Name 1:
Name 2:

COMPUTER NETWORKING

LAB EXERCISES (TP) 2
L2 V.S. L3, NAT, PHYSICAL CONNECTION, AND TROUBLESHOOTING
With Solutions

October 9, 2019

Abstract

In this Lab you will work with the virtual environment introduced in Lab 1. First you will see the different behavior of networking devices that work on layer 2 and layer 3; then you will configure your virtual network to be able to access the Internet; and finally you will connect one physical machine to another one and use its Internet connection.

1 PREPARING THE LAB

1.1 LAB REPORT

Type your answers in this document. We recommend you use Adobe Reader XI to open this PDF. When you finish, save the report and upload it on Moodle. Don't forget to write your names on the first page of the report. **The deadline is Wednesday, October 23, 23:59:59**

1.2 SET UP

In this Lab, you will work with the same virtual machine that you created in Lab 1. Copy the **lab2 resources** folder from Moodle into the shared folder of your VM before starting the lab. In Lab 1 you have learned how to create a shared folder in a VM.

2 LAYER 2 VS. LAYER 3 NETWORKING

The aim of this section is to illustrate the difference between networking devices that work at layer 2 and layer 3.

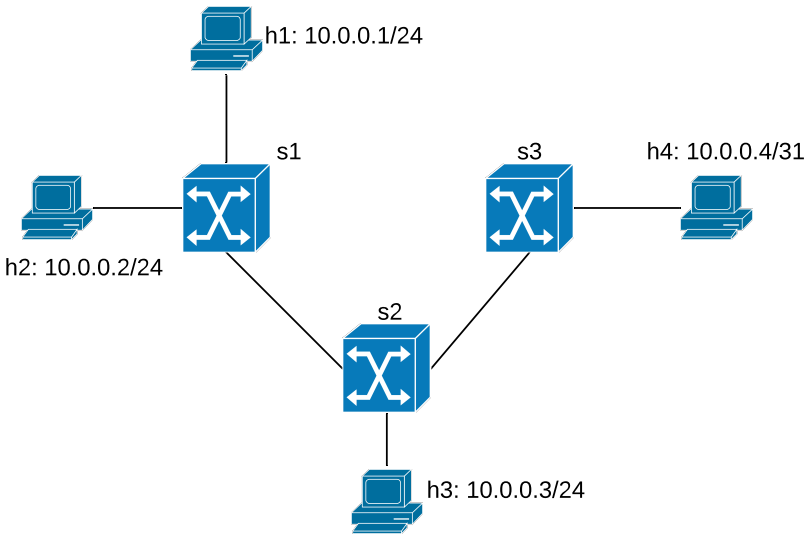

Figure 1: Loop-free network configuration with three switches

2.1 USING A SWITCH AS A NETWORKING DEVICE

A switch is a MAC-layer device which expands a LAN by making forwarding decision based on destination MAC-address. In this section you will learn how they work.

Open a terminal in your VM and run the script `topo1.py` as root (*password: lca2*), which should be located in the shared folder on the Desktop. If not, refer to Section 1.2.

```
# sudo python topo1.py
```

This will create the network described in Figure 1, and redirect you to the Mininet CLI. Additionally, one terminal will appear for each of the four hosts. The four new terminals will be labeled (`h1, h2, h3, h4`) for convenience. `h1, h2` and `h3` should be configured with the `10.0.0.0/24` subnet with the fourth byte of their IP address being `1, 2` and `3`, respectively. Also, `h4` should have the IP address 10.0.0.4 with the subnet mask of `255.255.255.254`. Additionally, every host is automatically assigned an IPv6 address. **Q1/** How many LANs, with respect to physical point of view, are there in the figure?

Solution. *There is only one LAN as there are only bridges and no router. The indication of "physical point of view" is to emphasize that the host `h4` is not counted as one LAN because it has a different network address. From physical point of view, since there is no router, the number of LANs is one.*

Q2/ What are the IPv6 addresses of all hosts? What kind of addresses are these?

Solution. *They might be different for every machine, but all the hosts should have 64 bits prefix starting with `fe80`.*
`h1-eth0`: $fe80::74d9:b8ff:fef8:73e0$
`h2-eth0`: $fe80::94d4:aeff:fe79:3320$
`h3-eth0`: $fe80::289b:33ff:fec6:4847$
`h4-eth0`: $fe80::0493:8bff:feab:01bb$.
These are link-local addresses, i.e. addresses that can be used for communicating in the same LAN only.

Now, let's test our configuration. Start Wireshark on all four hosts. It will be hard to keep track of which Wireshark window corresponds to which host. One way to do so would be to start Wireshark on the hosts in order, i.e. `h1`, then `h2`, then `h3`, and finally on `h4`. This way the Wireshark windows will be in this same order in the taskbar. Start capturing on all the eth0 interfaces.

```
# wireshark &
```

Try to ping from each host to the others using its IPv6 address by executing the following command:

```
ping6 -I <interface name of host> <IPv6 address of destination>
```

Q3/ Which hosts do not receive ping-reply? Explain.

