Lab1: Basic Networking Tools - Solutions

COM-208: Computer Networks

Welcome to the first Computer Networks lab! The goal of the lab is to get you familiar with basic networking tools. Before you start, please watch the lecture videos and read the "Doing the Labs" document posted on Moodle.

Network interfaces and their names

Every computer in the world has at least one **network interface**. Whenever an entity outside the computer wants to communicate with the computer, it needs to name one of its network interfaces. Different entities use different names to refer to a computer's network interface: the network layer uses **IP addresses**, the link layer uses **MAC addresses**, the computer's operating system (OS) uses **local interface names**.

The **ifconfig** utility lists a computer's network interfaces and displays or updates their configuration.

Type `ifconfig` in the command line and answer the following questions:

- How many network interfaces does your computer have?

```
$ ifconfig -a

ens160: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.93.20.34  netmask 255.255.248.0
                                          broadcast 10.93.23.255
        inet6 fe80::f915:cd29:a812:3bf9  prefixlen 64
                                             scopeid 0x20<link>
        ether 00:50:56:b8:ce:2b  txqueuelen 1000  (Ethernet)
        RX packets 179723  bytes 49342556 (49.3 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 85719  bytes 31695783 (31.6 MB)
```

```
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 110162  bytes 370030090 (370.0 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 110162  bytes 370030090 (370.0 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

This computer has two interfaces: a wired Ethernet interface with local name `ens160`, and what is called a **loopback** interface, with local name `lo` (more on this in a moment).

If you are using an INF3 computer, the output may look like the following:

```
$ ifconfig -a

eno2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 128.178.158.142  netmask 255.255.255.0
                                        broadcast 128.178.158.255
        inet6 fe80::a6bb:6dff:fe4f:c56b  prefixlen 64
                                            scopeid 0x20<link>
        ether a4:bb:6d:4f:c5:6b  txqueuelen 1000  (Ethernet)
        RX packets 1093229  bytes 580963781 (580.9 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 56084  bytes 12509105 (12.5 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 20  memory 0xe4400000-e4420000

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 19946  bytes 2182828 (2.1 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 19946  bytes 2182828 (2.1 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlo1: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether 3c:58:c2:30:70:c8  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
```

```
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

This computer has three interfaces: a wired Ethernet interface named `eno2`, a wireless Ethernet interface named `wlo1`, and a loopback interface named `lo`.

- Can you guess why it has more than one?

If a computer is connected to the network through multiple network links, then it has one network interface for each link. E.g., the INF3 computer is connected through a wired Ethernet link, as well as a wireless Ethernet link.

The loopback interface is what we call a **virtual** network interface. This means that it is not associated with an actual physical link. It is typically used for testing and debugging, and when a process running locally on the computer wants to communicate with another process also running locally on the computer. In the latter case, there is no need to communicate through a "normal" network interface, associated with an actual physical link, since both processes are running on the same computer.

- What is the IP address of each interface?

In the vdi computer, the IP addresses are 10.93.20.34 for `ens160` and 127.0.0.1 for the loopback interface.

In the INF3 computer, the IP addresses are 128.178.158.142 `eno2` and 127.0.0.1 for the loopback interface. The wireless interface, `wlo1`, has no IP address, which means that it cannot be used for Internet communication.

- What is the MAC address of each interface?

In the vdi computer, `ens160`'s MAC address is 00:50:56:b8:ce:2b.

In the INF3 computer, the MAC addresses are a4:bb:6d:4f:c5:6b for `eno2` and 3c:58:c2:30:70:c8 for `wlo1`. Note that `wlo1` does have a MAC address, even if it is not up. This is because, as we will see during the semester, a MAC address is an *inherent* property of a network interface that is associated with a physical link; so, it is there, whether the network interface is actually connected to a network or not. In contrast, an IP address is a property that is *assigned* to a network interface; so, a network interface may not have an IP address associated with it yet.

The loopback interfaces do not have MAC addresses.

- Why could it be that some interfaces do not have a MAC address? (This is a tough question. It's normal if you don't know how to approach it yet. Come back to it at the end of the lab, after having worked on the Internet layers.)

Recall that the role of the link layer is to get a packet across a single link. A virtual interface (like the loopback interface) does not have a MAC address, because it is not associated with an actual physical link, so it does not need to support a link-layer interface. Think about what happens when two processes running locally on your computer communicate over the loopback interface. Their packets do not cross any network link, so, they are not processed by any packet switch, and they do not need to carry any link-layer header.

DNS names and IP addresses

Humans use special names, called **DNS names**, to refer to computers (more precisely, to the network interfaces of computers). When you instruct your computer to communicate with a remote computer that has a given DNS name, your computer translates, under the covers, the given DNS name to an IP address.

The **host** utility helps you map DNS names to IP addresses. E.g., if you type `host` *target* in the command line, where *target* is a DNS name or IP address, that will display the IP address(es) and potentially other DNS names of the target network interface.

Use the `host` utility to answer the following questions:

- What are the IP addresses of www.epfl.ch?

```
$ host www.epfl.ch

www.epfl.ch is an alias for www.epfl.ch.cdn.cloudflare.net.
www.epfl.ch.cdn.cloudflare.net has address 104.20.228.42
www.epfl.ch.cdn.cloudflare.net has address 104.20.229.42
www.epfl.ch.cdn.cloudflare.net has address 172.67.2.106
www.epfl.ch.cdn.cloudflare.net has IPv6 address
                                          2606:4700:10::6814:e42a
www.epfl.ch.cdn.cloudflare.net has IPv6 address
                                          2606:4700:10::ac43:26a
www.epfl.ch.cdn.cloudflare.net has IPv6 address
                                          2606:4700:10::6814:e52a
```

The IP addresses of www.epfl.ch are 104.20.228.42, 104.20.229.42, and 172.67.2.106. (There are also 3 IPv6 addresses, but we have not talked about IPv6 yet.)

