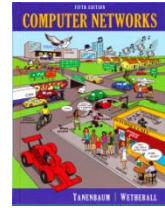


Lab Exercise – ICMP



Objective

To see how ICMP (Internet Control Message Protocol) is used. ICMP is a companion protocol to IP that helps IP to perform its functions by handling various error and test cases. It is covered in §5.6.4 of your text. Review that section before doing this lab.

Requirements

Wireshark: This lab uses the Wireshark software tool to capture and examine a packet trace. A packet trace is a record of traffic at a location on the network, as if a snapshot was taken of all the bits that passed across a particular wire. The packet trace records a timestamp for each packet, along with the bits that make up the packet, from the lower-layer headers to the higher-layer contents. Wireshark runs on most operating systems, including Windows, Mac and Linux. It provides a graphical UI that shows the sequence of packets and the meaning of the bits when interpreted as protocol headers and data. It color-codes packets by their type, and has various ways to filter and analyze packets to let you investigate the behavior of network protocols. Wireshark is widely used to troubleshoot networks. You can download it from www.wireshark.org if it is not already installed on your computer. We highly recommend that you watch the short, 5 minute video “Introduction to Wireshark” that is on the site.

tracert / traceroute: This lab uses “tracert” to find the router level path from your computer to a remote Internet host. `tracert` is a standard command-line utility for discovering the Internet paths that your computer uses. It is widely used for network troubleshooting. It comes pre-installed on Windows and Mac, and can be installed using your package manager on Linux. On Windows, it is called “tracert”. It has various options, but simply issuing the command “tracert `www.uwa.edu.au`” will cause your computer to find and print the path to the remote computer (here `www.uwa.edu.au`).

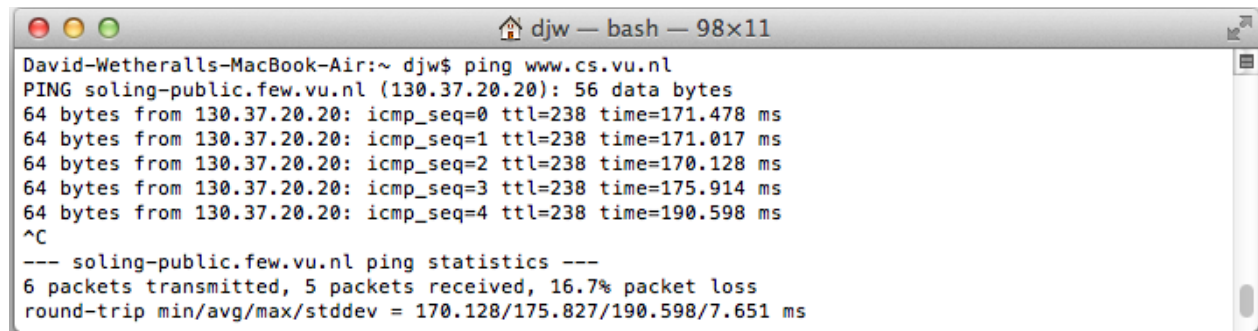
ping: This lab uses “ping” to send and receive messages. `ping` is a standard command-line utility for checking that another computer is responsive. It is widely used for network troubleshooting and comes pre-installed on Windows, Linux, and Mac. While `ping` has various options, simply issuing the command “ping `www.bing.com`” will cause your computer to send a small number of ICMP ping requests to the remote computer (here `www.bing.com`), each of which should elicit an ICMP ping response.

Step 1: Capture a Trace

Proceed as follows to capture a trace of **ICMP traffic that results from ping and traceroute**; alternatively, you may use a supplied trace.