Solution. *All hosts receive ping-reply. Since each host is using its link-local IPv6 address and all hosts are in the same LAN, it can send ping-request and receive ping-reply from other hosts.*

Now, from terminal of `h1`, ping `h2` using its IPv4 address:

```
# ping h2
```

Q4/ Describe the different types of packets observed on `h1, h2, h3` and `h4`.

Solution. *On `h1` and `h2` we are able to see ICMP echo-request, ICMP echo-reply, ARP requests and ARP replies (if any exists). In `h3` and `h4` we only observed ARP request packets (broadcast), if any exists.*

Now, ping from `h1` to `h3` using IPv4.

Q5/ Compare the packets sent by `h1` to the ones received by `h3`, specifically at source/destination MAC-addresses. Explain the similarities and differences, if any.

Solution. *Traffic is the same, same ethernet header, same source/destination MAC-addresses. The Ethernet bridge does not affect any source/destination MAC-address, it is transparent to the MAC and IP layers.*

Q6/ Ping from `h4` to `h1` using IPv4. Observe the traffic captured and explain your findings.

Solution. *We don't see any packets. h4 will use its network mask on h1's IP address and check if they are in the same subnet. As they are not in the same subnet, h4 will attempt to contact its default gateway to send the packet, and if no default gateway is configured (which is this case), it will not send any packet at all.*

Q7/ Fix the configuration issue with host `h4`. What commands did you execute?

Solution. *`h4` is not the same subnet as the other hosts are. You have to change the subnet mask in `h4`:*

```
ip addr flush dev h4-eth0
ip addr add 10.0.0.4/24 dev h4-eth0
```

Q8/ What is the benefit of IPv6 over IPv4? Explain your answer based on your findings in previous questions.

Solution. *IPV6 hosts usually self-allocate a link-local address automatically. This allows alls hosts that are in the same LAN to communicate using IPv6 without any configuration. In contrast, for the linux distribution we are using, no such thing happens with IPv4 (in contrast, link-local IPv4 addresses may be automatically allocated by Windows systems).*

Exit Mininet and clean up the topology before going to next subsection:

```
mininet> exit
# mn -c
```

2.2 CONFIGURE A SWITCH TO HANDLE LOOPS

The goal of this subsection is to configure a LAN with loops. Similarly to the previous subsection, there are four hosts connected through three switches. The switches are forming a loop.

Run `topo2.py`. It creates the topology depicted in Figure 2.

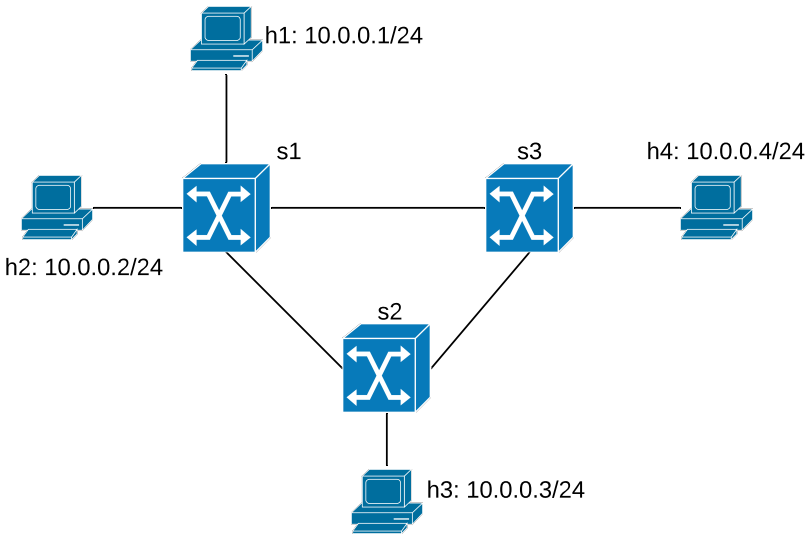

Figure 2: Network configuration with switches forming a loop

Now, perform a reachability test in Mininet using IPv4. A reachability test is a test to determine which hosts can 'reach' one another. This is performed by having each host ping all other hosts using its IPv4 address. A quick way to do this test in Mininet is by running the following command:

```
mininet> pingall
```

Q9/ What is the percentage of dropped packets? Why does it happen?

Solution. *100%. There are two reasons: 1) The broadcasting of packets does not work due to the loop in the network; and 2) Since the switches learn and update their forwarding table based on the response*

packets, the existence of a loop in the network creates inconsistency in the forwarding tables. For example, h2 wants to send a packet to h4 for the first time. The forwarding table on s1 does not have any information about this MAC address of h4 in its forwarding table. It sends a broadcast packet to the LAN, searching for the owner of this MAC address (h4). The packet to h4 traverses through s3 and also through s2 (and then s3). h4 responds to both messages and send them back from their own path and in return the forwarding tables of the switches are updated. The switch s1 receives both responses and has to update its forwarding table with two different value. This may cause inconsistency in the LAN.