- Why could it be that www.epfl.ch maps to more than one IP addresses?

For **fault tolerance** and/or what we call **load balancing**.

You already know that a computer can be connected through multiple network links, hence have multiple network interfaces and IP addresses: if one link fails, then the computer can be reached through another.

Moreover, popular services are typically **replicated** over multiple computers. In our particular example, when a user types in their web browser www.epfl.ch, there are multiple computers running web-server processes that can serve that user. Each of these computers has, of course, its own IP address.

Such replication provides fault tolerance: if one computer fails, then another can serve. It also provides load balancing: different users can be served by different computers, such that the load if not concentrated on one computer.

- What is the IP address of www.google.com?

```
$ host www.google.com
```

```
www.google.com has address 172.217.168.68
www.google.com has IPv6 address 2a00:1450:400a:801::2004
```

The IP address of www.google.com is 172.217.168.68.

- Answer the same question again in an hour or so. Has anything changed? If so, what could be the reason for the change?

This is another approach to load balancing:

We said above that a service may be replicated over multiple computers, which means that a DNS name, like www.epfl.ch, maps to multiple IP addresses, each belonging to a different computer serving the target DNS name.

In the above example, when you typed `host www.epfl.ch`, you were given the IP addresses of all the computers serving www.epfl.ch.

An alternative is to be given the IP address of only one computer: the one that you should use at the moment to access the target DNS name, because it happens to be the closest to you, or it happens to be the least busy (among the ones serving the target DNS name). At a later moment, you may be given a different IP address that belongs to a different computer, which happens to be the least busy at that moment.

Google recently implemented this alternative, and this is why you were given only one IP address for www.google.com. Things were different a few years back, when we typed the same command:

```
$ host www.google.com
www.google.com has address 173.194.40.32
www.google.com has address 173.194.40.33
www.google.com has address 173.194.40.34
www.google.com has address 173.194.40.35
www.google.com has address 173.194.40.36
www.google.com has address 173.194.40.37
www.google.com has address 173.194.40.38
www.google.com has address 173.194.40.39
www.google.com has address 173.194.40.40
www.google.com has address 173.194.40.41
www.google.com has address 173.194.40.46
```

Reachability

The **ping** utility helps you check whether a remote computer is "reachable" from your computer. E.g., if you type `ping` *target* in the command line, where *target* is a DNS name or IP address, that will tell you whether your computer can reach the target network interface.

When one computer pings another, it sends to it a small packet, requesting a response, and it measures the time it takes from the moment each request is sent until the corresponding response is received. This time has a special name: it is called the **round-trip time (RTT)** between your computer and the target.

Use `ping` to answer the following questions:

- Are the following computers (more precisely, network interfaces) reachable from yours: www.epfl.ch, www.20min.ch, www.swisscom.ch, 8.8.8.8, www.microsoft.com, www.auth.gr, en.sjtu.edu.cn, www.adelaide.edu.au?

```
$ ping www.epfl.ch

PING www.epfl.ch.cdn.cloudflare.net (104.20.229.42)
                                              56(84) bytes of data.
64 bytes from 104.20.229.42 (104.20.229.42):
                                    icmp_seq=1 ttl=57 time=1.34 ms
64 bytes from 104.20.229.42 (104.20.229.42):
                                    icmp_seq=2 ttl=57 time=1.33 ms
64 bytes from 104.20.229.42 (104.20.229.42):
                                    icmp_seq=3 ttl=57 time=1.40 ms

--- www.epfl.ch.cdn.cloudflare.net ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2002ms
rtt min/avg/max/mdev = 1.338/1.362/1.406/0.052 ms
```

```
$ ping www.20min.ch

PING www.20min.ch (13.224.95.84) 56(84) bytes of data.
64 bytes from server-13-224-95-84.zrh50.r.cloudfront.net
                 (13.224.95.84): icmp_seq=1 ttl=244 time=3.31 ms
64 bytes from server-13-224-95-84.zrh50.r.cloudfront.net
                 (13.224.95.84): icmp_seq=2 ttl=244 time=3.36 ms
64 bytes from server-13-224-95-84.zrh50.r.cloudfront.net
                 (13.224.95.84): icmp_seq=3 ttl=244 time=3.31 ms
```

```
--- www.20min.ch ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2002ms
rtt min/avg/max/mdev = 3.316/3.332/3.365/0.052 ms
```

```
 $ ping www.swisscom.ch

PING www-swisscom-ch.hdb-cs04.ellb.ch (195.186.208.154)
                                          56(84) bytes of data.

--- www-swisscom-ch.hdb-cs04.ellb.ch ping statistics ---
3 packets transmitted, 0 received, 100% packet loss, time 2040ms
```

```
 $ ping 8.8.8.8

PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8:
                                    icmp_seq=1 ttl=117 time=3.97 ms
64 bytes from 8.8.8.8:
                                    icmp_seq=2 ttl=117 time=3.98 ms
64 bytes from 8.8.8.8:
                                    icmp_seq=3 ttl=117 time=3.97 ms

--- 8.8.8.8 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2001ms
rtt min/avg/max/mdev = 3.972/3.980/3.989/0.007 ms
```

```
 $ ping www.microsoft.com

PING e13678.dspb.akamaiedge.net (23.54.112.217) 56(84) bytes of
                                                              data.
64 bytes from a23-54-112-217.deploy.static.akamaitechnologies.com
                (23.54.112.217): icmp_seq=1 ttl=57 time=3.22 ms
64 bytes from a23-54-112-217.deploy.static.akamaitechnologies.com
                (23.54.112.217): icmp_seq=2 ttl=57 time=3.23 ms
64 bytes from a23-54-112-217.deploy.static.akamaitechnologies.com
                (23.54.112.217): icmp_seq=3 ttl=57 time=3.28 ms

--- e13678.dspb.akamaiedge.net ping statistics ---
```

```
3 packets transmitted, 3 received, 0% packet loss, time 2002ms
rtt min/avg/max/mdev = 3.229/3.249/3.280/0.022 ms
```

```
 $ ping www.auth.gr

PING www.ccf.auth.gr (155.207.1.12) 56(84) bytes of data.
64 bytes from www.ccf.auth.gr (155.207.1.12):
                                    icmp_seq=1 ttl=52 time=43.0 ms
64 bytes from www.ccf.auth.gr (155.207.1.12):
                                    icmp_seq=2 ttl=52 time=43.0 ms
64 bytes from www.ccf.auth.gr (155.207.1.12):
                                    icmp_seq=3 ttl=52 time=43.0 ms

--- www.ccf.auth.gr ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2001ms
rtt min/avg/max/mdev = 43.022/43.046/43.066/0.018 ms
```

```
 $ ping en.sjtu.edu.cn

PING en.sjtu.edu.cn (202.120.2.113) 56(84) bytes of data.
64 bytes from 202.120.2.113 (202.120.2.113):
                                    icmp_seq=1 ttl=39 time=225 ms
64 bytes from 202.120.2.113 (202.120.2.113):
                                    icmp_seq=2 ttl=39 time=225 ms
64 bytes from 202.120.2.113 (202.120.2.113):
                                    icmp_seq=3 ttl=39 time=225 ms

--- en.sjtu.edu.cn ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2001ms
rtt min/avg/max/mdev = 225.300/225.367/225.465/0.070 ms
```

```
 $ ping www.adelaide.edu.au

PING online-media.adelaide.edu.au (129.127.149.1) 56(84) bytes of
                                                              data.
64 bytes from online-media.adelaide.edu.au (129.127.149.1):
                                    icmp_seq=1 ttl=237 time=253 ms
64 bytes from online-media.adelaide.edu.au (129.127.149.1):
```

```
                                        icmp_seq=2 ttl=237 time=253 ms
64 bytes from online-media.adelaide.edu.au (129.127.149.1):
                                        icmp_seq=3 ttl=237 time=253 ms

--- online-media.adelaide.edu.au ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2001ms
rtt min/avg/max/mdev = 253.316/253.341/253.364/0.581 ms
```

All computers are reachable through `ping`, except www.swisscom.ch.

- Notice the RTT reported by `ping` for each target computer. Do you see a pattern? Which targets tend to have longer RTTs?

In general, RTT increases with the physical distance to the target. For example, www.epfl.ch (which should be somewhere in Switzerland) replies much faster than www.adelaide.edu.au (which should be somewhere in Australia).

- If you let it run, `ping` makes many efforts to reach each target. As you can see, the RTT (to the *same* target) changes with every effort. What could possibly be the reason for this change?

It could be that the path from your computer to the target changes, or that the traffic on this path changes, making communication over the path faster/slower.

- At least one of the targets should be unreachable through `ping`. Try to reach it by typing its DNS name in your web browser. How could it be that the same target is unreachable through `ping` but reachable through your browser?

Indeed, www.swisscom.ch is unreachable through `ping` but reachable through a web browser. This could happen because a packet switch inside the Swisscom network inspects packets and drops the ones that are carrying ping messages, while it accepts and forwards those that are carrying web (HTTP) messages.

Network paths and packet switches

When two computers (end-systems) communicate with each other over the Internet, their communication traverses multiple **packet switches**. There are two general types of packet switches on the Internet: link-layer switches and network-layer switches (the latter are also called **routers**).

The **traceroute** utility lists the routers that are located between your computer and a remote one. E.g., if you type `traceroute` *target* in the command line, where *target* is a DNS name or IP address, that will display a list of router DNS names and/or IP addresses and the RTTs that were measured between your computer and each router.

(The idea behind how traceroute works is a little wonder, and we will explore it later in the semester. If you feel curious, you can already google it.)

Use `traceroute` to answer the following questions:

