ELEC-C7420 - Basic principles in networking

Tutorial on basic IP routing and ICMP tools

Introduction

The Internet Protocol (IP) is the main communications protocol that enables the internetworking and exchange of packets and datagrams within a network and in essence permits the communications in the modern Internet. Its principal function is to permit the addressing of the NICs in the layer 3 through IP addresses, encapsulate the payload into datagrams and permit the fragmentation and reassembly of the packets in the network. The Internet Protocol suite contains several protocols in the different layers of the OSI model like ARP, SMTP or SNMP, in this sense, a very useful protocol to troubleshoot and discover anomalies or behaviors in the network is the ICMP protocol which have two powerful implementations for this purpose: `ping` and `traceroute`. In this tutorial we will review how the `ping` command could be used to check connectivity issues, confirm that the packets are being received in the destination host and evaluate the network quality and performance by checking the round trip time and the packet loss indicators, on the other hand we will learn how the `traceroute` command will help us to discover the path a packet is following to reach its destination and other useful information this command provides. Also we will review the basics of IP routing to give a flavor on this topic to understand in a high-level manner the overview on IP routing protocols, since routing & switching is a complex and wide issue which have entire books dedicated to its analysis and also industry certifications for specialization like the Cisco CCNA, CCNP, CCIE or the Huawei HCIA, HCIP, HCIE.

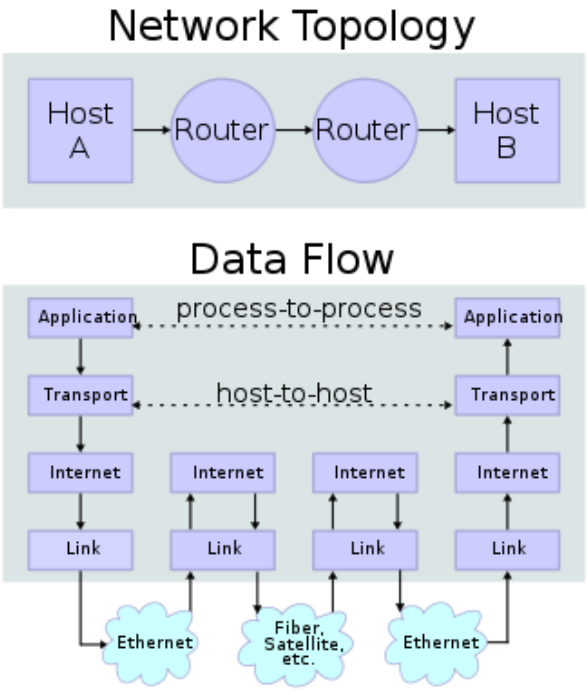

Figure 1. IP communication between two hosts (https://en.wikipedia.org/wiki/Internet_protocol_suite)

Background

IP Routing

The process of IP routing, also known in networking as IP forwarding, is the process where the hosts or routers decide the interface on which they will transmit or forward the IP packet. The routing process works by comparing the destination IP address of the IP packet against the addresses stored in the routing table. We can define the routing table as a database where the router keeps the information to know which direction to send a given datagram so the packet can reach its final destination. The routing table does not contain the detailed information of all the path that has to be followed by the packet, instead, it only stores the next hop router, so we can identify three cases inside the routing table:

- Direct forwarding → the interfaces directly connected
- Indirect forwarding → the routes in the router which does not have a direct connection and are learnt by a routing protocol or by a static route (explicitly defined)
- Indirect forwarding via default router → when a default route is configured (if available)

In general we can define a router as a host with multiple network interfaces that is able to forward the packets or datagrams between different networks by sending the traffic to the proper interface enabling the IP packets to reach their final destination. This devices may be small ones for home use or very complex ones for network or service providers which are able to process millions of packets per second. In terms of routing, the router perform the following tasks:

- Routing → which is the process of building a map to match the IP addresses in the network and provide directions for the packets.
- Forwarding → which is the process of moving the packets to the corresponding interfaces according to the routing directions.

During the IP routing process the router needs to find the path to deliver the IP packets accordingly to ensure the communication, the path that each packet will follow is determined by the information received from routing protocols or by the network administrator in case there are static routes configured. As you can imagine, the routers may have several interfaces connecting to different other routers to create the network, so there may be several alternative paths to forward the packet. In addition, the routing process could be affected by some factors for example: topology updates, predefined policies and/or network metrics like quantity of hops, filters by protocols, delay, bandwidth, cost, etc.

