Lab 1: IPv6

Reti di Calcolatori II – Università di Trieste – Martino Trevisan

In this lab, you will practice with IPv6, assigning configuration parameters to a set of virtual hosts. You will use a software called Mininet, which makes it playing with virtual networks rather easy.

First, visit the Mininet website (http://mininet.org/) and check how it works. Then, install it. If you use Ubuntu, just run the following command: `sudo apt-get install mininet`

Mininet works by creating virtual hosts using Linux containers. Containers are connected one with each other using a virtual links and virtual switches. It is possible to customize the network topology to create arbitrary complex networks.

Once the network is created, you can run any command on the virtual hosts, such as ping, curl, etc. Hosts are automatically configured with private IPv4 addresses, but you can modify their configuration with the normal Linux tools.

Start Mininet

Mininet can emulate various network topologies. Start a simple topology with four hosts connected with a single switch with:

```
sudo mn --topo single,4
```

Then, you are prompted to an interactive shell, where you can execute commands on any host. When you type a command on the shell, the first word must indicate the virtual host where you want to execute the command. The remaining part constitutes the bash command.

For example, to check the IP address on `h1`, you should run the `ip addr` Bash command on host `h1`, with:

```
h1 ip addr
```

To ping `h2` from `h3`, type:

```
h3 ping h2
```

Assigning IPv6 addresses

Check, using the proper commands, if the host already have any IPv6 address assigned. Can you already ping a host using IPv6 from any other host? You can do this, by specifying an IPv6 address as argument of the `ping` command, such as:

```
h1 ping fd00::4
```

Now, assign to the four hosts an IPv6 address from the `fd00::/8` subnet. To add an IPv6 address to an interface, use the command:

```
ip -6 addr add <ip>/<class> dev <interface>
```

Then, ping a host using IPv6. Does it work?

Check the generated traffic and compare it with the traffic you get when running a ping on IPv4. Does it vary? Which protocols are used to obtain the MAC address of the destination host given its IP address?

Hint 1: when you start Mininet, various virtual interfaces are automatically created. Run Wireshark on those interfaces to intercept the traffic between virtual hosts.

Hint 2: To clean the ARP cache, run the command:

```
ip -s -s neigh flush all
```

The error ping6: 10.0.0.2: Address family for hostname not supported indicates that you are trying to use an IPv4 address (10.0.0.2) with the ping6 command, which is designed specifically for IPv6. This happens because, although your hosts have link-local IPv6 addresses, they do not yet have globally routable IPv6 addresses assigned. As a result, the command cannot properly resolve the parameters.

After assigning IPv6 addresses manually to the hosts (e.g., using h1 ip -6 addr add fd00::1/64 dev h1-eth0 and h2 ip -6 addr add fd00::2/64 dev h2-eth0), I observed that the ping6 command worked as expected. Running h1 ip -6 addr confirmed the assignment, showing that the interface address had changed from inet6 fe80::dcf3:26ff:fee4:355c/64 (link-local) to inet6 fd00::1/64 scope global, indicating the IPv6 address was correctly assigned.

When using ping for IPv4 and ping6 for IPv6, I noticed some differences:

1. Header Size: IPv6 uses larger headers compared to IPv4. IPv6 headers are 40 bytes, whereas IPv4 headers are only 20 bytes.
2. Address Resolution Protocol: IPv4 uses the ARP (Address Resolution Protocol) to resolve MAC addresses. In IPv6, this functionality is replaced by NDP (Neighbor Discovery Protocol), which operates via ICMPv6.

Duplicate IP addresses

Assign the same IPv6 address to two hosts. What happens when you ping the duplicate IP address from another host? How many Neighbour Solicitations the host receive?

Try the same with IPv4 (use the `ip` command to have duplicate addresses). How many ARP replies the host receive?

Hint: check the traffic a host generates when you assign a new IPv6 address to it. Check deeply the `ip addr` command output.

In IPv4, there is no equivalent to IPv6's DAD (Duplicated Address Detection) process. When two nodes share the same IPv4 address, both will respond to the same ARP (Address Resolution Protocol) request, causing an ARP conflict.

