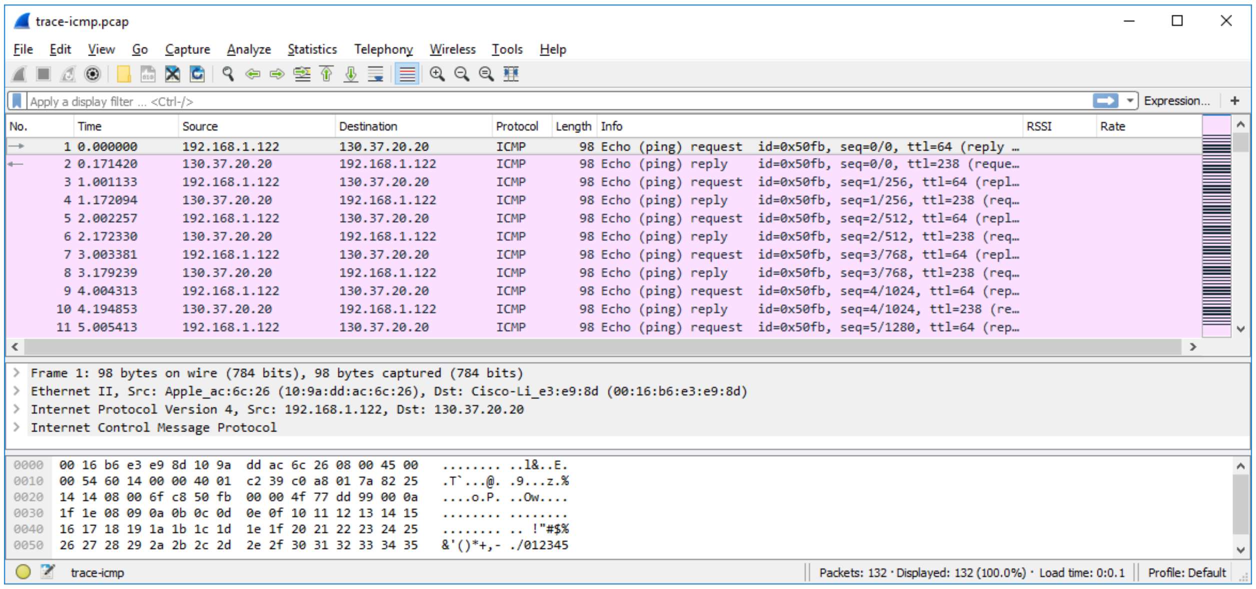
Lab Exercise – ICMP

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. The **Internet Control Message Protocol** (**ICMP**) is a supporting protocol in the Internet protocol suite. It is used by network devices, including routers, to send error messages and operational information indicating, for example, that a requested service is not available or that a host or router could not be reached. ICMP differs from transport protocols such as TCP and UDP in that it is not typically used to exchange data between systems, nor is it regularly employed by end-user network applications (with the exception of some diagnostic tools like ping and traceroute).