The router must select the interface where the packet will be forwarded, i.e it must perform an IP route lookup based on the destination IP address to decide the next hop for the packet, in this case the router will look for the “most precise match” routing, it means that more specific prefixes will be preferred over more general prefixes.

Example: given the following routing table:

```
Netlab-router#
Netlab-router#
Netlab-router#
Netlab-router# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, P - PIM, A - Babel,
       > - selected route, * - FIB route

S>* 0.0.0.0/0 [1/0] via 192.168.172.17, eth0
C>* 127.0.0.0/8 is directly connected, lo
S>* 192.168.172.0/24 [1/0] via 192.168.172.17, eth0
C>* 192.168.172.16/30 is directly connected, eth0
C>* 192.168.172.20/30 is directly connected, eth1
S>* 192.168.172.48/29 [1/0] via 192.168.172.22, eth1
C>* 192.168.172.56/29 is directly connected, eth2
Netlab-router#
Netlab-router#
Netlab-router#
Netlab-router#
Netlab-router#
Netlab-router#
```

Figure 2. Routing table example

- If a packet has the destination IP address: **192.168.172.51/32**
  - Which interface will the router select to forward the packet?
  - Why?

- If a packet has the destination IP address: **82.130.10.121/32**
  - Which interface will the router select to forward the packet?
  - Why?

Static and Dynamic routes

The most simple mechanism to handle routing tables is by applying static routing. In this scenario the routing tables are configured manually by a network administrator and remain fixed until further updated also manually. Since this approach is simple, it cannot automatically adapt to network changes such as a link failures or congestion. Dynamic routing enables routing tables to be populated and adjusted automatically in order to adapt changing network conditions. Dynamic routes are updated with any change happened on the network based on the network status at that moment. In the case of dynamic routing the network administrator does not make routing decisions and updates the router's routing tables manually. Instead, each device collects network connectivity information from other routers and makes its own routing decisions and populate its routing tables based on routing protocols like BGP, RIP, OSPF, IS-IS. Dynamic routing therefore depends on: protocols to distribute network connectivity information and distributed algorithms that can compute routing tables using this information.

- What is a default route?
  - Is this route dynamic or static?
  - Why?

In the IP forwarding process the router is the one that decides which interface the packet will be delivered to and the forwarding table is filled by the routing process. In this case the IP forwarding decision may be based on the destination address, the class of service or the local requirements (packet filtering). Now, we can distinguish two elements from this process:

- RIB → it is the routing table (Routing Information Base)
    - Contains mainly the list of all destinations known by the router and the corresponding next hop for each destination.
    - Also contains other information like the type of routing protocol and the interfaces names.

- FIB → it is the forwarding table (Forwarding Information Base)
    - Contains the destinations and the interfaces to reach those destinations
    - Used by the router to decide where to send the packet