Try to ping `h4` from host `h1` using their IPv6 address.

Q10/ What is the percentage of dropped packets when using IPv6 address? Why does it happen?

Solution. *100%. Similarly to IPv4 addressing, using IPv6 addresses has the same issues in the network with loop: the broadcast issue and the inconsistency in the forwarding tables.*

The standard solution to this problem is to enable the Spanning Tree Protocol (STP) at every switch.

Q11/ What does the spanning tree protocol achieve?

Solution. *It breaks the loops in the LAN by forcing the active topology to be a tree; here it disables one of the switch interfaces.*

The following command enables STP at the switch `s1`.

```
mininet > sh ovs-vsctl set bridge s1 stp-enable=true
```

Enable STP for all other switches in the network; then perform a reachability test again and verify the connectivity of all hosts.

Let's check how STP effects the network of Figure 2. First, we open a terminal from Mininet:

```
mininet > xterm s1
```

Then, open a Wireshark from the terminal. You should be able to see all the interfaces for every switch in the network (not only switch `s1`). You can see the volume of traffic beside each interface.

Execute the ping command for the following pairs of hosts.

- From `h1` to `h3`
- From `h3` to `h4`
- From `h2` to `h4`

Q12/ Write down the path of the packets for each pair of hosts.

Solution. *One link is disabled as a result of running STP. Here, the link between switches s1 and s2 is disabled (this might differ for each person).*

- *From h1 to h3 :* $h1 \rightarrow s1 \rightarrow s3 \rightarrow s2 \rightarrow h3$
- *From h3 to h2 :* $h3 \rightarrow s2 \rightarrow s3 \rightarrow h2$
- *From h2 to h4 :* $h2 \rightarrow s1 \rightarrow s3 \rightarrow h4$

Q13/ Are the hosts following the shortest path to send their packets to the destinations? Explain.

Solution. *No. The shortest path between `h1` and `h3` is: $h1 \rightarrow s1 \rightarrow s2 \rightarrow h3$ (might be different for each person; but at least one of the pairs should not follow the shortest path). However, the link between the switches `s1` and `s2` is disabled due to STP and `h1` to `h3` does not follow the shortest path.*

Shut down one of the active links between two switches, namely `si` and `sj`, using the following command Mininet:

```
mininet > link <si> <sj> down
```

Now take a break and come back in `5` minutes.

Use the Ping command to check the connectivity of the hosts.

Q14/ Write down the status (enabled or disabled) of each link between the switches. How did STP react when the link broke down? Explain.

Solution. *The link that was disabled by STP is now enabled. It takes some time for STP to detect a broken link; as soon as discovery of no connectivity in the network, STP updates the forwarding tables of the switches again.*

Q15/ Write down again the path of the packets for each pair of Q12.

Solution.

- *From `h1` to `h3` : $h1 \rightarrow s1 \rightarrow s3 \rightarrow h3$*
- *From `h3` to `h2` : $h3 \rightarrow s2 \rightarrow s3 \rightarrow s1 \rightarrow h2$*
- *From `h2` to `h4` : $h2 \rightarrow s1 \rightarrow s3 \rightarrow h4$*

Exit Mininet and clean up the topology before going to next subsection:

```
mininet> exit
# mn -c
```

2.3 Using a Router as a Networking Device

We have already configured a router in Lab 1, but we did not address how it worked. In this section we learn about the process of routing a packet. To do so, run the script `topo3.py`. It creates the network topology consists of four hosts and three routers as shown in Figure 3.

Q16/ Ping all hosts from each other using IPv4 and IPv6. Which hosts are unable to reach one another?

Solution. *Neither of hosts can ping each other.*

We will now attempt to fix the problem. First, open the `topo3.py` script and inspect it.

Q17/ What are the interfaces and respective IP addresses (v4 and v6) of the router `r1`?

Solution.
r1-eth0 → IPv4: `10.0.1.1`; IPv6: `2019::1:1`
r1-eth1 → IPv4: `10.0.2.1`; IPv6: `2019::2:1`