1. Pick a remote computer such as a server as a target, e.g., `www.cs.vu.nl`, and check that you can successfully ping it. To run `ping`, simply bring up a command line and type, e.g., “ping `www.cs.vu.nl`” for our choice of computer. This command should work across Windows, Mac or Linux for our choice of computer. You may have to stop the `ping` program from endlessly sending pings by pressing ctrl-C. You are looking at the `ping` output to see that the re-

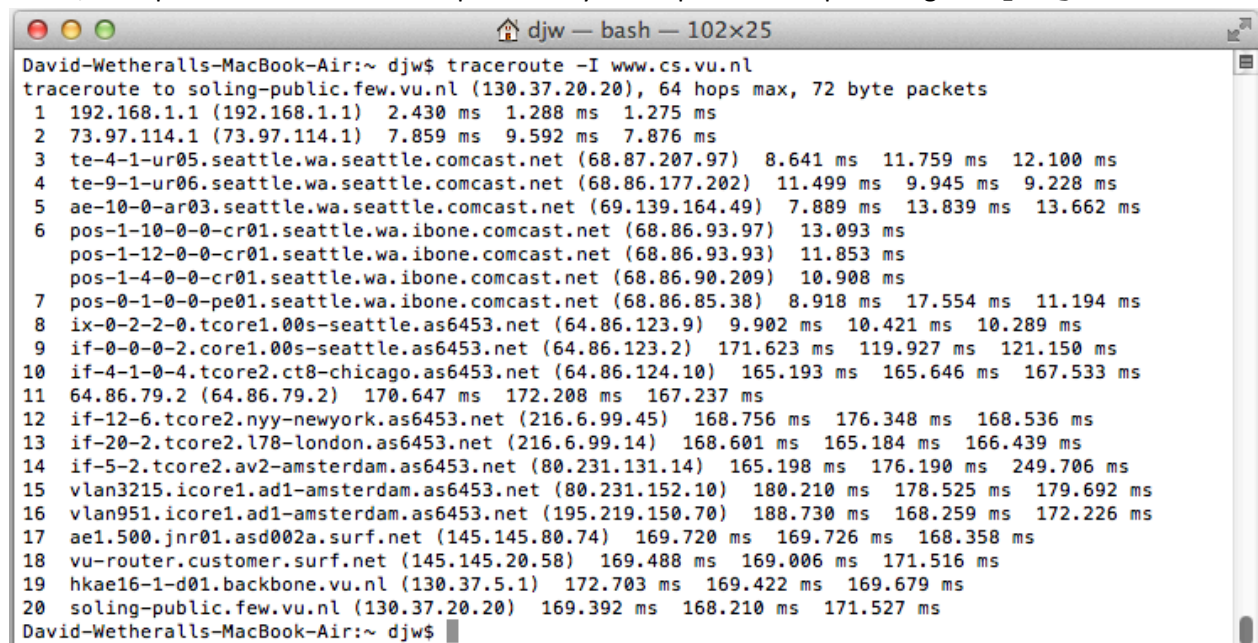
remote computer replies to your ping requests. A successful example is shown in the figure below. If you are not seeing ping replies then pick another remote computer to try.



```
David-Wetheralls-MacBook-Air:~ djw$ ping www.cs.vu.nl
PING soling-public.few.vu.nl (130.37.20.20): 56 data bytes
64 bytes from 130.37.20.20: icmp_seq=0 ttl=238 time=171.478 ms
64 bytes from 130.37.20.20: icmp_seq=1 ttl=238 time=171.017 ms
64 bytes from 130.37.20.20: icmp_seq=2 ttl=238 time=170.128 ms
64 bytes from 130.37.20.20: icmp_seq=3 ttl=238 time=175.914 ms
64 bytes from 130.37.20.20: icmp_seq=4 ttl=238 time=190.598 ms
^C
--- soling-public.few.vu.nl ping statistics ---
6 packets transmitted, 5 packets received, 16.7% packet loss
round-trip min/avg/max/stddev = 170.128/175.827/190.598/7.651 ms
```

Figure 1: Pinging a remote computer (on Mac)