Step 1: Open the Trace

The trace is here: https://kevincurran.org/com320/labs/wireshark/trace-icmp.pcap

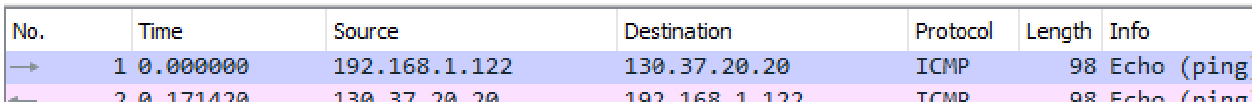

trace-icmp.pcap
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Expression...
trace-icmp
Packets: 132 · Displayed: 132 (100.0%) · Load time: 0:0.1
Profile: Default

```
No.  Time      Source         Destination    Protocol Length Info                                                          RSSI  Rate
  1 0.000000   192.168.1.122  130.37.20.20   ICMP     98 Echo (ping) request  id=0x50fb, seq=0/0, ttl=64 (reply …
  2 0.171420   130.37.20.20   192.168.1.122  ICMP     98 Echo (ping) reply    id=0x50fb, seq=0/0, ttl=238 (reque…
  3 1.001133   192.168.1.122  130.37.20.20   ICMP     98 Echo (ping) request  id=0x50fb, seq=1/256, ttl=64 (repl…
  4 1.172094   130.37.20.20   192.168.1.122  ICMP     98 Echo (ping) reply    id=0x50fb, seq=1/256, ttl=238 (req…
  5 2.002257   192.168.1.122  130.37.20.20   ICMP     98 Echo (ping) request  id=0x50fb, seq=2/512, ttl=64 (repl…
  6 2.172330   130.37.20.20   192.168.1.122  ICMP     98 Echo (ping) reply    id=0x50fb, seq=2/512, ttl=238 (req…
  7 3.003381   192.168.1.122  130.37.20.20   ICMP     98 Echo (ping) request  id=0x50fb, seq=3/768, ttl=64 (repl…
  8 3.179239   130.37.20.20   192.168.1.122  ICMP     98 Echo (ping) reply    id=0x50fb, seq=3/768, ttl=238 (req…
  9 4.004313   192.168.1.122  130.37.20.20   ICMP     98 Echo (ping) request  id=0x50fb, seq=4/1024, ttl=64 (rep…
 10 4.194853   130.37.20.20   192.168.1.122  ICMP     98 Echo (ping) reply    id=0x50fb, seq=4/1024, ttl=238 (re…
 11 5.005413   192.168.1.122  130.37.20.20   ICMP     98 Echo (ping) request  id=0x50fb, seq=5/1280, ttl=64 (rep…
```

```
> Frame 1: 98 bytes on wire (784 bits), 98 bytes captured (784 bits)
> Ethernet II, Src: Apple_ac:6c:26 (10:9a:dd:ac:6c:26), Dst: Cisco-Li_e3:e9:8d (00:16:b6:e3:e9:8d)
> Internet Protocol Version 4, Src: 192.168.1.122, Dst: 130.37.20.20
> Internet Control Message Protocol
```

```
0000  00 16 b6 e3 e9 8d 10 9a  dd ac 6c 26 08 00 45 00   ........ ..l&..E.
0010  00 54 60 14 00 00 40 01  c2 39 c0 a8 01 7a 82 25   .T`...@. .9...z.%
0020  14 14 08 00 6f c8 50 fb  00 00 4f 77 dd 99 00 0a   ....o.P. ..Ow....
0030  1f 1e 08 09 0a 0b 0c 0d  0e 0f 10 11 12 13 14 15   ........ ........
0040  16 17 18 19 1a 1b 1c 1d  1e 1f 20 21 22 23 24 25   ........ .. !"#$%
0050  26 27 28 29 2a 2b 2c 2d  2e 2f 30 31 32 33 34 35   &'()*+,- ./012345
```

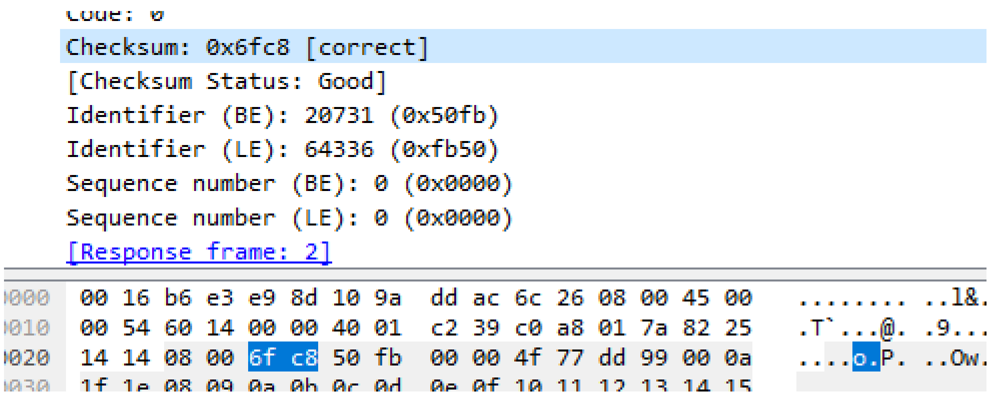
Note how an ICMP packet does not have source and destination port numbers because it was designed to communicate network-layer information between hosts and routers, not between application layer processes.



```
No.  Time       Source          Destination     Protocol  Length Info
  1 0.000000    192.168.1.122   130.37.20.20    ICMP          98 Echo (ping
  2 0.171420    130.37.20.20    192.168.1.122   ICMP          98 Echo (ping
```

Each ICMP packet has a "Type" and a "Code". The Type/Code combination identifies the specific message being received. Since the network software itself interprets all ICMP messages, no port numbers are needed to direct the ICMP message to an application layer process.

