconds [ethernet]routerC. test. ca (10. 1. 3. 254) at 00: 00: 0a: 00: 03: fe on e0 expires in 666 seconds [ethernet]Comment :- The above output enlist the entries in the arp cache table , ie address resolution of the IP to MAC of host C1 nd C2

telnet to a nonexistent host:

[student@hostC1 ~]$ date; telnet 10. 1. 3. 5 discard; dateTue Jan 22 14: 41: 53 AST 2013Trying 10. 1. 3. 5... telnet: connect to address 10. 1. 3. 5: Operation timed outtelnet: Unable to connect to remote hostTue Jan 22 14: 43: 08 AST 2013Comment :- Performing a Telnet operation from host C1 to a nonexistent host at 10. 1. 3. 5 and showing date on the output .

tcpdump output:

[student@hostC1 ~]$ sudo tcpdump -n port discard or arptcpdump: verbose output suppressed, use -v or -vv for full protocol decodelistening on e0, link-type EN10MB (Ethernet), capture size 96 bytes14: 41: 53. 160312 ARP, Request who-has 10. 1. 3. 5 tell 10. 1. 3. 1, length 2814: 41: 56. 159482 ARP, Request who-has 10. 1. 3. 5 tell 10. 1. 3. 1, length 2814: 41: 59. 358293 ARP, Request who-has 10. 1. 3. 5 tell 10. 1. 3. 1, length 2814: 42: 02. 557514 ARP, Request who-has 10. 1. 3. 5 tell 10. 1. 3. 1, length 2814: 42: 05. 756347 ARP, Request who-has 10. 1. 3. 5 tell 10. 1. 3. 1, length 2814: 42: 08. 955366 ARP,

Request who-has 10. 1. 3. 5 tell 10. 1. 3. 1, length 2814: 42: 15. 153501 ARP, Request who-has 10. 1. 3. 5 tell 10. 1. 3. 1, length 2814: 42: 27. 349574 ARP, Request who-has 10. 1. 3. 5 tell 10. 1. 3. 1, length 2814: 42: 51. 542219 ARP, Request who-has 10. 1. 3. 5 tell 10. 1. 3. 1, length 28^C9 packets captured27 packets received by filterComment :- The above tcpdump output explains that when there is no reply of the ARP i. e Ethernet broadcast to get the MAC address of the Destination host . It tries again to send ARP request after 3 seconds for first 6 tries and then ARP request is made after an interval of 7 seconds . i. e. it grows exponentially .

ARP cache after the ping:

[student@hostC1 ~]$ arp -ahostC1. test. ca (10. 1. 3. 1) at 00: 00: 0a: 00: 03: 01 on e0 permanent [ethernet]hostC2. test. ca (10. 1. 3. 2) at 00: 00: 0a: 00: 03: 02 on e0 expires in 569 seconds [ethernet]routerC. test. ca (10. 1. 3. 254) at 00: 00: 0a: 00: 03: fe on e0 expires in 1005 seconds [ethernet]

•

Comment :- The above output shows the no change in the " arp" cache table as we there was not ARP reply. Discussion :- The above lab explains how the ARP request time grows exponentially when a telnet request is made to a non - existent host after not receiving the ARP reply the Telnet operation gives up in 75 seconds from when the request was made initially

Conclusion:

We have observed ARP requests to a non-existent host. We also observed the ??? exponential backoff??? used by TCP and the overall TCP timeout of 75 seconds.

Exercises 5. 2

Solution :- The mechanism that can be used for preventing the collision delaying the redundant response from the servers by making use of counter or timing , as the client works on the first reply . Lab 6. 5: Use tcpdump to show an exchange of messages that includes an unreachable port message. Briefly describe how you generated/caused the unreachable port message, show the usage of commands/programs and also annotate the exchange of messages. Explain how the ICMP message is used to identify the IP datagram which caused the error.

Purpose of the Lab:

The purpose of the lab is to observe an ICMP port unreachable message and to see how the ICMP message is used to identify the IP datagram which caused the error.