2. Check to see that you can traceroute to the same computer while invoking traceroute with options to send ICMP traffic rather than other traffic such as UDP. To do this, run it as “tracert www.cs.vu.nl” (Windows, uses ICMP) or “traceroute -I www.cs.vu.nl” (Mac/Linux). That was a “minus capital i” option, which tells traceroute to send ICMP probes. It may take a little while to run, printing the next line of output after a pause of several seconds or more. You are looking at the output to see that you can find most of the path to the remote computer. Each line of output gives information about the next segment of the network path. When part of the path cannot be found, then the traceroute output will be a “*” entry that denotes unanswered probes; it is common to have some “*” entries, and this is OK, but for an interesting trace they should not be the majority of the path. A successful example is shown in the figure below. If you are not seeing some information about the path to the remote computer, then pick another remote computer to try and repeat this step starting with ping.



```
David-Wetheralls-MacBook-Air:~ djw$ traceroute -I www.cs.vu.nl
traceroute to soling-public.few.vu.nl (130.37.20.20), 64 hops max, 72 byte packets
 1  192.168.1.1 (192.168.1.1)  2.430 ms  1.288 ms  1.275 ms
 2  73.97.114.1 (73.97.114.1)  7.859 ms  9.592 ms  7.876 ms
 3  te-4-1-ur05.seattle.wa.seattle.comcast.net (68.87.207.97)  8.641 ms  11.759 ms  12.100 ms
 4  te-9-1-ur06.seattle.wa.seattle.comcast.net (68.86.177.202)  11.499 ms  9.945 ms  9.228 ms
 5  ae-10-0-ar03.seattle.wa.seattle.comcast.net (69.139.164.49)  7.889 ms  13.839 ms  13.662 ms
 6  pos-1-10-0-cr01.seattle.wa.ibone.comcast.net (68.86.93.97)  13.093 ms
   pos-1-12-0-cr01.seattle.wa.ibone.comcast.net (68.86.93.93)  11.853 ms
   pos-1-4-0-cr01.seattle.wa.ibone.comcast.net (68.86.90.209)  10.908 ms
 7  pos-0-1-0-pe01.seattle.wa.ibone.comcast.net (68.86.85.38)  8.918 ms  17.554 ms  11.194 ms
 8  ix-0-2-2-tcore1.00s-seattle.as6453.net (64.86.123.9)  9.902 ms  10.421 ms  10.289 ms
 9  if-0-0-2-tcore1.00s-seattle.as6453.net (64.86.123.2)  171.623 ms  119.927 ms  121.150 ms
10  if-4-1-0-4-tcore2.ct8-chicago.as6453.net (64.86.124.10)  165.193 ms  165.646 ms  167.533 ms
11  64.86.79.2 (64.86.79.2)  170.647 ms  172.208 ms  167.237 ms
12  if-12-6-tcore2.nyy-newyork.as6453.net (216.6.99.45)  168.756 ms  176.348 ms  168.536 ms
13  if-20-2-tcore2.l78-london.as6453.net (216.6.99.14)  168.601 ms  165.184 ms  166.439 ms
14  if-5-2-tcore2.av2-amsterdam.as6453.net (80.231.131.14)  165.198 ms  176.190 ms  249.706 ms
15  vlan3215.icore1.ad1-amsterdam.as6453.net (80.231.152.10)  180.210 ms  178.525 ms  179.692 ms
16  vlan951.icore1.ad1-amsterdam.as6453.net (195.219.150.70)  188.730 ms  168.259 ms  172.226 ms
17  ae1.500.jnr01.asd002a.surf.net (145.145.80.74)  169.720 ms  169.726 ms  168.358 ms
18  vu-router.customer.surf.net (145.145.20.58)  169.488 ms  169.006 ms  171.516 ms
19  hkae16-1-d01.backbone.vu.nl (130.37.5.1)  172.703 ms  169.422 ms  169.679 ms
20  soling-public.few.vu.nl (130.37.20.20)  169.392 ms  168.210 ms  171.527 ms
David-Wetheralls-MacBook-Air:~ djw$
```

Figure 2: Traceroute, with ICMP probes, to a remote computer (Mac)

3. Launch Wireshark and start a capture with a filter of "icmp". Make sure to check "enable network name resolution". This will translate the IP addresses of the computers and routers sending packets into names, which will help you to recognize the organizations on the network path taken by your packets. Your capture window should be similar to the one pictured below, other than our highlighting. Select the interface from which to capture as the main wired or wireless interface used by your computer to connect to the Internet. If unsure, guess and revisit this step later if your capture is not successful. Uncheck "capture packets in promiscuous mode". This mode is useful to overhear packets sent to/from other computers on broadcast networks. We only want to record packets sent to/from your computer. Leave other options at their default values. The capture filter, if present, is used to prevent the capture of other traffic your computer may send or receive. On Wireshark 1.8, the capture filter box is present directly on the options screen, but on Wireshark 1.9, you set a capture filter by double-clicking on the interface.