If you examine one of the ping request packets sent by your host, you will see the checksum, sequence number and identifier fields are two bytes each. The Checksum is listed below.



```
Code: 0
Checksum: 0x6fc8 [correct]
[Checksum Status: Good]
Identifier (BE): 20731 (0x50fb)
Identifier (LE): 64336 (0xfb50)
Sequence number (BE): 0 (0x0000)
Sequence number (LE): 0 (0x0000)
[Response frame: 2]

000  00 16 b6 e3 e9 8d 10 9a  dd ac 6c 26 08 00 45 00   ........ ..l&.
010  00 54 60 14 00 00 40 01  c2 39 c0 a8 01 7a 82 25   .T`...@. .9...
020  14 14 08 00 6f c8 50 fb  00 00 4f 77 dd 99 00 0a   ....o.P. ..Ow.
030  1f 1e 08 09 0a 0b 0c 0d  0e 0f 10 11 12 13 14 15
```

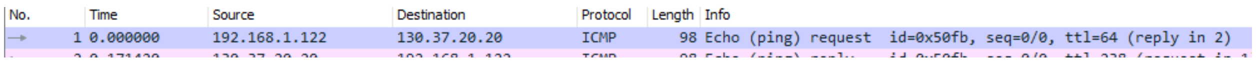
Note that an ICMP error packet is not the same as the ping query packets as it contains both the IP header and the first 8 bytes of the original ICMP packet that the error is for.

The two most important ICMP messages are Echo Request (8) and Echo Reply (0). Echo Request and Echo Reply are utilized by the `ping` command to test network connectivity. `ping` is an extremely useful tool for network troubleshooting.

In order to monitor the uptime on servers and verify that the servers are ONLINE and RESPONDING through monitoring tools, PING and ICMP must be enabled. Failure to have these ports open will break the monitoring tools at a lot of companies running online services which will result in prolonged downtime for the customers as the company would not know if a server was offline until a user reports it.

Step 2: Echo (ping) Packets

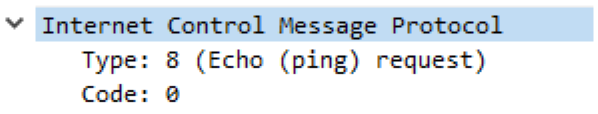
Select the #1 echo (ping) request and reply packet at the start of the trace.



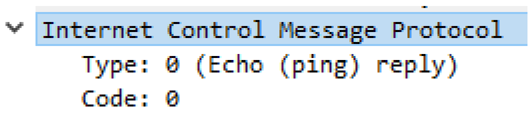
```
No.   Time       Source          Destination    Protocol  Length  Info
  1   0.000000   192.168.1.122   130.37.20.20   ICMP      98      Echo (ping) request  id=0x50fb, seq=0/0, ttl=64 (reply in 2)
```

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.

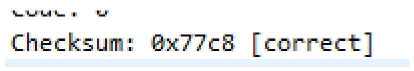


```
Internet Control Message Protocol
    Type: 8 (Echo (ping) request)
    Code: 0
```



```
Internet Control Message Protocol
    Type: 0 (Echo (ping) reply)
    Code: 0
```

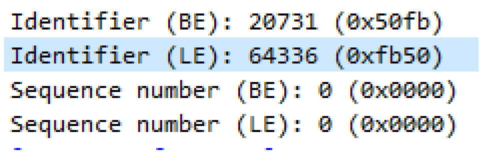
- The Type/Code is followed by a 16-bit checksum over the complete ICMP message, to check that it was received correctly.



```
Checksum: 0x77c8 [correct]
```

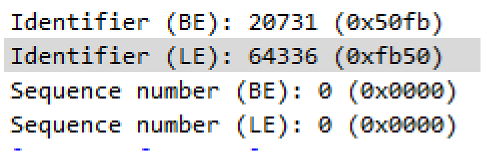
- Next comes an Identifier and Sequence Number field. These fields are used to link echo request and reply packets together. Look at the Identifier and Sequence Number values for several requests and replies. Compare the values of a request and matching reply, and of successive replies.

  e.g. Request



```
Identifier (BE): 20731 (0x50fb)
Identifier (LE): 64336 (0xfb50)
Sequence number (BE): 0 (0x0000)
Sequence number (LE): 0 (0x0000)
```