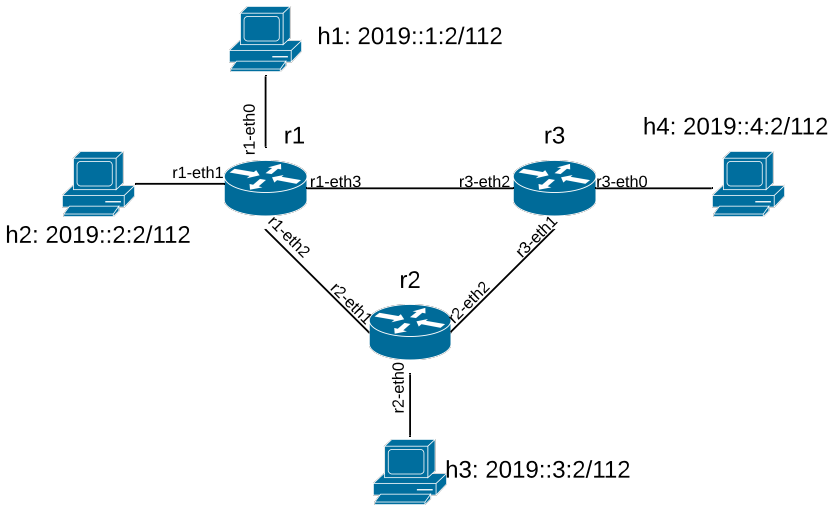

Figure 3: Network configuration with routers

r1-eth2 → IPv4: `10.0.5.1`; IPv6: `2019::5:1`
r1-eth3 → IPv4: `10.0.7.1`; IPv6: `2019::7:1`

Q18/ Can you spot any misconfiguration in the file?

Solution. *We noticed that ip forwarding is disabled at `r1`; the default gateway of `h2` is wrong: it is set to `10.5.0.100` for IPv4 and `2019::10:2` for IPv6, but these IP addresses do not exist in our network; and routing rules are not set in `r2`.*

Solve the issue at `r1`.

Q19/ What is the command you used at `r1`?

Solution. *We enable on `r1` IPv4 forwarding:*

```
# echo 1 > /proc/sys/net/ipv4/ip_forward
```

and IPv6 forwarding:

```
# echo 1 > /proc/sys/net/ipv6/conf/all/forwarding
```

Try again to ping hosts from each other with IPv4 and IPv6 addresses.

Q20/ Which hosts are still unable to ping each other?

Solution. *Only `h1` can ping `h4` and vice versa.*

Now solve the issue concerning `h2`.

Q21/ What are the commands you used in `h2` to achieve this?

Solution. *For IPv4:*

```
ip route add default via 10.0.2.1
```

For IPv6:

```
ip -6 route add default via 2019::2:1
```

Q22/ Try again to ping hosts from each other with IPv4 and IPv6 addresses. Is there any change in ping results?

Solution. *Hosts cannot ping `h3` and vice versa.*

You can check the routing table on router `r2` using the following command for IPv4 and IPv6, respectively:

```
ip route show
ip -6 route show
```

Q23/ Now, check the routing table of `r2` for IPv4 addresses. What is the problem with it?

Solution. *The routing table is not set properly on `r2`.*

Q24/ Solve the issue in `r2`. What commands do you use?

Solution.

```
ip route add 10.0.1.0/24 via 10.0.5.1
ip route add 10.0.2.0/24 via 10.0.5.1
ip route add 10.0.4.0/24 via 10.0.6.2
ip route add 10.0.7.0/24 via 10.0.6.2
```

Q25/ Check the routing table for IPv6 addresses. Is there any problem with it? Write down the commands to solve it.

Solution. *The routing table for IPv6 addresses is not set properly on `r2`.*

```
ip -6 route add 2019::1:0/112 via 2019::5:1
ip -6 route add 2019::2:0/112 via 2019::5:1
ip -6 route add 2019::4:0/112 via 2019::6:2
ip -6 route add 2019::7:0/112 via 2019::6:2
```

Ping again each host from another one using both IPv4 and IPv6 addresses, and confirm that your fix solves the problem.

Based on your observations, conclude this section by comparing switches and routers in a network.

Q26/ How do they differ in subnet mask and IP address assignment (IPv4 and IPv6)?

Solution. *In a network with switches only, all hosts receive link-local IPv6 addresses automatically therefore can communicate with each other without configuration. With IPv4, all hosts should be configured with the same IPv4 subnet mask and prefix (except if IPv4 auto-configuration is happening).*

In contrast, when using routers, each of its interface is connected to a separate IPv4/v6 network, i.e., subnet prefixes are different, and hosts can take whatever IPv4 and IPv6 addresses within the valid range. In this case, the hosts still receive link-local IPv6 addresses, but they cannot use them to communicate with another host in another subnet.

Q27/ How do they differ in creating routing/forwarding tables?

Solution. *The switches set the forwarding tables by learning. In a network without loops, no configuration is needed to connect the hosts. If the network has loops of switches, then by enabling STP in all of switches the forwarding tables are created . However, in a network of routers, the routing tables must be set manually (this can be done automatic by running protocols like OSPF (you will do it in the next labs).*

Q28/ How do they differ in performance/efficiency in a network with loops?

Solution. *When there is a loop in the network of switches, one (or several) link is disabled to break the loop. The side-effects of this action is increasing the load on the other active links in the network and paths are not the shortest. However, in the network of routers, all links are used and the traffic is balanced. Moreover, all paths can be the shortest, e.g., if you run OSPF.*

Now, exit Mininet and clean up the topology before going to next section:

```
mininet> exit
# mn -c
```

3 Connecting Virtual Environment to the Real World Using Network Address Translation (NAT)

In this section we will use what we learned from Lab1 about manipulating the `iptables` filter. The purpose of the section is to connect an isolated virtual network that we have deployed so far, to the real Internet.