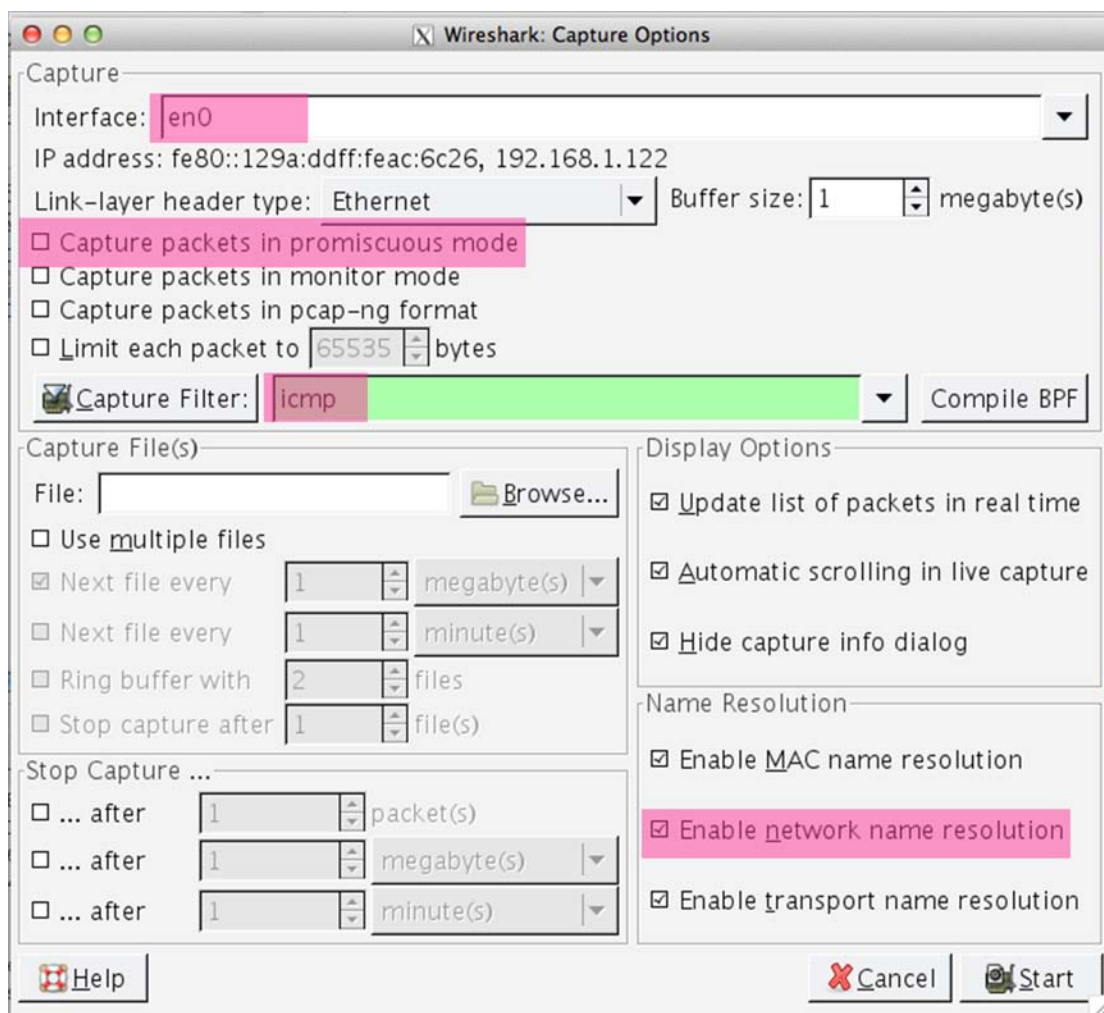


Figure 3: Setting up the capture options

4. When the capture is started, repeat the ping command you tested, wait a few seconds, and then repeat the traceroute command as well. This time, the ICMP packets sent and received by these two programs will be recorded by Wireshark.

- After the commands are complete, return to Wireshark and use the menus or buttons to stop the trace. You should have a short trace similar to that shown in the figure below. We have expanded the detail of the ICMP header for a ping request packet in our view. Be sure to save the output from the ping and traceroute commands. You will need it for the later steps.