Lab Setup:

Start tcpdump on hostC1.[student@hostC1 ~]$ sudo tcpdump -N -vvv -s512 udp and port 32000 or icmptcpdump: listening on e0, link-type EN10MB (Ethernet), capture size 512 bytesComment :- The above command on captures traffic of icmp or udp on port 32000-N means not to provide fully qualified domain name-vvv provide verbose or fully descriptive output-s512 means the packet size should be of 512On hostC1 send a single UDP datagram (UDP data size = 100 bytes) to a non-existent port on hostA1. sock -i -u -n1 -w 100 hostA1 32000Comment: ??? sock??? creates an end point communication socket host with following conditions :-i: means source data should be send through this socket-u: ensures UDP encapsulation .-n1:

determines that number of datagram to be sent should be 1.-w: write the raw packet to file of 100 bytes. hostA1 32000: means the host A1 is the destination with port 32000Observe the tcpdump output. (Observation is done below)

Results and Discussion:

Results:

tcpdump output:

[student@hostC1 ~]$ sudo tcpdump -N -vvv -s512 udp and port 32000 or icmptcpdump: listening on e0, link-type EN10MB (Ethernet), capture size 512 bytes16: 08: 37. 711566 IP (tos 0x0, ttl 64, id 21806, offset 0, flags [none], proto UDP (17), length 128)hostC1. 31080 > hostA1. 32000: [udp sum ok] UDP, length 10016: 08: 37. 734810 IP (tos 0x0, ttl 62, id 21819, offset 0, flags [none], proto ICMP (1), length 56)hostA1 > hostC1: ICMP hostA1 udp port 32000 unreachable, length 36IP (tos 0x0, ttl 62, id 21806, offset 0, flags [none], proto UDP (17), length 128)hostC1. 31080 > hostA1. 32000: UDP, length 100^CComment :- The above output shows the ICMP error message generated when no connectivity was made for the transfer for the UDP packet at port number 32000 , in reply host C1 gets a ICMP error message stating the " UDP " port is unreachable . The ICMP error message includes the IP header of the datagram that generated the error along with the first 8 byte following the IP header . Discussion: In the above LAB we learnt how for a unreachable UDP port the sender gets a ICMP reply ie ICMP error message in which it contains the IP header of the datagram that generated the error along with the first 8 bytes of the header followed by the UDP header

Ethernet Header

IP header

ICMP header

IP header of datagram that generated the error

UDP header

Conclusion:

We have observed an ICMP port unreachable message and we have shown how the ICMP message is used to identify which IP datagram caused the errorLab 7. 2: Show the result of performing a ping on our network. As usual, show relevant portions of the tcpdump output in each of the cases. Send only 5 ping requests to keep the output short. Explain how the round trip times (RTT) are computed.

Purpose of the Lab:

The purpose of the lab is to observe an ICMP echo request/reply and to see how the round trip times are calculated.

Lab Setup:

Start tcpdump on hostC1.[student@hostC1 ~]$ sudo tcpdump -n -vv icmpComment :- The above command is used to capture the icmp traffic on host C1-n :- means not to convert the host address in to names .-vv :- the output should be verbose i. e. descriptive. On hostC1 send 5 pings to hostA1[student@hostC1 ~]$ ping -c5 hostA1Comment :- The above command is performing a ping operation from host C1 to A1-c5 means the count should be 5 , i. e. do ping 5 timesObserve the tcpdump output. (Observation below)

Results and Discussion:

Results:

Send 5 pings to hostA1:

[student@hostC1 ~]$ ping -c5 hostA1PING hostA1. test. ca (10. 1. 1. 1): 56 data bytes64 bytes from 10. 1. 1. 1: icmp_seq= 0 ttl= 62 time= 22. 302 ms64 bytes from 10. 1. 1. 1: icmp_seq= 1 ttl= 62 time= 22. 505 ms64 bytes from 10. 1. 1. 1: icmp_seq= 2 ttl= 62 time= 21. 452 ms64 bytes from 10. 1. 1. 1: icmp_seq= 3 ttl= 62 time= 22. 478 ms64 bytes from 10. 1. 1. 1: icmp_seq= 4 ttl= 62 time= 21. 524 ms--- hostA1. test. ca ping statistics ---5 packets transmitted, 5 packets received, 0. 0% packet lossround-trip min/avg/max/stddev = 21. 452/22. 052/22. 505/0. 466 msComment :- Above output shows the result of 5 consecutive " ping " operation form host C1 to A1It provides the minimum , maximum , round trip time of the request to reply and the standard devition in the result .

tcpdump output:

[student@hostC1 ~]$ sudo tcpdump -n -vv icmptcpdump: listening on e0, link-type EN10MB (Ethernet), capture size 96 bytes16: 42: 16. 314827 IP (tos 0x0, ttl 64, id 21819, offset 0, flags [none], proto ICMP (1), length 84)10. 1. 3. 1 > 10. 1. 1. 1: ICMP echo request, id 4108, seq 0, length 6416: 42: 16. 337005 IP (tos 0x0, ttl 62, id 21829, offset 0, flags [none], proto ICMP (1), length 84)10. 1. 1. 1 > 10. 1. 3. 1: ICMP echo reply, id 4108, seq 0, length 6416: 42: 18. 031328 IP (tos 0x0, ttl 64, id 21820, offset 0, flags [none], proto ICMP (1), length 84)10. 1. 3. 1 > 10. 1. 1. 1: ICMP echo request, id 4108, seq 1, length 6416: 42: 18. 053794 IP (tos 0x0, ttl 62, id 21830, offset 0, flags [none], proto ICMP (1), length 84)10. 1. 1. 1 > 10. 1. 3. 1: ICMP echo

reply, id 4108, seq 1, length 6416: 42: 19. 760838 IP (tos 0x0, ttl 64, id 21821, offset 0, flags [none], proto ICMP (1), length 84)10. 1. 3. 1 > 10. 1. 1. 1: ICMP echo request, id 4108, seq 2, length 6416: 42: 19. 782248 IP (tos 0x0, ttl 62, id 21831, offset 0, flags [none], proto ICMP (1), length 84)10. 1. 1. 1 > 10. 1. 3. 1: ICMP echo reply, id 4108, seq 2, length 6416: 42: 21. 486417 IP (tos 0x0, ttl 64, id 21822, offset 0, flags [none], proto ICMP (1), length 84)10. 1. 3. 1 > 10. 1. 1. 1: ICMP echo request, id 4108, seq 3, length 6416: 42: 21. 508870 IP (tos 0x0, ttl 62, id 21832, offset 0, flags [none], proto ICMP (1), length 84)10. 1. 1. 1 > 10. 1. 3. 1: ICMP echo reply, id 4108, seq 3, length 6416: 42: 23. 208102 IP (tos 0x0, ttl 64, id 21823, offset 0, flags [none], proto ICMP (1), length 84)10. 1. 3. 1 > 10. 1. 1. 1: ICMP echo request, id 4108, seq 4, length 6416: 42: 23. 229602 IP (tos 0x0, ttl 62, id 21833, offset 0, flags [none], proto ICMP (1), length 84)10. 1. 1. 1 > 10. 1. 3. 1: ICMP echo reply, id 4108, seq 4, length 64Comment :- The above output of 5 ping commands performed results in 5 pairs of request and reply messageEach ICMP request has identifier 21819 and reply identifier of 21819+10 = 21829, These identifier are incremented by 1 in the next ping operation . Each ICMP request /reply has a sequence number , in this lab of 5 ping we have seq 0 , seq1 ... upto seq4 as the sequence numberAll the above request and reply has the same id of 4108Discussion: In the above lab we learnt how the 5 consecutive ping results . The ping operations provides the " Round Trip Time " calculation , as it records the time when a ICMP request is made and when the echo reply is received it subtract this value from the current time . The results also provide information on how ICMP echo request and reply works with different sequence , identifier numbers .