- How many routers are there between your computer and www.mcgill.ca?

```
$ traceroute www.mcgill.ca

traceroute to www.mcgill.ca (132.216.177.160), 30 hops max,
                                                  60 byte packets
 1  nx-srv1-l3-1-v812.epfl.ch (10.93.16.2)
                                0.390 ms  0.724 ms  0.831 ms
 2  backbone-nx-srv1-l3-1-26.epfl.ch (10.0.2.26)
                                0.141 ms  0.310 ms  0.335 ms
 3  c6-ext-cv-backbone-97.epfl.ch (10.0.2.97)
                                0.290 ms  0.327 ms  0.316 ms
 4  swiel2.epfl.ch (192.33.209.33)
                                0.716 ms  0.739 ms  0.720 ms
 5  swige3-100ge-0-0-1-1.switch.ch (130.59.36.82)
                                1.704 ms  1.736 ms  1.713 ms
 6  swice1-100ge-0-1-0-6.switch.ch (130.59.38.193)
                                1.603 ms  1.613 ms  1.925 ms
 7  switch.mx1.gen.ch.geant.net (62.40.124.21)
                                1.532 ms  1.561 ms  1.463 ms
 8  ae6.mx1.par.fr.geant.net (62.40.98.183)
                                8.834 ms  8.865 ms  8.838 ms
 9  canarie-bckp-gw.mx1.par.fr.geant.net (62.40.124.226)
                                91.108 ms  91.108 ms  91.117 ms
10  otwa3rtr1.canarie.ca (205.189.32.178)
                                93.982 ms  93.884 ms  93.923 ms
```

```
11  205.189.32.57 (205.189.32.57)
                                93.896 ms  93.527 ms  93.566 ms
12  * * *
13  * * *
14  * * *
15  mcgill-canet-membre1.risq.net (206.167.128.50)
                                96.875 ms  96.875 ms  96.873 ms
16  * * *
17  * * *
18  * * *
19  * * *
20  www.mcgill.ca (132.216.177.160)
                                98.335 ms  98.355 ms  98.275 ms
```

There are 20 routers.

If the x-th row is "* * *", that means that the xth router between your computer and the target could not be identified.

Note that your traceroute output may slightly differ from the above, as network paths may change.

- How many of these routers are inside the EPFL network? How many, would you guess, are inside EPFL's Internet Service Provider (ISP)?

We can guess by looking at the DNS names of the routers:

The DNS names of the first 4 routers have suffix epfl.ch, so they should be inside the EPFL network.

The first router that is after the EPFL routers (so, the 5th router) should belong to EPFL's ISP. Moreover, we notice that the DNS name of the 6th router has the same suffix as that of the 5th router (switch.ch), so, that should also belong to EPFL's ISP. (EPFL's ISP is called SWITCH, by the way.)

- Between which of these routers, do you think, your packets cross the Atlantic?

Observe how RTT "jumps" from 8.8ms to 91.1ms between routers 8 and 9.

```
 8  ae6.mx1.par.fr.geant.net (62.40.98.183)
                     8.834 ms  8.865 ms  8.838 ms
 9  canarie-bckp-gw.mx1.par.fr.geant.net (62.40.124.226)
                     91.108 ms  91.108 ms  91.117 ms
```

It makes sense to assume that these two routers are located at opposite ends of a transatlantic link.

- Now `traceroute` to www.google.com. Does the network path from your computer to www.mcgill.ca overlap with the path from your computer to www.google.com?

```
$ traceroute www.google.com

traceroute to www.google.com (172.217.168.4), 30 hops max,
                                          60 byte packets
 1  nx-srv1-l3-1-v812.epfl.ch (10.93.16.2)
                                  0.468 ms  0.670 ms  0.745 ms
 2  backbone-nx-srv1-l3-1-26.epfl.ch (10.0.2.26)
                                  0.151 ms  0.316 ms  0.394 ms
 3  c6-ext-cv-backbone-97.epfl.ch (10.0.2.97)
                                15.245 ms  15.271 ms  15.300 ms
 4  swiel2.epfl.ch (192.33.209.33)
                                  0.728 ms  0.714 ms  0.719 ms
 5  swils1-100ge-0-0-0-0.switch.ch (130.59.38.54)
                                  1.227 ms  1.257 ms  1.238 ms
 6  swizh3-100ge-0-0-0-1.switch.ch (130.59.36.94)
                                  3.463 ms  3.436 ms  3.465 ms
 7  72.14.195.4 (72.14.195.4)
                                  4.059 ms  4.045 ms  4.066 ms
 8  74.125.243.145 (74.125.243.145)
                                  4.002 ms  4.019 ms  3.997 ms
 9  172.253.50.17 (172.253.50.17)
                        5.030 ms 172.253.50.19 (172.253.50.19)
                                           4.766 ms  4.771 ms
10  zrh11s03-in-f4.1e100.net (172.217.168.4)
                                  3.978 ms  4.012 ms  3.994 ms
```

Yes, the network paths from our computer to www.mcgill.ca and to www.google.com overlap in routers 1-4. This indicates that the two paths are the same inside the EPFL network, but then diverge.

- Traceroute again to www.google.com in an hour or so. Is the output (the sequence of routers) the same as before? What does the answer say about the network path from your computer to www.google.com?

```
$ traceroute www.google.com

traceroute to www.google.com (172.217.168.4), 30 hops max,
                                                    60 byte packets
 1  nx-srv1-l3-1-v812.epfl.ch (10.93.16.2)
                                     0.772 ms  1.359 ms  1.391 ms
 2  backbone-nx-srv1-l3-1-26.epfl.ch (10.0.2.26)
                                     0.159 ms  0.309 ms  0.330 ms
 3  c6-ext-cv-backbone-97.epfl.ch (10.0.2.97)
                                     0.277 ms  0.299 ms  0.319 ms
 4  swiel2.epfl.ch (192.33.209.33)
                                     0.718 ms  0.741 ms  0.710 ms
 5  swils1-100ge-0-0-0-0.switch.ch (130.59.38.54)
                                     0.855 ms  0.881 ms  0.868 ms
 6  swizh3-100ge-0-0-0-1.switch.ch (130.59.36.94)
                                     3.523 ms  3.540 ms  3.468 ms
 7  72.14.195.4 (72.14.195.4)
                                     4.159 ms  4.160 ms  4.127 ms
 8  74.125.243.161 (74.125.243.161)
                                     4.952 ms  4.896 ms
                          74.125.243.145 (74.125.243.145)  4.012 ms
 9  172.253.50.19 (172.253.50.19)
                                     4.927 ms  4.955 ms  4.924 ms
10  zrh11s03-in-f4.1e100.net (172.217.168.4)
                                     3.999 ms  3.983 ms  3.991 ms
```

The output changed, in particular, the 8th and 9th rows. This indicates that the path from our computer to www.google.com changed, or that there are multiple paths, and our traffic followed a different one each time.