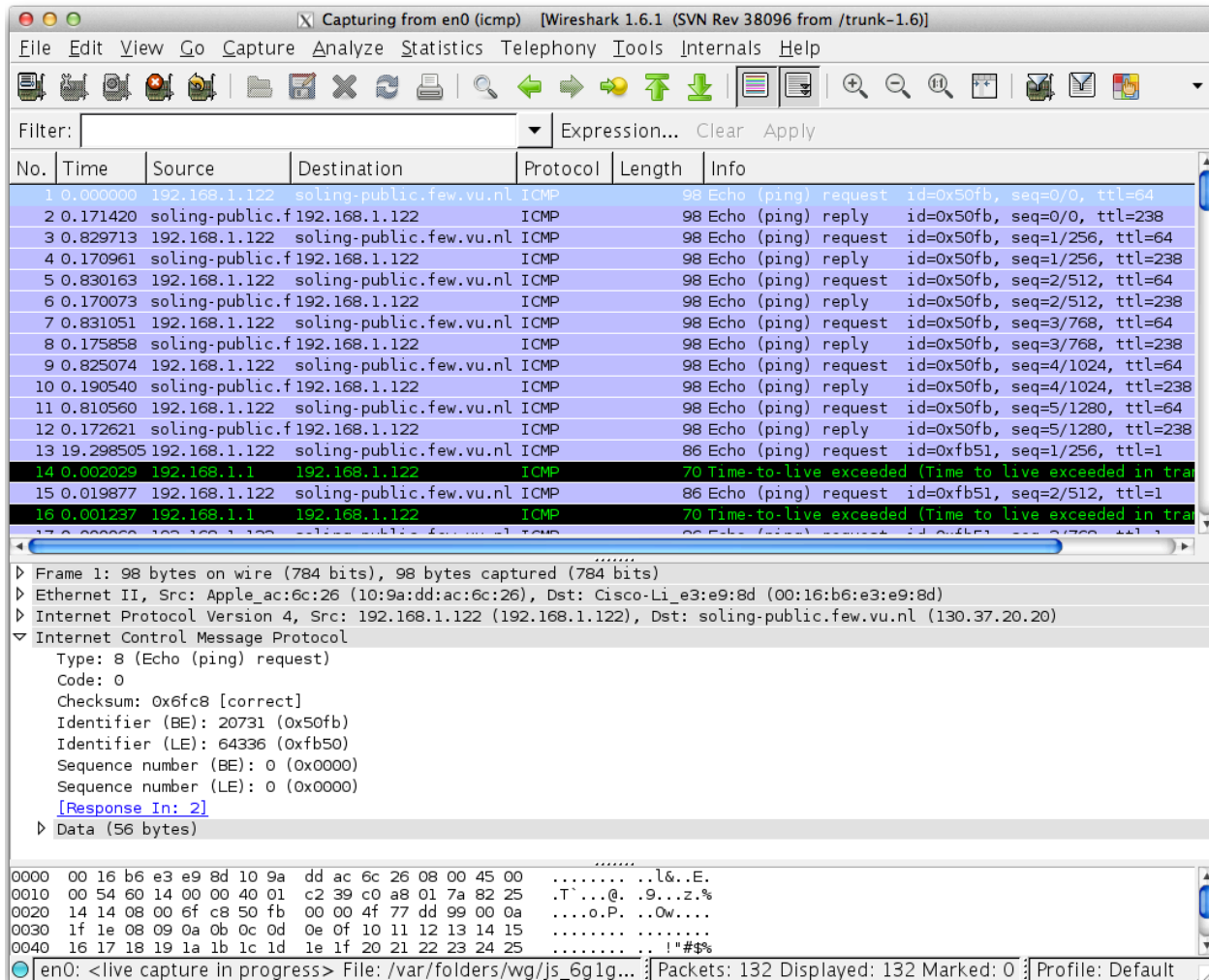


Figure 4: Trace of ping/traceroute traffic showing details of the ICMP header for a ping request.