Conclusion:

We have observed how the ping program works including how the RTT is calculated. Lab 7. 3: Show the result of a ping with Record Route Option through at least 3 routers on our network. Send only 1 ping request to keep the output short. Use the -vvv switch of tcpdump to see the IP header options. Show how the tcpdump output indicates the options portion of the IP header. Show how our tcpdump indicates the pointer position.

Purpose of the Lab:

The purpose of the lab is to observe how ping -R is implemented using the Record Route IP header option.

Lab Setup:

Start tcpdump on hostA1.[student@hostA1 ~]$ sudo tcpdump -n -vvv icmpComment :- The above command is used to capture the icmp traffic on host C1-n :- means not to convert the host address in to names .-vv :- the output should be verbose i. e. descriptive. On hostA1 send 1 ping with the record route option to hostD1[student@hostA1 ~]$ ping -R -c1 hostD1Comment :- performing ping operation from A1 to host D1-R means record route-c1 means the count should be 1Observe the tcpdump output.

Results and Discussion:

Results:

ping output:

[student@hostA1 ~]$ ping -R -c1 hostD1PING hostD1. test. ca (10. 1. 4. 1): 56 data bytes64 bytes from 10. 1. 4. 1: icmp_seq= 0 ttl= 61 time= 44. 911

msRR: routerA. test. ca (10. 1. 6. 1)routerC. test. ca (10. 1. 3. 254)routerD. test. ca (10. 1. 4. 254)hostD1. test. ca (10. 1. 4. 1)routerD. test. ca (10. 1. 3. 3)routerC. test. ca (10. 1. 6. 2)routerA. test. ca (10. 1. 1. 254)hostA1. test. ca (10. 1. 1. 1)--- hostD1. test. ca ping statistics ---1 packets transmitted, 1 packets received, 0. 0% packet lossround-trip min/avg/max/stddev = 44. 911/44. 911/44. 911/0. 000 ms[student@hostA1 ~]$Comment :- The above output of shows the output interfaces from where the ICMP request were forwarded. We are able to receive the output interfaces because of the record route option enabled in the ping command .

tcpdump output:

[student@hostA1 ~]$ sudo tcpdump -n -vvv icmptcpdump: listening on e0, link-type EN10MB (Ethernet), capture size 96 bytes17: 26: 38. 333266 IP (tos 0x0, ttl 64, id 21848, offset 0, flags [none], proto ICMP (1), length 124, options (RR 0. 0. 0. 0 0. 0. 0. 0 0. 0. 0. 0 0. 0. 0. 0 0. 0. 0. 0 0. 0. 0. 0 0. 0. 0. 0 0. 0. 0. 0 0. 0. 0. 0, EOL))10. 1. 1. 1 > 10. 1. 4. 1: ICMP echo request, id 45839, seq 0, length 6417: 26: 38. 378053 IP (tos 0x0, ttl 61, id 21827, offset 0, flags [none], proto ICMP (1), length 124, options (RR 10. 1. 6. 1, 10. 1. 3. 254, 10. 1. 4. 254, 10. 1. 4. 1, 10. 1. 3. 3, 10. 1. 6. 2, 10. 1. 1. 254, 0. 0. 0. 0 0. 0. 0. 0, EOL))10. 1. 4. 1 > 10. 1. 1. 1: ICMP echo reply, id 45839, seq 0, length 64Comment :- = The above output shows the ICMP request with no addresses in it , and the ICMP reply message with 7 IP addresses i. e. the outbound interfaces of the routers through which the ICMP request was forwarded from A1 to D1 and reply that came back from D1 to A1Discussion: The record route option in the ping command - R , helps in knowing the recording the output interfaces of the routers used in from the A1 to D1. In

the ICMP reply we get all the 7 output interfaces IP addresses used . Not all the routers work on the record route operations and the size of the field to accommodate all the IP is also small because of which trace route command is used instead of Ping -R

Conclusion:

We have observed how the ping program with the ??? R option uses the Record Route IP header option to record the outgoing interfaces of the routers in the path from source to destination.