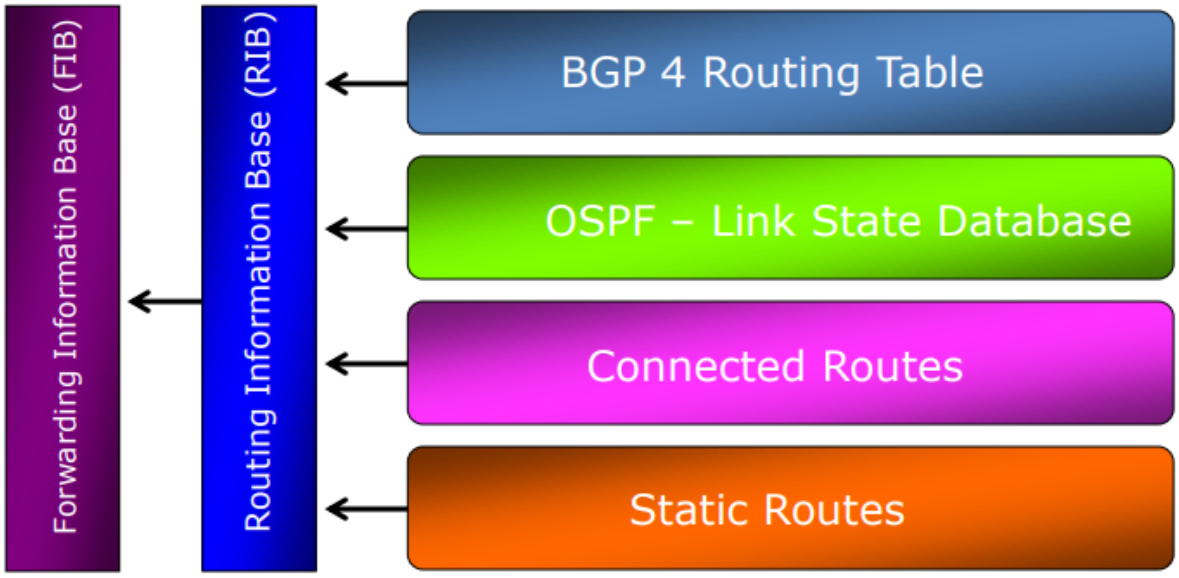

Figure 3. The routing table feeds the forwarding table

Following an example of how an FIB looks like:

```
router#show ip cef

Prefix              Next Hop            Interface
0.0.0.0/32          receive
192.168.0.0/30      attached            Serial2/0/0:1
192.168.0.0/32      receive
```

Figure 4. Example of FIB

ICMP protocol

As we could see in the previous section, the Internet Protocol is quiet simple because it has only one type of packet and in the case of the IP routing/forwarding, the network has to make its best effort to let a given packet reach its destination. However, if along the way of the packet path there are some errors or failures it would be very hard for the routers to know this adverse conditions if there isn't a way to pass back this kind of information.

In this sense, the Internet Control Message Protocol (ICMP) carries this kind of information about some network conditions, this is why in the IP world all the routers and hosts are expected to implement this protocol and understand its messages. In the case of ICMP, most of its packets are related to take diagnostic information which is sent back to the source of the packet when a router in the path of the datagram destroys this packet for a reason or another, e.g if the destination is unreachable or the TTL has expired, this is why ICMP is aimed to provide feedback to the sources about the network conditions, but not to make the service itself more reliable.

Diagnostics:

All ICMP messages start with a 32-bit header that contains three fields: type, code and checksum:

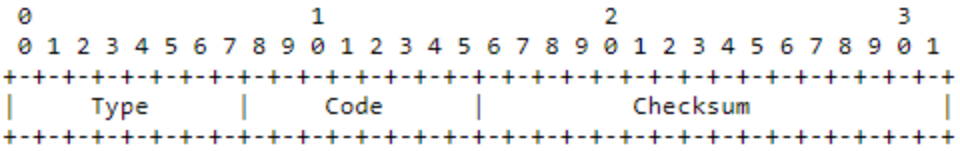

Figure 5. ICMP header

The field 'checksum' is used to determine the consistency of the data in the packet and to detect if there was any change on the bits of the packet during the transmission. The field 'type' indicates the ICMP message that is being sent and ICMP defines several message types:

```
 0  Echo Reply

 3  Destination Unreachable

 4  Source Quench

 5  Redirect

 8  Echo

11  Time Exceeded

12  Parameter Problem

13  Timestamp

14  Timestamp Reply

15  Information Request

16  Information Reply
```

In terms of network diagnostics, the most common messages used for this purpose are the ones that reports or give feedback about operational problems that could involve packet loss, these messages are for example, “time exceeded” (type 11), “destination unreachable” (type 3), “source quench” (type 4) and “parameter problem” (type 12). The above mentioned packets have the same ICMP format:

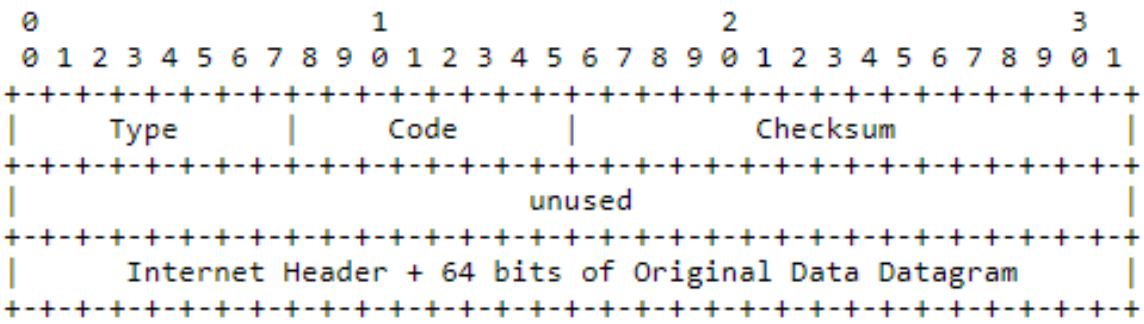

Figure 6. The most frequent ICMP message format

In particular the “destination unreachable” packet results useful for troubleshooting because it provides information in terms of connectivity. In this type of packet the field “code” provides more information on the reason why the packet did not reach its destination:

```
0 = net unreachable;

1 = host unreachable;

2 = protocol unreachable;

3 = port unreachable;

4 = fragmentation needed and DF set;

5 = source route failed.
```

In the case of the “time exceeded” message, these messages are sent when a packet has been destroyed due to its TTL has expired. The “source quench” message is sent by a router in a congestion scenario to let the source know about this condition and it is supposed to reduce its sending rate. In the case of the “parameter problem” message, the header includes a field named “pointer” which indicates the exact octet in the original message where the error was detected:

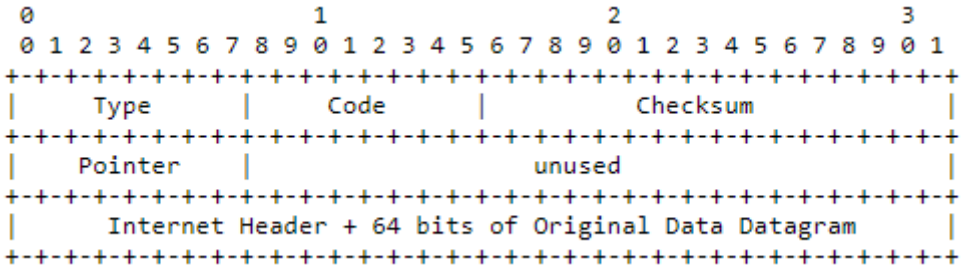

Figure 7. Parameter problem ICMP header

The ICMP protocol is useful to diagnosis of the network and to detect failures like packet loss, congestion or errors, it is very useful as means of troubleshooting specially for connectivity and configuration problems, it has two powerful tools implemented over this protocol for diagnostics purposes: `ping` and `traceroute`. However, for a more detailed and reliable diagnosis there are specific tools like network analyzers that run performance tests like the RFC 2544.

Ping:

`Ping` is a debugging program implemented based on the echo messages included in ICMP. It works by sending an echo ICMP message (type 8) and the receiver of this message must answer with an "echo reply" ICMP message (type 0). This program tests the connectivity with a given IP address. Also it provides useful statics information like the percentage of lost packets during the `ping` test, the round-trip time the packet took in the path and the ICMP-sequence number.

In the networking world it is common to refer as "pinging a host" to run a `ping` to an IP address in order to check for connectivity or to confirm that this device is up and running.

`Ping` is aimed to test the end-to-end connectivity between two devices. Following an example of a successful `ping` test to the IP 192.168.172.50:

```
client1:~#
client1:~# ping -c 20 192.168.172.50
PING 192.168.172.50 (192.168.172.50): 56 data bytes
64 bytes from 192.168.172.50: seq=0 ttl=62 time=0.234 ms
64 bytes from 192.168.172.50: seq=1 ttl=62 time=0.213 ms
64 bytes from 192.168.172.50: seq=2 ttl=62 time=0.214 ms
64 bytes from 192.168.172.50: seq=3 ttl=62 time=0.211 ms
64 bytes from 192.168.172.50: seq=4 ttl=62 time=0.101 ms
64 bytes from 192.168.172.50: seq=5 ttl=62 time=0.227 ms
64 bytes from 192.168.172.50: seq=6 ttl=62 time=0.213 ms
64 bytes from 192.168.172.50: seq=7 ttl=62 time=0.211 ms
64 bytes from 192.168.172.50: seq=8 ttl=62 time=0.225 ms
64 bytes from 192.168.172.50: seq=9 ttl=62 time=0.212 ms
64 bytes from 192.168.172.50: seq=10 ttl=62 time=0.094 ms
64 bytes from 192.168.172.50: seq=11 ttl=62 time=0.212 ms
64 bytes from 192.168.172.50: seq=12 ttl=62 time=0.212 ms
64 bytes from 192.168.172.50: seq=13 ttl=62 time=0.211 ms
64 bytes from 192.168.172.50: seq=14 ttl=62 time=0.212 ms
64 bytes from 192.168.172.50: seq=15 ttl=62 time=0.211 ms
64 bytes from 192.168.172.50: seq=16 ttl=62 time=0.192 ms
64 bytes from 192.168.172.50: seq=17 ttl=62 time=0.207 ms
64 bytes from 192.168.172.50: seq=18 ttl=62 time=0.210 ms
64 bytes from 192.168.172.50: seq=19 ttl=62 time=0.092 ms

--- 192.168.172.50 ping statistics ---
20 packets transmitted, 20 packets received, 0% packet loss
round-trip min/avg/max = 0.092/0.195/0.234 ms
client1:~#
client1:~#
```