Step 2: Echo (ping) Packets

Start your exploration by selecting an echo (ping) request and reply packet at the start of the trace. **Expand the ICMP block** (by using the “+” expander or icon) to see the ICMP header and payload details:

- The ICMP header starts with a Type and Code field that identify the kind of ICMP message. Look to see the values for an echo request and an echo reply and how they compare.
- The Type/Code is followed by a 16-bit checksum over the complete ICMP message, to check that it was received correctly.
- Next comes an Identifier and Sequence Number field. These fields are used to link echo request and reply packets together. Look at the Identifier and Sequence Number values for several re-

quests and replies. Compare the values of a request and matching reply, and of successive replies. Note that Wireshark may show these fields in two ways: as a Big Endian (BE) value and a Little Endian (LE). The difference is the order in which the bits are organized into bytes, e.g., 00000001 on the wire might represent “1” or “256” depending on whether the first bit transmitted is the least (LE) or most (BE) significant bit.

- Finally, there is a Data field as the ICMP payload. This field is variable length.

Answer the following questions to demonstrate your understanding of ICMP echo messages:

1. What are the Type/Code values for an ICMP echo request and echo reply packet, respectively?
2. How do the Identifier and Sequence Number compare for an echo request and the corresponding echo reply?
3. How do the Identifier and Sequence Number compare for successive echo request packets?
4. Is the data in the echo reply the same as in the echo request or different?

Turn-in: Hand in your answers to the above questions.

Step 3: TTL Exceeded (traceroute) Packets

Next, explore traceroute traffic by selecting any Time Exceeded ICMP packet in your trace. Expand the ICMP block (by using the “+” expander or icon) to see the ICMP header and payload details:

- The ICMP header starts with a Type and Code field that identify the kind of ICMP message, just as for echo packets. Look to see the values for a TTL Exceeded packet and how they compare to the echo packets.
- The Type/Code is followed by a 16-bit checksum over the complete ICMP message to check that it was received correctly, just as for echo packets.
- The next structure you will see (apart from a possible length field that is part of modern implementations of ICMP) is an IP header! How can this be? For ICMP error messages that are produced by an error during forwarding, such as TTL Exceeded, the start of the packet that triggered the error is included in the payload of the ICMP packet. That is why there is an IP header inside the ICMP packet. Depending on how much of the packet that caused the error is included in the ICMP payload, you may see its headers beyond IP. In the case of our traceroute traffic, this will be an ICMP header of the echo request packet.

Draw a picture of one ICMP TTL Exceeded packet to make sure that you understand its nested structure. On your figure, show the position and size in bytes of the IP header, ICMP header with details of the Type/Code and checksum subfields, and the ICMP payload. Within the ICMP payload, draw another rectangle that shows the overall structure of the contents of the payload. As usual, your figure can simply show the packet as a long, thin rectangle. To work out sizes, observe that when you click on a protocol block in the middle panel (the block itself, not the “+” expander) then Wireshark will highlight the bytes it corresponds to in the packet in the lower panel and display the length at the bottom of the window.

Answer the following questions:

1. What is the Type/Code value for an ICMP TTL Exceeded packet?

2. Say how the receiver can safely find and process all the ICMP fields if it does not know ahead of time what kind of ICMP message to expect. The potential issue, as you have probably noticed, is that different ICMP messages can have different formats. For instance, Echo has Sequence and Identifier fields while TTL Exceeded does not.
3. How long is the ICMP header of a TTL Exceeded packet? Select different parts of the header in Wireshark to see how they correspond to the bytes in the packet.
4. The ICMP payload contains an IP header. What is the TTL value in this header? Explain why it has this value. Guess what it will be before you look!

Turn-in: Hand in your drawing of an ICMP TTL Exceeded packet and the answers to the questions above.

Step 4: Internet Paths

The source and destination IP addresses in an IP packet denote the endpoints of an Internet path, not the IP routers on the network path the packet travels from the source to the destination. `traceroute` is a utility for discovering this path. It works by eliciting ICMP TTL Exceeded responses from the router 1 hop away from the source towards the destination, then 2 hops away from the source, then 3 hops, and so forth until the destination is reached. The responses will identify the IP address of the router. The output from `traceroute` normally prints the information for one hop per line, including the measured round trip times and IP address and DNS names of the router. The DNS name is handy for working out the organization to which the router belongs. Since `traceroute` takes advantage of common router implementations, there is no guarantee that it will work for all routers along the path, and it is usual to see “*” responses when it fails for some portions of the path.