Look at the Figure 4. The NAT in the box "Physical Machine" is the one created by VirtualBox. It connects the network interface of "LCA2 VM" to the physical interface of your laptop (**Note that in the network setting of the VM, there should be one Network Adapter which is set to "NAT"**).

As soon as you turn on the VM, remove the IP configuration of the interface connected to the NAT, as it is going to be used by Mininet. Get the list of interfaces in the VM and use the following command to flush the interface of the VM connected to NAT:

```
# sudo ip addr flush dev <interface name of VM connected to NAT>
```

Remember that the root password is `lca2`. Run the script `topo4.py`. This creates the network described in the box "Network in Mininet" shown in Figure 4. In this network, `h1` and `h2` are hosts, `r1` is also a host but configured to act as a perimeter router where we will have our connection to the real world. The goal of the switch `s3` is connecting `r1-eth1` to the network interface of the LCA2 VM. However, we know that LCA2 VM interface is used by the virtual machine itself. Therefore, we add a port to `s3` and connect it the network interface of LCA2 VM.

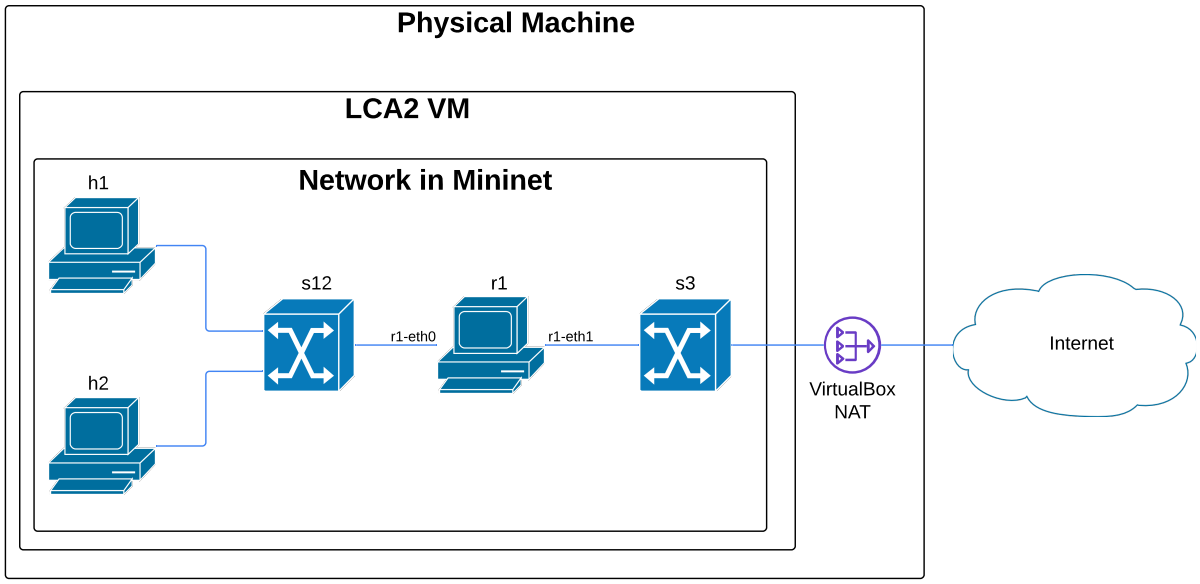

Figure 4: Network configuration with a connection to the real world

3.1 Connecting the perimeter router to the Internet

The goal of this section is to connect `r1` to the Internet. To perform the bridging between physical and virtual network-adapters, execute the following command from Mininet terminal to connect the interface of VM to the switch `s3`.

```
mininet> sh ovs-vsctl add-port s3 <interface name of VM connected to
NAT>
```

Replace `<interfacename>` with the interface that accesses the Internet in your VM.

The next step is to assign an appropriate IP address to the `r1-eth1` interface of `r1`. The address should be in the same subnet as `<interface name of VM connected to NAT>` because both are connected by a switch and are thus in the same subnet. In a physical network, address allocation can be done manually or using a DHCP server. The same holds in the VM, as VirtualBox also provides a DHCP server. We use DHCP in order to avoid conflict with IP addresses that the VM might have allocated to other interfaces in the same subnet as the interface of the VM connected to the VirtualBox NAT. You can ask the DHCP server of VirtualBox to provide a valid IP address by using the following command in the terminal of `r1`:

```
# dhclient r1-eth1
```

This automatically sets a usable IPv4 address to the `eth1` interface of `r1`, allowing it to access the internet through the bridge we just set up. Test the configuration by pinging `8.8.8.8`.

Q29/ What is the IPv4 address allocated by the DHCP server of VirtualBox to `r1-eth1`? Is it a private address or public one? What is the IPv4 address of the DHCP server? Hint: You can use Wireshark and to capture the packets when asking for an IP address from DHCP.