Figure 8. Example of `ping` test

Traceroute:

Another popular implementation of the ICMP protocol is the debugging tool called `traceroute` which is intended to discover the different hops the packet passes through on its way to reach its final destination. This program sends regular IP packets on an unused "User Datagram Protocol" (UDP) port, i.e it sends UDP packets, to the destination increasing the TTL field progressively, so the first packet is sent with a TTL = 1, the first hop will receive the packet and will set the TTL = 0, destroy the packet and send back a "TTL expired" message, the source will identify the first hop in the path and then will send a packet with TTL = 2 in order to identify the second hop, if exists, and the process will continue until the final destination is reached or until the point where the connectivity is broken.

```
client1:~#
client1:~# traceroute 192.168.172.50
traceroute to 192.168.172.50 (192.168.172.50), 30 hops max, 46 byte packets
 1  netlab.aalto.fi (192.168.172.57)  0.005 ms  0.004 ms  0.004 ms
 2  netcafe.aalto.fi (192.168.172.22)  0.003 ms  0.004 ms  0.004 ms
 3  client3.aalto.fi (192.168.172.50)  0.004 ms  0.004 ms  0.004 ms
client1:~#
client1:~#
client1:~#
client1:~#
```

Figure 9. Example of `traceroute` test

Example: Consider the following network topology:

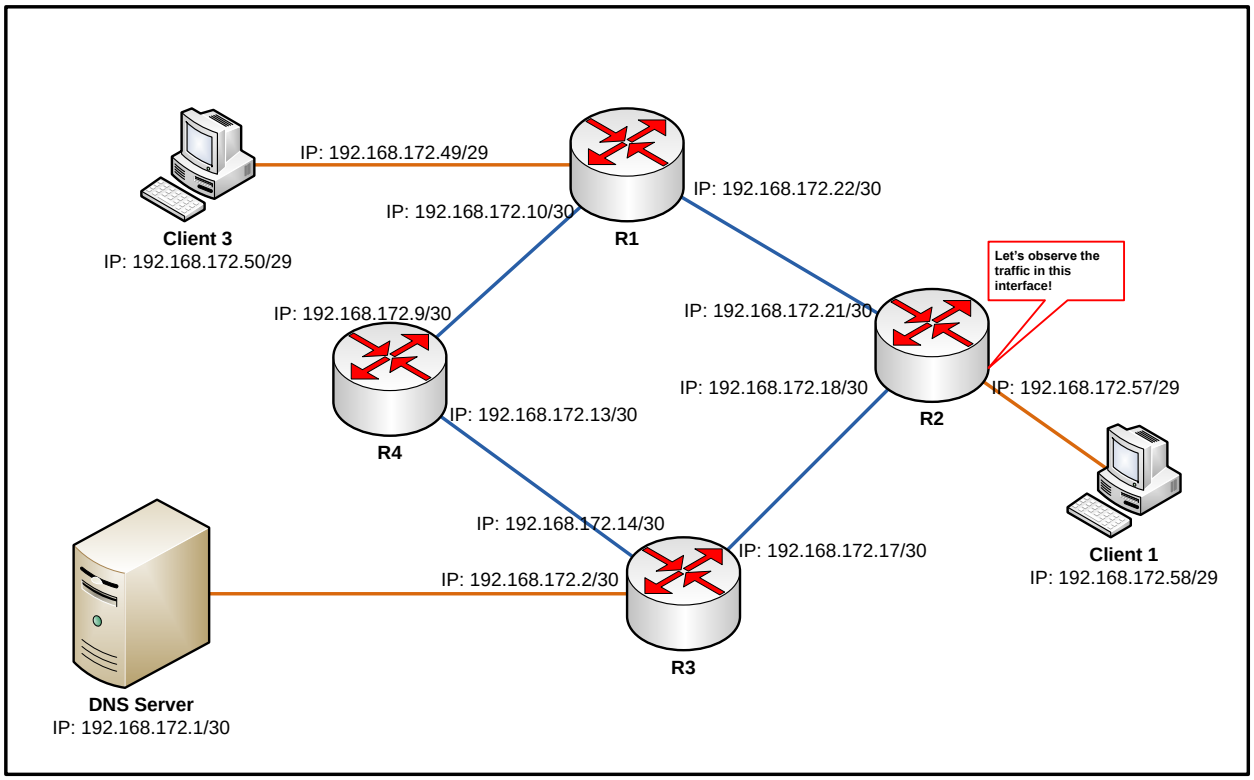

Figure 10. Simple network topology

- We will send 10 `ping` packets from client 1 to client 3 in order to test the connectivity between this two hosts:

```
client1:~#
client1:~# ping -c 10 192.168.172.50
PING 192.168.172.50 (192.168.172.50): 56 data bytes
64 bytes from 192.168.172.50: seq=0 ttl=62 time=0.121 ms
64 bytes from 192.168.172.50: seq=1 ttl=62 time=0.226 ms
64 bytes from 192.168.172.50: seq=2 ttl=62 time=0.076 ms
64 bytes from 192.168.172.50: seq=3 ttl=62 time=0.228 ms
64 bytes from 192.168.172.50: seq=4 ttl=62 time=0.226 ms
64 bytes from 192.168.172.50: seq=5 ttl=62 time=0.232 ms
64 bytes from 192.168.172.50: seq=6 ttl=62 time=0.225 ms
64 bytes from 192.168.172.50: seq=7 ttl=62 time=0.082 ms
64 bytes from 192.168.172.50: seq=8 ttl=62 time=0.224 ms
64 bytes from 192.168.172.50: seq=9 ttl=62 time=0.233 ms

--- 192.168.172.50 ping statistics ---
10 packets transmitted, 10 packets received, 0% packet loss
round-trip min/avg/max = 0.076/0.187/0.233 ms
client1:~#
client1:~#
client1:~#
```

Figure 11. `Ping` results of pinging client 3 from client 1

  - From the above results, is the connectivity between both hosts ok? Why?
  - How many ICMP packets should be captured in the Wireshark trace? Why?
  - Open the trace **'ping_c1_to_c3.pcap'** in your computer and check the ICMP packets captured. Are they consistent with the quantity of packets you answered in the previous item? Why?
  - From the wireshark trace open the ICMP header of an "echo" message and check the "type" field, is it consistent with the ICMP specification? Repeat the same process for the "echo-reply" message.
  - Are there other protocols involved in the communication? If there are other protocols which are they? Why do you think they are present in the capture?

- Now we try to `ping` a public IP address in the internet: 82.130.10.121:

```
client1:~#
client1:~#
client1:~# ping -c 10 82.130.10.121
PING 82.130.10.121 (82.130.10.121): 56 data bytes

--- 82.130.10.121 ping statistics ---
10 packets transmitted, 0 packets received, 100% packet loss
client1:~#
client1:~#
client1:~#
```

Figure 12. `Ping` results of pinging Internet from client 1

  - From the above results, is the connectivity between both hosts ok? Why?
  - Now what would you expect in terms of ICMP packets captured in the Wireshark trace? Why?
  - Open the trace **'ping_c1_to_internet.pcap'** in your computer and check the ICMP packets captured. Are they consistent with the quantity of packets you answered in the previous item? Why?
  - From the wireshark capture which ICMP packets can you identify? Why?
  - Are there other protocols involved in the communication? If there are other protocols what are they? Why do you think they are present in the capture?

As explained before, `ping` helps us to characterize the end-to-end connectivity between two hosts or devices. If we want to know the path a packet follows between two points we can use `traceroute` to determine which hops are involved in the communication. This is particularly useful in cases where we would like to confirm that the traffic is going through the links we configured for this traffic and also helps us to check in which equipment (IP address) the traffic is being dropped in case the end-to-end connectivity is broken.