- There exist **traceroute servers** that allow you to traceroute from them to any other computer in the world. For example, this one, or this one. Traceroute from a traceroute server to your computer, then from your computer to that server. Are the two paths symmetric? You can find other traceroute servers at www.traceroute.org and play around with them, e.g., traceroute from one server to another, check if the paths are symmetric, and try to guess the geographic locations of the servers.

The output typically indicates that the paths traversed by a packet from one computer to another and vice versa are not necessarily symmetrical.

Remote connection

The **ssh** utility enables you to establish a secure communication channel between your computer and a remote one. E.g., you can type `ssh` *username@target* in the command line, where *target* is the remote computer's DNS name or IP address.

Use `ssh` to connect to one of the computers in INF3, e.g., icin3pc01.epfl.ch, using your gaspar username. If you are working through vdi, you may be unsuccessful, but this does not really affect the next steps of the lab; just leave the `ssh` command running, waiting to be prompted for your gaspar password.

```
$ ssh mygasparname@icin3pc42.epfl.ch
```

If you were successful, you are now logged into the remote INF3 computer, until you run `exit`.

Port numbers

A computer's Operating System (OS) assigns a **port number** to every process running on the computer. Certain processes are always assigned the *same* port number on every computer where they run. Differently said, certain port numbers are universally reserved for specific processes. In any Unix-like environment, the file `/etc/services` lists these special processes and port numbers.

Either over the ssh connection you previously established or, if you were unsuccessful, on your own computer, open `/etc/services` and take a look at the contents, e.g., by typing `cat /etc/services` in the command line.

- Do you recognize any of the special processes (also called "services") listed on the left?

You should recognize at least http (the "World Wide Web", as the comments helpfully indicate).

- Which is the port number that is reserved for ssh-server processes? What about web-server processes and mail-server processes?

For ssh-server processes:

```
$ grep "ssh" /etc/services
```

```
ssh             22/udp          # SSH Remote Login Protocol
ssh             22/tcp          # SSH Remote Login Protocol
```

For web-server processes:

```
$ grep "http" /etc/services
```

```
http            80/tcp          www             # WorldWideWeb HTTP
https           443/tcp               # http protocol over TLS/SSL
http-alt        8080/tcp        webcache     # WWW caching service
http-alt        8080/udp
```

For mail-server processes:

```
$ grep "smtp" /etc/services
```

```
smtp            25/tcp          mail
```

- Next to each port number, there is a "udp" or "tcp" label. Do you remember what these are from the video lectures?

They are transport-layer technologies/protocols. Their role is to transfer a packet from one end-system to another.

- Interspersed with the protocol names and port numbers are some human names. What do you think those are? Do you recognize any of them?

In MacOS, this file typically includes the names of the people who registered the various port numbers, who typically also invented the corresponding protocols. Notice "Tim Berners-Lee" next to http.

This is typically not the case in Ubuntu – other than protocol names and port numbers, the file contains only human-friendly descriptions of the protocols.

Active "communication sessions"

The **netstat** utility displays the contents of various network-related data structures that are stored in your computer. E.g., if you type `netstat -t` in the command line, that will display the list of "communication sessions" that are active between your computer and remote computers (actually, they are TCP connections, but you are not supposed to know about them yet...)

- The "Local Address" column lists processes that are running in the application layer of your computer. Notice that the names of all (or most of) these processes share a common prefix. Why is that? What does this prefix correspond to? (Did you run into it earlier in this lab?)

```
$ netstat -t

Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address
                                                                State
tcp        0      0 localhost:6101          localhost:32944
                                                                ESTABLISHED
tcp        0      0 localhost:53366         localhost:34451
                                                                ESTABLISHED
tcp        0      0 localhost:6101          localhost:32796
                                                                ESTABLISHED
tcp        0      0 localhost:54152         localhost:34451
                                                                ESTABLISHED
tcp        0      0 localhost:51522         localhost:38337
                                                                ESTABLISHED
tcp        0     39 CO-IN-SC-13:22443       itvdigw02.epfl.ch:29570
                                                                ESTABLISHED
tcp        0      0 CO-IN-SC-13:981         files8.epfl.ch:nfs
                                                                ESTABLISHED
tcp        0      0 CO-IN-SC-13:44464       dashboard.snapcra:https
                                                                ESTABLISHED
tcp        0      0 CO-IN-SC-13:60970       api.snapcraft.io:https
                                                                ESTABLISHED
tcp        0      0 CO-IN-SC-13:34424       api.snapcraft.io:https
                                                                ESTABLISHED
tcp        0      0 localhost:32944         localhost:6101
                                                                ESTABLISHED
tcp        0      0 CO-IN-SC-13:52254       ad2.epfl.ch:3268
                                                                ESTABLISHED
```

```
tcp        0      0 localhost:38337         localhost:51522
                                                    ESTABLISHED
tcp        0      0 CO-IN-SC-13:37522       oscp-router02.gno:https
                                                    ESTABLISHED
tcp      128      0 localhost:32796         localhost:6101
                                                    ESTABLISHED
tcp        0      0 CO-IN-SC-13:33354       oscp-router03.gno:https
                                                    ESTABLISHED
tcp        0      0 localhost:53364         localhost:34451
                                                    ESTABLISHED
tcp        0      0 CO-IN-SC-13:52754       dashboard.snapcra:https
                                                    ESTABLISHED
tcp        0      0 CO-IN-SC-13:42316       ad5.epfl.ch:ldap
                                                    ESTABLISHED
tcp6       0      0 localhost:34451         localhost:53364
                                                    ESTABLISHED
tcp6       0      0 localhost:34451         localhost:54152
                                                    ESTABLISHED
tcp6       0      0 CO-IN-SC-13:44230       itvdibroker01.epfl:4002
                                                    ESTABLISHED
tcp6       0      0 localhost:34451         localhost:53366
                                                    ESTABLISHED
```