Solution. *The IPv4 address is 10.0.2.16. The IPv4 address is a private one (the network behind the NAT of VirtualBox). The IPv4 address of the DHCP server of VirtualBox is 10.0.2.2. When running the `dhclient` command, you can run Wireshark and see the destination address of the packet sent to DHCP server.*

3.2 Providing Internet access to Mininet hosts

The goal of this subsection is to provide Internet access to hosts `h1` and `h2`, via `r1`.

Q30/ The perimeter router `r1` has one interface that is connected to the Internet. Imagine that we setup `r1` as a router and use it to connect `h1` and `h2` to the Internet. Which problems does this create ?

Solution. *We need to allocate addresses to `h1` and `h2`. Whatever addresses we end up allocating, we need that the network prefix used by `h1` and `h2` is visible to the rest of the virtual network, we should make VirtualBox aware of it. This may be hard to do and we prefer to use a NAT inside `r1`.*

In order to give Internet access to `h1` and `h2` we will configure `r1` as a NAT. Indeed, the situation is the same as if `r1` would be connected to an ADSL modem at home: `r1` receives a single IP address from its provider (here: VirtualBox) and we want to use it to connect more devices (here: `h1` and `h2`).

Q31/ Propose the `iptables -t nat` command you need to properly configure `r1` to this end.

Solution.

```
# iptables -t nat -A POSTROUTING -o r1-eth1 -j MASQUERADE
```

Test from `h1` and `h2` that you have Internet connectivity by pinging `8.8.8.8`. Next, let's explore in detail the result of our configuration.

Do `traceroute` to Google from `h2` and then from `r1`, while capturing `eth0` and `eth1` traffic on `r1` using Wireshark. Explore the difference in the traffic on both cases.

Q32/ When doing `traceroute` from `h2`, what is the difference in the packets captured on `r1`'s `eth0` and `eth1`?

Solution. *The source IP address is modified according to the NAT rule, replacing the IP address of `h2` with the IP address configured on `eth1` in `r1`.*

Q33/ Focus on the `traceroute` from `r1`. What is the difference in the packets as compared to `h2`?

Solution. *Source and destination ports are different. Source and destination IP addresses are the same.*

Q34/ Which field in the UDP packet is used to identify the (local) source IP address of `h2` in order to properly forward incoming ICMP replies back to it?

Solution. *This is done via UDP port, and NAT device keeps a track on the private source IP/port and the correspondent public IP/port.*

Do `ping` to Google from `h1` and `r1`, while capturing the traffic on `r1` (both on `eth0` and `eth1`) using Wireshark. Explore the difference in the traffic in both cases.

Q35/ What is the difference in the request ICMP packets captured between packets sent from `h1` and packets sent from `r1` when capturing on the exit interface of each?

Solution. *The ICMP query ID is different.*

Q36/ Conclude how the incoming ICMP replies are forwarded back to `h1` when doing `ping` from `h1`. In particular, which field in the request/reply ICMP packets was used to identify the (local) source IP address?

Solution. *ICMP query replies are forwarded back to `h1` based on the QueryID taken from the ICMP packet header. If we issue the `iptables -t nat -L -v -n` command, we will see this IP address to Query ID mapping. This is handled according to* ***RFC 5508***

4 POINT-TO-POINT WIRED CONNECTION OF TWO PHYSICAL MACHINES

In this section, you will connect two physical machines via an Ethernet cable. The goal of this section is to give you a feel about the communication between physical machines.

To accomplish this section, you are required to have access over two physical machines, e.g. your laptop and your friend's, and an Ethernet cable. If you need a machine or an Ethernet cable, you may use the ones in the Internet Engineering Workshop (IEW)/INF019.

Figure 5: Point-to-point wired connection of two physical machines

4.1 SETTING UP THE CONNECTION

The goal of this subsection is to make a point-to-point connection between two physical machines, namely `M1` and `M2`, via cable. To avoid any complication in the process, please turn off the Wi-Fi connection of the machines (or set it to flight mode). Now, physically connect the two machines by plugging in one port of an Ethernet cable to `M1` and the other port to `M2`.

Q37/ What are the interfaces in `M1` and `M2` that are connected through the Ethernet cable? What are their corresponding IPv4 and IPv6 addresses? What types of addresses are these ?

Solution. *Get the list of interfaces by running `ifconfig` in MacOS and Linux, and `ipconfig` in Windows. Open Wireshark and start pinging from `M1` and `M2` the IP address `8.8.8.8` separately. You can see which interface is used for pinging.*

To find out their IP addresses, use the commands seen in the previous labs (e.g. with Linux and MacOS, `ip addr show`; with Windows, use `ipconfig`).

The IPv6 addresses are link-local addresses (auto-configured). For IPV4, it depends on the system. Some systems auto-configure link-local addresses `169.254.x.x`, some others configure private addresses, yet some others don't configure any address.

If you are using Windows, turn off the Windows Firewall now, as it may block traffic between interfaces, for security reasons.