- Now we will check the hops from client 1 to client 3 using `traceroute`

```
client1:~#
client1:~# traceroute 192.168.172.50
traceroute to 192.168.172.50 (192.168.172.50), 30 hops max, 46 byte packets
 1  192.168.172.57 (192.168.172.57)  0.003 ms  0.007 ms  0.004 ms
 2  192.168.172.22 (192.168.172.22)  0.005 ms  0.010 ms  0.002 ms
 3  192.168.172.50 (192.168.172.50)  0.003 ms  0.010 ms  0.002 ms
client1:~#
client1:~#
client1:~#
client1:~#
```

Figure 13. `traceroute` results from client 1 to client 3

- From the results showed in figure 13, can we confirm if the connectivity between the two clients is OK? Why?
- How many hops were detected in the trace?
- Based on the topology showed in the figure 10 and the results of the `traceroute`, which routers are forwarding the packets from client 1 to client 3?
- Is there another route or path available for the traffic? If so, why do you think this route was not used for the packets?
- Open the trace **'traceroute_c1_to_c3.pcap'** in your computer, which protocols can you find in the capture? Are the packets found in the capture consistent with the behavior of `traceroute` explained before? What is the type of ICMP message? Why?

Practice:

1) From the topology shown in the figure 10 try to figure out how the DNS service works based on the packet capture from a `ping` test on the name of the client3 (trace **'dns_c1_to_c3.pcap'**):

```
client1:~#
client1:~#
client1:~# ping -c 10 client3.aalto.fi
PING client3.aalto.fi (192.168.172.50): 56 data bytes
64 bytes from 192.168.172.50: seq=0 ttl=62 time=0.093 ms
64 bytes from 192.168.172.50: seq=1 ttl=62 time=0.152 ms
64 bytes from 192.168.172.50: seq=2 ttl=62 time=0.210 ms
64 bytes from 192.168.172.50: seq=3 ttl=62 time=0.259 ms
64 bytes from 192.168.172.50: seq=4 ttl=62 time=0.198 ms
64 bytes from 192.168.172.50: seq=5 ttl=62 time=0.209 ms
64 bytes from 192.168.172.50: seq=6 ttl=62 time=0.198 ms
64 bytes from 192.168.172.50: seq=7 ttl=62 time=0.194 ms
64 bytes from 192.168.172.50: seq=8 ttl=62 time=0.074 ms
64 bytes from 192.168.172.50: seq=9 ttl=62 time=0.323 ms

--- client3.aalto.fi ping statistics ---
10 packets transmitted, 10 packets received, 0% packet loss
round-trip min/avg/max = 0.074/0.191/0.323 ms
client1:~#
```

Figure 14. `ping` results from client 1 to client 3 using domain name

2) The trace **'tcp_http.pcap'** contains the capture of the packets obtained by downloading all the images from the site www.google.com . Open the capture in your computer and analyze how the HTTP service works. You can observe for example the TCP sessions, HTTP messages exchanged and other relevant information from this capture.

   [Note: you can use the following resource to learn how to analyze TCP sessions in Wireshark: https://sharkfestus.wireshark.org/assets/presentations/B5%20-%20TCP%20Analysis%20-%20First%20Steps.pdf]

References:

- https://au.int/sites/default/files/documents/31363-doc-session_2-1-_routing_basics.pdf
- https://en.wikipedia.org/wiki/Internet_protocol_suite
- http://www-sop.inria.fr/members/Vincenzo.Mancuso/ReteInternet/12_routing.pdf
- https://docs.oracle.com/cd/E19120-01/open.solaris/819-3000/gcvjj/index.html
- http://www.ece.ubc.ca/~edc/4550.jan2014/lectures/lec17.pdf
- https://www.cisco.com/c/en/us/support/docs/ios-nx-os-software/ios-software-releases-120-mainline/47205-cef-whichpath.html
- https://www.inetdaemon.com/tutorials/internet/tcp/3-way_handshake.shtml
- https://tools.ietf.org/html/rfc792 - RFC792 of the ICMP protocol. 1981.
- Routing in the Internet. 2nd edition, year 2000. Christian Huitema.