In our computer, most local processes have names that start with `localhost` or `CO-IN-SC-13`.

As we said in class, the first part of a process's name identifies a network interface that belongs to the computer where the process is running. Since all the local processes are, of course, running on our computer, the first part of their names identifies a network interface of our computer.

- The "Foreign Address" column lists all the processes that are running in the application layer of a remote computer that your computer is communicating with. Can you tell which one corresponds to INF3 computer you have ssh-ed into? (Reminder: if you are working through vdi, you may have been unable to connect to an INF3 computer, in which case `ssh` is still waiting to be prompted for a password; do not kill it and do not exit the password prompt.)

The new connection is:

```
tcp        0      0 CO-IN-SC-13:47210       icin3pc42.epfl.ch:ssh
                                                          ESTABLISHED
```

Layers and headers

The Internet architecture operates in **layers**. As a result, a packet that traverses the Internet looks, in a way, like an onion: On the "outside," it is "wrapped up" in a link-layer header (which can be understood only by the link layer of computers and packet switches). If we "peel away" the link-layer header, we will find a network-layer header (which can be understood only by the network layer of computers and packet switches). If we also peel away the network-layer header, we will find a transport-layer header. And if we peel that away, too, we will find the application-layer header and data, which is the actual message that this packet is carrying.

So, if we look inside an Internet packet, we will find a lot more information than the application-layer message that the packet is carrying: we will find meta-data, in the form of headers, which are needed by the various Internet layers in order to get the message from its source to its destination.

We will now use an application called **Wireshark** to do precisely that: look inside Internet packets. To get started, do the following:

- Start your web browser and clear its cache. If you are using Firefox, click on the ≡ symbol on the upper-right, go to Preferences ⟶ Privacy & Security ⟶ Cookies and Site Data ⟶ Clear Data.
- Start the Wireshark tool, e.g., by typing `wireshark` in the command line. You should see a list of your computer's network interfaces. Identify the one whose packets you will capture. If you are working through an INF3 computer or connected through vdi, you want to capture packets from your ethernet network interface. If you are working on a wirelessly connected computer, you want to capture packets from your WiFi interface.
- Start a capture by double-clicking on the target network interface. You should see data rolling inside the top part of your Wireshark window. These are the packets that are departing from and arriving at your network interface. They most likely make no sense, and that's normal (by the end of the course, they will).
- Use your web browser to visit www.mit.edu.
- Stop capturing packets when the web page is fully loaded, by clicking on the square red button at the left of the top menu.
- Right underneath the top menu, you can specify a filter that you want to apply to the packets that you see.

Answer the following questions:

- What messages were exchanged at the **application layer**, i.e., between your web browser and the MIT web server? The communication protocol used between web browsers and web servers is HTTP (Hypertext Transfer Protocol), so type `http` in the filter line to see all the packets carrying HTTP messages. What information are these messages carrying?

  As we can see in the "Info" column of Fig 1, two types of application-layer messages were exchanged:

  - GET HTTP/1.1 requests, sent by the web browser, to get content from the web server;
  - HTTP/1.1 responses, sent by the web server providing the requested content.

- Which technology/communication protocol was used at the **transport layer**? There are two of them, TCP (Transmission Control Protocol) and UDP (User Datagram Protocol), and you need to figure out which one was used. To answer, click on one of the packets in the top section of your Wireshark window, then check the detailed information about this packet that appears in the middle section of your window. You should see information about each layer. Near the bottom, you should see a line that refers to the application layer (it says "Hypertext Transfer Protocol"). What does the line on top of that say?

  TCP.