Q38/ Ping `M1` from `M2` and vice versa, using IPv6 address. Does it work ?

Solution. *Yes. They are able to ping each other using IPv6 addresses. Since two computers are in the same LAN and link-local IPv6 addresses are autoconfigured, they can ping each other.*

Q39/ Ping `M1` from `M2` and vice versa using IPv4. Do you receive any ping-reply? Explain.

Solution. *It depends on the system. If link-local IPv4 addresses were auto- configured, this should work. Otherwise, if private addresses are configured they should not be able to receive ping-reply because they are not in the same subnet.*

We now set up a private IPv4 network between `M1` and `M2`, using private but routable addresses.

Q40/ Write down how you do this.

Solution. *The link local addresses are not routable, so we cannot use them if they were auto-allocated. With Linux and MacOS, do for example:* `10.2.2.0/24`.

at `M1`*:*

In Linux:

```
ip addr flush dev <M1 interface>
ip addr add 10.2.2.2/24 dev <M1 interface>
```

In Mac, use System Preference/Network to set up IP address manually.

In Windows, you need to go to Settings and change the IPv4 properties (address and network mask) of the corresponding interfaces.

at `M2`*:*

In Linux:

```
ip addr flush dev <M2 interface>
ip addr add 10.2.2.3/24 dev <M2 interface>
```

In Mac, use System Preference/Network to set up IP address manually.

With Windows, you need to go to Settings and change the IPv4 properties (address and network mask) of the corresponding interfaces.

Verify the connection of the machines by executing the ping command again with IPv4 on `M1` and `M2`.

4.2 Measuring the Bandwidth of the Communication Link

So far, you have built a point-2-point IPv4/v6 network between two physical machines via an Ethernet cable. The goal of this subsection is to find out the practical bandwidth of the Ethernet cable.

In Lab1, you worked with `iperf` and how to measure the physical bandwidth of a communication link. In this subsection, you want to measure the bandwidth of the Ethernet cable connecting `M1` and `M2`.

Run `iperf` as server in `M2`.

Q41/ What are the commands you used in `M2` to run it as an `iperf` server? What are the IP address and port of the server?

Solution.

```
iperf -s
```

The IP address is the one used by the interface of `M2` *and the port is written in the terminal when the server runs.*

Now run the `iperf` client in `M1`.

Q42/ What are the commands you used in `M1` to run it as an `iperf` client?

Solution.

```
iperf -c <IP of the server> -p <port number of the server>
```

Q43/ What is the practical bandwidth of the Ethernet cable?

Solution. *It should be higher than* 80*Mbps (for* 100*Mbps link) or* 950*Mbps (for* 1*Gbps link).*

4.3 SHARING INTERNET ACCESS

The goal of this subsection is to allow `M1` to access the Internet via `M2`. This is similar to "tethering", when you share a mobile phone's internet access with other devices that do not have Internet access. Assume the

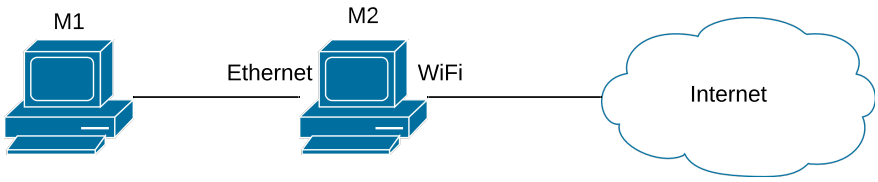

Figure 6: Sharing Internet with a friend!

configuration is as in Figure 6. We want to connect `M1` via `M2` to the Internet. We could setup `M2` as a bridge, a router, or a NAT.

Q44/ Discuss the pros and cons of each of these configurations.

Solution.

- *Bridge: this requires the IP address of M1 to be a public address and be in the same subnet as the WiFi IP address of M2. This requires that the internet service provider gives us several IP addresses, not just one. This may be sometimes possible with IPv6 but usually not with IPv4.*
- *Router: this requires that we setup a subnet prefix for the Ethernet between M1 and M2 and that this prefix be a set of public addresses in the same subnet as the WiFi IP address of M2. Again,This may be sometimes possible with IPv6 but usually not with IPv4.*
- *NAT: this does not place any constraint on the addresses allocated to the Ethernet segment and requires only one public address for M2. This works with IPv4 and IPv6.*

Turn on the Wi-Fi interface of the physical machine `M2` and check its connectivity by pinging `google.com`. Note that `M1` still does not have Internet access.

4.3.1 SETTING UP M2 AS A NAT

If `M2` is running Linux, then it can be setup as a NAT. If this is your case, continue with this section; otherwise jump to Section 4.3.2.

Q45/ Describe how you setup M2 as a NAT in Linux.

Solution.

- *Install the iptables package if it is not present; then do*

```
sudo echo 1 > /proc/sys/net/ipv4/ip_forward
sudo iptables -t nat -A POSTROUTING -o <VM Wi-Fi interface> -j
MASQUERADE
```

Now you may go directly to Section 4.3.3.