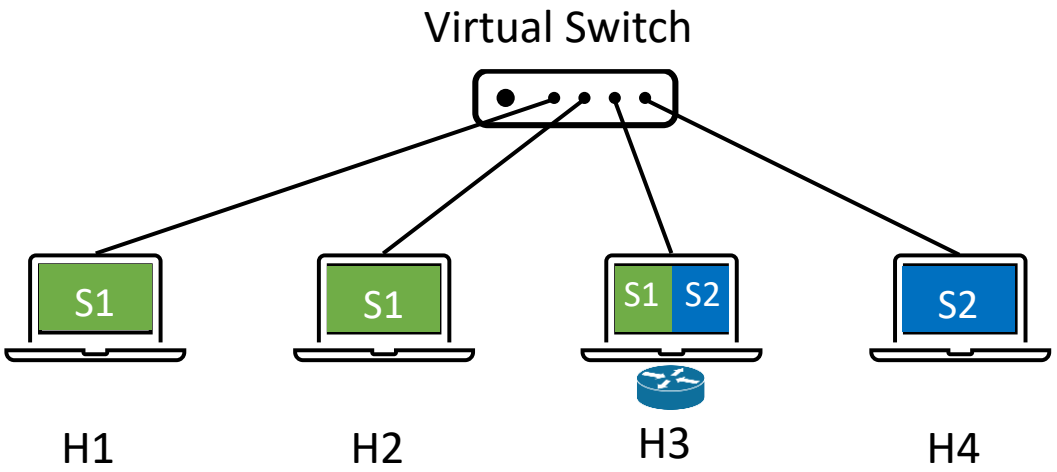
Using tools like Wireshark, you'll notice that a single ARP request results in two responses, which creates confusion for the sender, as it cannot determine which device to communicate with.

The symptoms of an IPv4 address conflict include unreliable ping results and potential error messages, such as **"**IP address conflict detected**"**. Unlike IPv6, where a dadfailed status indicates a duplicated global address, IPv4 doesn't have an automatic mechanism for detecting or preventing these conflicts.

However, the devices may still have link-local IPv6 addresses (which are automatically assigned and not affected by DAD). These link-local addresses allow the nodes to communicate within the local network segment, even though their global IPv6 addresses might conflict.

Play with IPv4 Routing (optional)

Now divide the four hosts into two subnets S1 and S2. Two hosts belong to the first subnet S1, one host belongs to the second subnet S2. The last host belongs to two subnets, thus it has two IP addresses and acts as a router, like in the following picture.



First, decide the subnet addresses and assign them with the commands:

```
ip addr add <ip>/<class> dev <interface>
```

Notice that the above command adds an IP address to an interface. A single interface can have more than one IP address. If you need to remove and address from an interface, use the command:

```
ip addr del <ip>/<class> dev <interface>
```

You should also modify the routing table of the devices using the command:

```
h1 ip route add <subnet> via <next hop> dev <interface>
```

Think carefully how routing tables of different hosts should look like. Notice that the nodes that acts as a router shall have two IP addresses.

Moreover, to make a host to act as a router you should enable IP forwarding with the command:

```
sysctl -w net.ipv4.ip_forward=1
```

You should also prevent the router host from sending erroneous ICMP redirect with the command:

```
h3 echo 0 | tee /proc/sys/net/ipv4/conf/*/send_redirects
```

Once everything is configured, run a ping between two hosts on different subnets. You must report in the answer a schematic view of your network, including routing tables and IP address. Moreover, write all frames generated when you launch a ping, detailing source and destination MAC and IP addresses and the meaning of each packet.

Enlarge the box as needed

Installation notes

There are various possibilities to run Mininet

Native Ubuntu: The best way to use Mininet is to install it on an Ubuntu Machine. Just run : `sudo apt-get install mininet`

Using the VM: You can download the VM image available on the official website (https://mininet.org/download/) and run it with Virtual Box. However, the VM has IPv6 disabled by default, and you must enable it with the following steps:

- Edit the `/etc/sysctl.conf`

  Delete the lines:

  ```
  net.ipv6.conf.default.disable_ipv6 = 1
  net.ipv6.conf.all.disable_ipv6 = 1
  net.ipv6.conf.lo.disable_ipv6 = 1
  ```

- Edit `/etc/default/grub` and delete the `ipv6.disable=1` string in the `GRUB_CMDLINE_LINUX` and `GRUB_CMDLINE_LINUX_DEFAULT` lines.
- Run: `sudo update-grub`
- Reboot the machine

Using Windows subsystem for Linux: install `mininet` and `bridge-utils` with: `sudo apt-get install mininet bridge-utils` . Then, you must add the `--switch lxbr` option when running Mininet, like for example:

```
sudo mn --topo single,4 --switch lxbr
```