Look at the `traceroute` portion of the trace, which will have a series of ICMP echo request packets followed by ICMP TTL Exceeded packets. The echo requests are sent from the source (your computer) to the destination whose path is being probed. The TTL Exceeded packets are coming from routers along the path back to your computer, triggered by the TTL field counting down to zero.

By looking at the details of the packets, answer the following questions:

1. How does your computer (the source) learn the IP address of a router along the path from a TTL exceeded packet? Say where on this packet the IP address is found. You might proceed by looking at an echo packet to see the source and destination IP addresses. The routers along the path will have a different IP address, and this address will be present on the TTL Exceeded packet. If you are unsure, you can examine the `traceroute` text output to see the IP addresses of routers and look for these addresses on the TTL Exceeded packets.
2. How many times is each router along the path probed by `traceroute`? Look at the TTL Exceeded responses and see if you can discern a pattern.
3. How does your computer (the source) craft an echo request packet to find (by eliciting a TTL Exceeded response) the router N hops along the path towards the destination? Describe the key attributes of the echo request packet. The echo request packets sent by `traceroute` are probing successively more distant routers along the path. You can look at these packets and see how they differ when they elicit responses from different routers.

Now that you have an understanding of the ICMP packets involved, let us look at the output of the `tracert` program. If you are using the supplied trace, note that we have provided the corresponding `tracert` output as a separate file. The output describes the path from your computer to the remote destination using information gleaned from the TTL Exceeded responses.

Using the `tracert` output, sketch a drawing of the network path. Show your computer (lefthand side) and the remote server (righthand side), both with IP addresses, as well as the routers along the path between them numbered by their distance on hops from the start of the path. You can find the IP address of your computer and the remote server on the echo packets in the trace that you captured. The output of `tracert` will tell you the hop number for each router.

To finish your drawing, label the routers along the path with the name of the real-world organization to which they belong. To do this, you will need to interpret the domain names of the routers given by `tracert`. If you are unsure, label the routers with the domain name of what you take to be the organization. Ignore or leave blank any routers for which there is no domain name (or no IP address).

This is not an exact science, so we will give some examples. Suppose that `tracert` identifies a router along the path by the domain name `arouter.cac.washington.edu`. Normally, we can ignore at least the first part of the name, since it identifies different computers in the same organization and not different organizations. Thus we can ignore at least “`arouter`” in the domain name. For generic top-level domains, like “.com” and “.edu”, the last two domains give the domain name of the organization. So for our example, it is “`washington.edu`”. To translate this domain name into the real-world name of an organization, we might search for it on the web. You will quickly find that `washington.edu` is the University of Washington. This means that “`cac`” portion is an internal structure in the University of Washington, and not important for the organization name. You would write “University of Washington” on your figure for any routers with domain names of the form `*.washington.edu`.

Alternatively, consider a router with a domain name like `arouter.syd.aarnet.net.au`. Again, we ignore at least the “`arouter`” part as indicating a computer within a specific organization. For country-code top-level domains like “.au” (for Australia) the last three domains in the name will normally give the organization. In this case the organization’s domain name is `aarnet.net.au`. Using a web search, we find this domain represents AARNET, Australia’s research and education network. The “`syd`” portion is internal structure, and a good guess is that it means the router is located in the Sydney part of AARNET. So for all routers with domain names of the form `*.aarnet.net.au`, you would write “AARNET” on your figure. While there are no guarantees, you should be able to reason similarly and at least give the domain name of the organizations near the ends of the path.

Turn-in: Hand in your answers to the above questions and your drawing of the path, plus `tracert` output if it was not supplied to you.

Explore on your own

We encourage you to explore ICMP on your own once you have completed this lab and also the IP lab, since IP and ICMP are strongly related. Some ideas:

- `tracert` comes in several versions with many options. Explore them; we recommend that you try Paris `tracert`. Often, they will send and receive different kinds of traffic, including ICMP traffic, to reason about network structure.
- We studied ICMP traffic produced by `ping` and `tracert`. There are a variety of other ICMP messages that indicate other network conditions, such as path MTU discovery and unreachable network destinations. See if you can capture other ICMP messages to examine.
- Explore other ICMP based tools that examine the network, such as `pathchar` which estimates the speed of the links along the network path.
- Read about IP alias resolution and learn how it works.

[END]