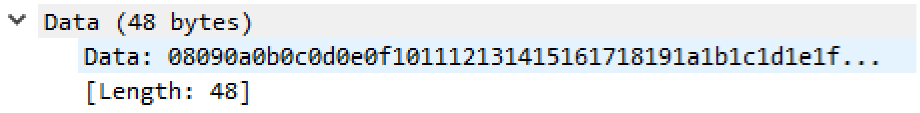
  And response



```
Identifier (BE): 20731 (0x50fb)
Identifier (LE): 64336 (0xfb50)
Sequence number (BE): 0 (0x0000)
Sequence number (LE): 0 (0x0000)
```

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.



```
Data (48 bytes)
    Data: 08090a0b0c0d0e0f101112131415161718191a1b1c1d1e1f...
    [Length: 48]
```

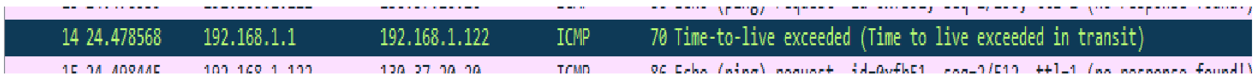
Note the Echo request has Type/Code of 8/0. Echo reply has Type/Code of 0/0.

Each echo request and corresponding echo reply have the same Identifier value and the same Sequence Number value. The values are used to match the echo request to the right echo reply. Typically, the Identifier is kept the same and the Sequence Number is incremented. This ensures that as a pair, successive echo requests will have different Identifier/Sequence Number values so they (and their corresponding replies) can be distinguished.

The data is the same in the echo reply & request. The echo request sender can use any convenient data, and the echo reply sender will copy its data from the request so that the payload returns to the original sender.

Step 3: TTL Exceeded (traceroute) Packets

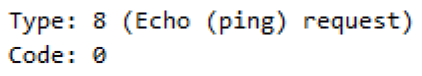
Next, explore `traceroute` *traffic by selecting any Time Exceeded ICMP packet in your trace.*



```
14 24.478568   192.168.1.1   192.168.1.122   ICMP   70 Time-to-live exceeded (Time to live exceeded in transit)
```

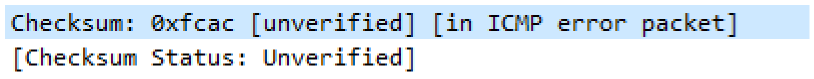
Expand the ICMP block to see the ICMP header and payload details:

- 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.



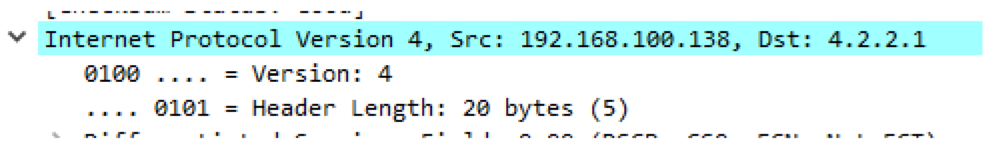
```
Type: 8 (Echo (ping) request)
Code: 0
```

- 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.



```
Checksum: 0xfcac [unverified] [in ICMP error packet]
[Checksum Status: Unverified]
```

- 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.



```
Internet Protocol Version 4, Src: 192.168.100.138, Dst: 4.2.2.1
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
```

We could draw a picture of one ICMP TTL Exceeded packet showing 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.

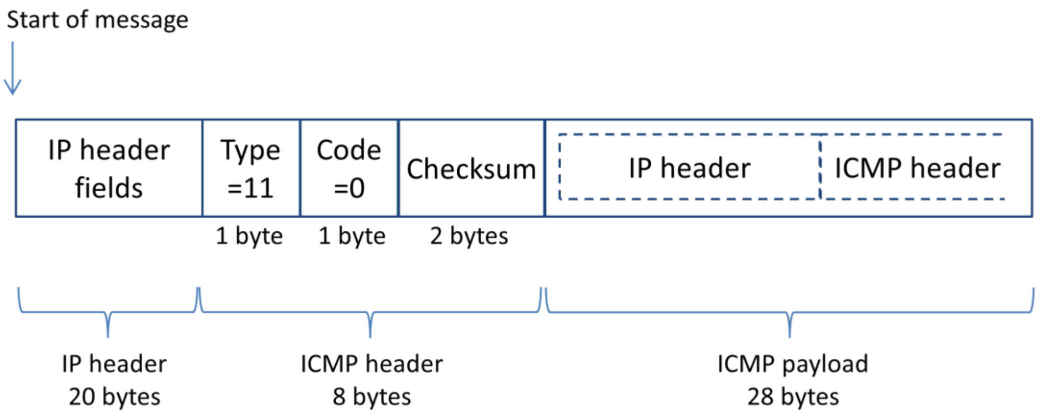


Figure 5: Format of an ICMP TTL Exceeded Message

There are several features to note:

- The length of 20 bytes is for a typical IPv4 header with no IP option fields.
- The Type and Code values are for an ICMP TTL Exceeded in transit message.
- The ICMP header is given as 8 bytes, yet the fields only add up to 4 bytes. There are an extra four bytes after the checksum that are historically unused. They are not shown in the figure because they are not shown in most versions of Wireshark.
- The size of the ICMP payload depends on the router implementation. The value of 28 bytes is what we saw in practice. The start of an IP packet is shown in these bytes, including an IP header and ICMP header for the echo request packet that triggered the ICMP TTL Exceeded message.

The Type/Code value for an ICMP TTL Exceeded packet is 11 (Time Exceeded) and Code=0 (TTL Exceeded in transit).

All ICMP messages start with the same Type/Code (and Checksum) fields, so the receiver can process these fields. Their value tells the receiver the kind of ICMP message, and hence what fields follow.

The Type/Code and Checksum fields take up 4 bytes. However, the ICMP header is actually 8 bytes long. These fields are followed by 4 bytes that are unused (except for recent ICMP extensions) and hence do not show up in Wireshark as named fields. You can still see that they are there by selecting the ICMP block and the payload and observing that they differ by 8 bytes.

The inner IP packet has TTL=1 in our case but depending on the router implementation it is possible that you will see TTL=0. It must be one of these values for the case of an ICMP TTL Exceeded message because the message is triggered when the TTL is decremented during processing and reaches 0, i.e., the TTL held a value of 1 when the packet arrived at the router.

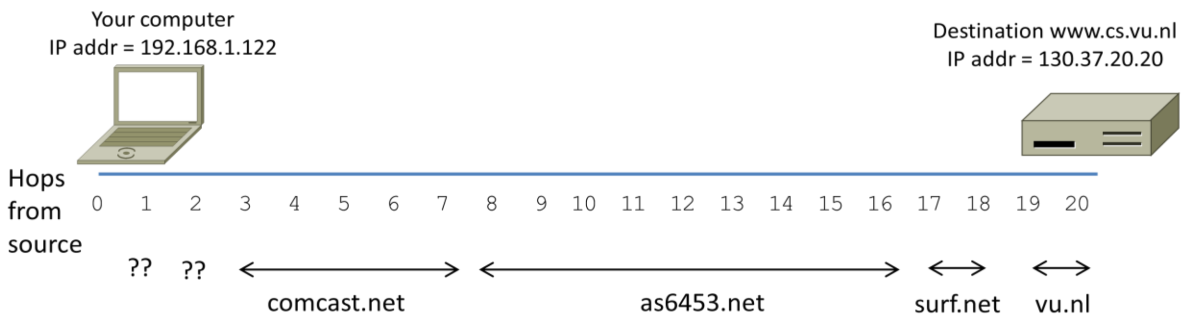
Step 4: Internet Paths

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.

The echo requests are sent from the source 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. The IP source address of the TTL Exceeded packet is the IP address of the router. This is because the router created the TTL Exceeded packet, putting its own IP address in the source field.

Traceroute probes each hop along the path more than once, in case of packet loss. Typically, it probes three times, in which case you will see a pattern of triples of echo / TTL exceeded from a given router. This pattern will not be exact because some TTL Exceeded packets may be lost, and some routers may not reply with TTL exceeded. These lost TTL Exceeded packets correspond to the "*" entries in the traceroute text output.

The echo request packet should have an IP source of the computer doing the tracert command, an IP destination of the far end of the path, and a TTL value set to N. The last part is the key; routers will decrement the TTL and it will reach zero N hops away from the source towards the destination. The ICMP TTL Exceeded message will be sent back to the source. Note that the contents of the ICMP fields in the echo request packet do not matter. They are there only in case the packet reaches the destination (which would then send an echo reply).



Here, the start of the path is not named because it starts within a home; the address 192.168.xx.xx is in private address space that is NATed to reach the public Internet. This will likely be the case if you run the traceroute from a home.