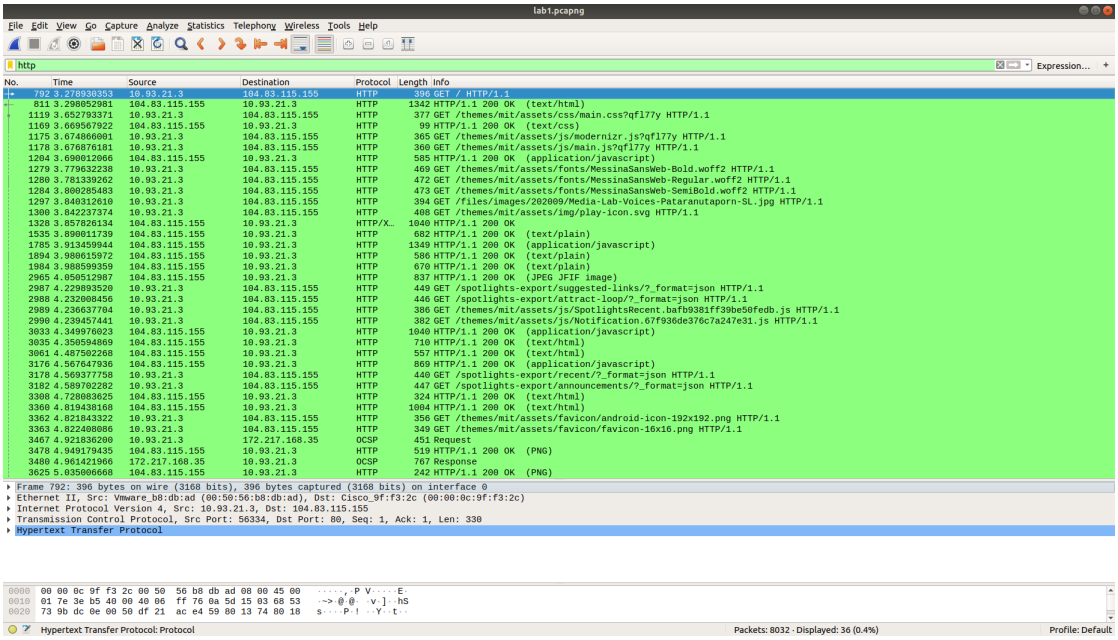

Figure 1: Messages containing the HTTP protocol

- What messages were exchanged at the **transport layer**, i.e., between the transport layer on your computer and the transport layer on the computer running the MIT web server? This is a little bit trickier to answer. First of all, you need enter in the filter line the correct transport-layer technology/communication protocol, which you figured out in the previous question. But if you do just that, then you will see ALL the messages exchanged by your computer using that protocol, whereas you only want the ones exchanged with the computer running the MIT web server. So, you need to add something more to the filter. Poke around a bit in Wireshark documentation on how to specify filters, and you should figure it out.

  A key point here is that the application-layer messages and the transport-layer messages were not carried in separate Internet packets. Rather, the same packets carried BOTH transport-layer and application-layer information, but the transport-layer information was stored inside the transport-layer header of each packet, whereas the application-layer information was stored inside the application-layer header and data.

  The filter we need to apply is `tcp and (ip.src==104.83.115.155 or ip.dst==104.83.115.155)`, because it displays all the TCP messages sent or received by the computer running the MIT web server (which has IP address 104.83.115.155). You may need to change this IP address to

a different one, if you happened to access the MIT web site through a different computer.

As we can see in Fig 2, the messages exchanged at the transport layer are:

- SYN (to initiate a TCP connection),
- SYN ACK,
- data packets,
- FIN (to end a connection),
- FIN ACK.

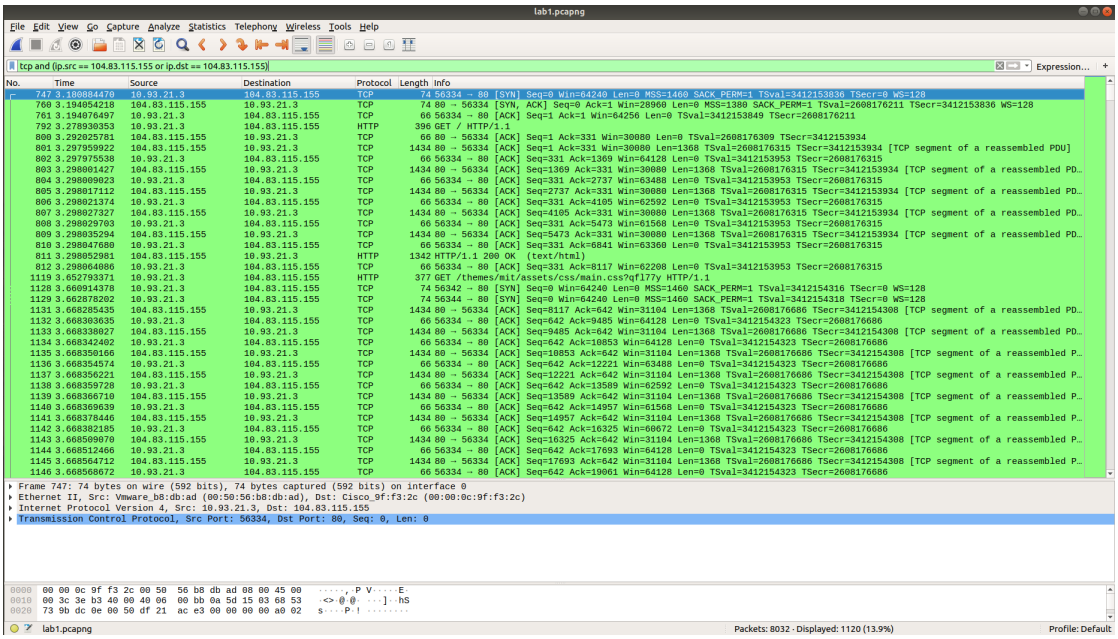

Figure 2: TCP messages exchanged with www.mit.edu

Now we will examine the concept of **encapsulation**, meaning that each message encapsulates a message that belongs to a higher layer. E.g., a network-layer message consists of a network-layer header plus a transport-layer message, which consists of a transport-layer header plus an application-layer message, which consists of an application-layer header plus data.

Display the messages that were exchanged at the application layer (the HTTP messages) and click on one of them:

- How many bytes does the HTTP message contain? To answer, check the packet details in the middle section of your Wireshark window. Look at the transport-

layer information and, in particular, the `Len` field, which specifies the size of the application-layer message that is encapsulated inside the transport-layer message.