4.3.2 Setting Up the LCA2 VM in M2 as a NAT

If `M2` cannot be natively configured as a NAT, we can use a VM inside `M2` and configure it as a NAT, since we know how to do this. This involves two steps:

1. Connecting the VM to the Ethernet port
2. Setting up the VM as a NAT

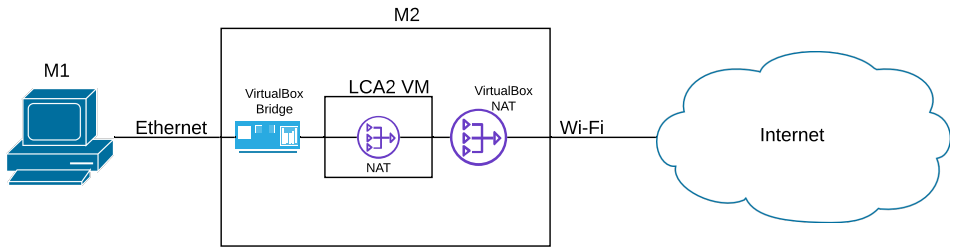

Figure 7: Sharing Internet with NAT in the VM

Connecting the VM to the Ethernet port As usual, the VM is connected to the Internet (i.e to the WiFi interface) via a NAT. We could do the same to connect the VM to the Ethernet adapter, but since we have full control of all IP addresses allocated in this Ethernet LAN, we can do a simpler solution, i.e., use a bridge.

Set two network adapters (a NAT and a Bridge) in host `M2`. To do so, you have to open "Settings" of the VM, then select "Network", go to tab "Adapter 1" and set it to "NAT" and tab "Adapter 2" and set it to "Bridged adapter"; the name should be the Ethernet interface of your physical machine as it is used to connect `M2` to `M1`.

Now power on the LCA2 vitual machine. Open a terminal in the VM and ping `epfl.ch`.

Q46/ The name of interfaces are changed inside the VM. Find out the names of the interface connected to the VirtualBox NAT and connected to the VirtualBox bridge.

Solution. *Open Wireshark. The interface that has traffic on is connected to NAT.*

Use the following command on the VM in `M2` separately to find out their IP addresses:

```
ip addr show
```

It shows three adapters: LoopBack (LO), one connected to NAT, and the other is connected to Bridge.

We need to check the connectivity of the VM in M2 to Internet via NAT (we already had it in Section 3) and set up its connection to M1 via bridge.

The first step to achieve this goal is to assign a suitable IPv4 address to the bridge interface of the VM.

Q47/ What is a suitable IPv4 address for the bridge interface in the VM at M2? Explain why you selected this IPv4 address.

Solution. *The IP address should be in the same subnet as M1 as they are in the same LAN.*

Q48/ What are the commands you need to execute in the VM to assign the IPv4 address? ***Solution.***

```
ip addr flush dev <bridge interface in VM of M2>
ip addr add 10.2.2.4/24 dev <bridge interface in VM of M2>
```

Now, ping M1 from the VM and verify the connectivity of VM and M1 through the bridge.

SETTING UP LCA2-VM AS A NAT So far, we have connected one interface of the VM in M2 to the Internet via the VirtualBox NAT and WiFi connection, and the other interface of VM to M1 via the VirtualBox bridge and Ethernet cable. The final step is to set up the LCA2-VM to work as a NAT.

Q49/ What are the commands you need to execute at the VM of M2 to enable NAT?

Solution.

```
sudo echo 1 > /proc/sys/net/ipv4/ip_forward
sudo iptables -t nat -A POSTROUTING -o <VM Wi-Fi interface> -j
MASQUERADE
```

4.3.3 FINAL STEPS!

Q50/ What are the commands you need to execute at M1 to route the traffic to M2?

Solution.

```
ip route add default via <IP address the bridge in the VM of M2>
```

Verify the Internet connection of M1 by pinging 8.8.8.8 as the Google DNS server.

Assume a router r is the next hop of M2. The host M1 starts to ping 8.8.8.8.

Q51/ If Wireshark is running on r, what would be the source IPv4 address of the ICMP packets send by M1? Explain the reason.

Solution. *It is the IP address of the Wi-Fi interface of M2. As on M1, NAT is enabled, all the incoming traffic from M2 is passed though the NAT in M2. According NAT, all the machines behind it are transparent from the Internet and are seen M2 as the sender/receiver of traffic.*

One application of such configuration is to share your own Internet access with other people who are not connected to the Internet. Suppose a friend of you visits Switzerland and would like to have Internet access; however, the cost of Roaming is too much for him/her. Therefore, you would like to do him/her a favor and share your own Internet with him/her. This can be done by the practical experience you have obtained in this section.

Disclaimer: You may think of sharing your EPFL Internet connection using your GASPAR credentials and give Internet access to your friend. We would like to warn you that this generous behavior is unfortunately forbidden.

Note: If you are using a Windows machine, turn on the Windows Firewall again.