- How many bytes do the transport-layer and network-layer headers add to the HTTP message?
- How many bytes does the link layer add?

> See Fig 3.
>
> This HTTP message contains 330 bytes, as shown in the `Len` field of the Transmission Control Protocol.
>
> The TCP header adds 32 bytes, and the IP header 20 bytes.
>
> The link layer adds 14 bytes.

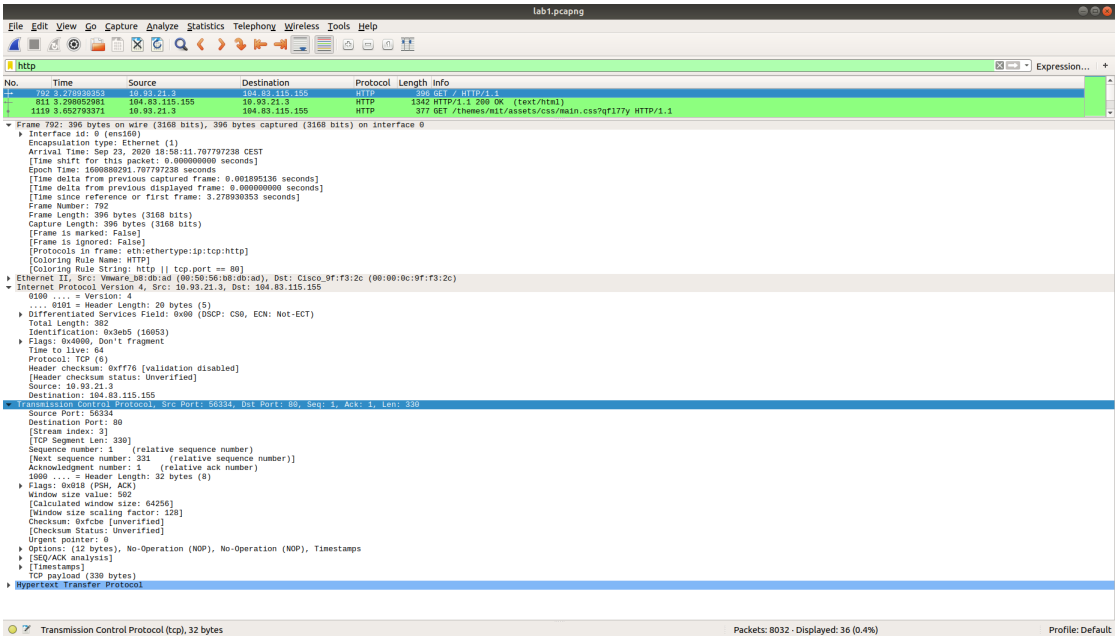

Figure 3: HTTP message

Finally, go to Statistics ⟶ Conversations window and check the "Limit to display filter" checkbox. This shows you a summary of the communications that your computer participated in in order to download the MIT web page. Can you tell how many remote computers (servers) your computer had to access?

See Fig 4.

According to this figure you accessed two servers, as there are two unique IP addresses in the "Address B" column. However, in reality, only one of them provided website content, the other one provided security related information that we will discuss later in the semester.

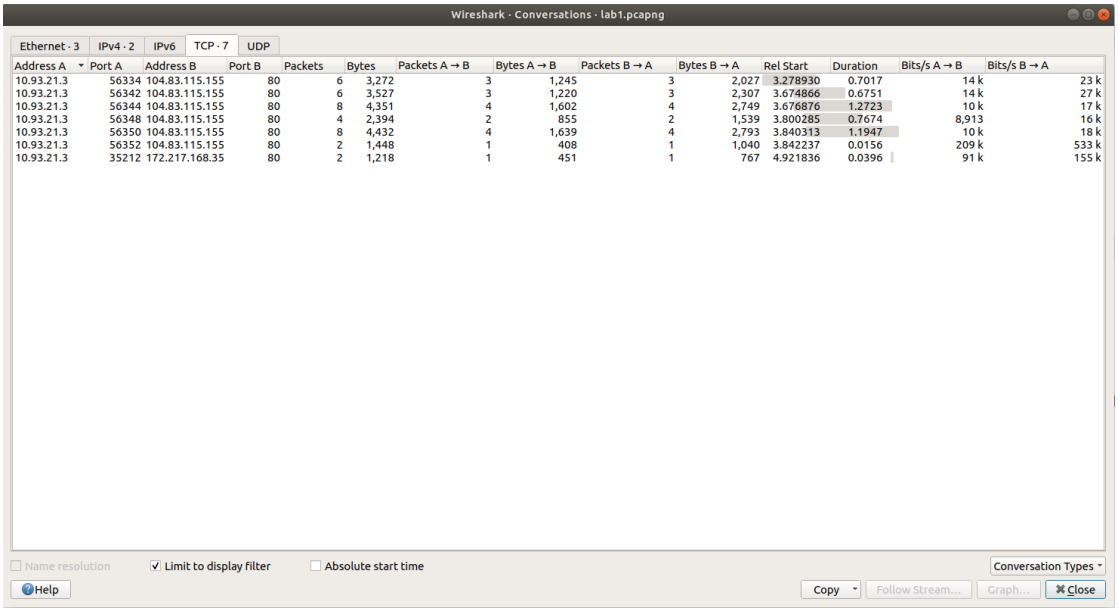

Figure 4: TCP conversations

This is it! If you feel like it, use Wireshark to capture your computer's communications (outside the lab). You may be surprised...