Lab 6: Designing and implementing a small network

50.012 Networks

Hand-out: November 1
eDimension hand-in: November 8, 11pm

1 Introduction

- Use knowledge from the class to design and implement a small network
- Once again, we are using mininet to run a set of virtualized hosts
- Your job is to connect these hosts correctly through switches and routers, and to configure critical services

2 Setup

- This exercise again assumes that you have a running miniNet installation
- Download the lab6.zip from eDimension and unpack to some local folder, e.g. ~/lab6/
- Change into the lab6 folder, and open up another terminal tab with CTRL+SHIFT+T
- Execute the `install.sh` script

```
sudo bash ./install.sh
```

3 Introduction

The general setup in this lab is shown in Figure 1. 1 gateway, 2 local servers and 5 local hosts are connected to one switch. The gateway is also connected to the *Internet*, in our case another router.

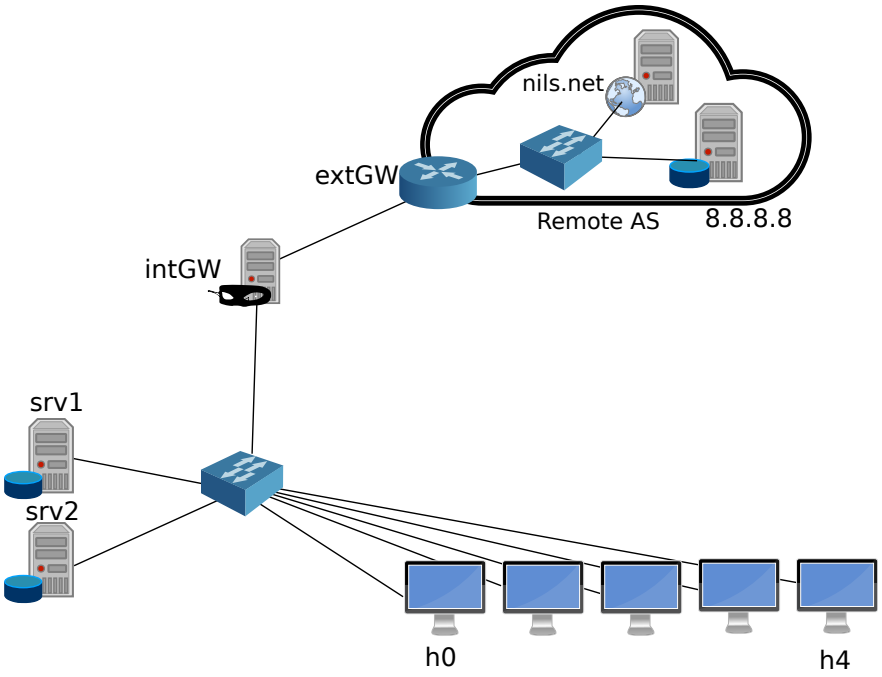

Figure 1: Basic topology in Mininet

3.1 Warming up

- Start up mininet

```
sudo python ./net.py
```

- Open an xterm on `h1` and check the local network configuration

```
mininet> xterm h1
# ifconfig
# route -n
```

- What is the local network that was chosen for the hosts?
- Are the two servers `srv1` and `srv2` in the same subnet?
- Can you observe the switch in `tracepath` from `h1` to `srv1`? Why not?
- What is the gateway for all devices?
- Can you ping/reach the server nils.net (8.8.8.2) from `h1`? If not, do you have an idea what is going wrong?
- Is a DHCP server running in the local network? On which machine? You can use `dhclient <IFNAME>` to request a new IP address manually:

```
mininet> xterm h1
# dhclient h1-eth0
```

- Observe the DHCP traffic in a suitable wireshark session if necessary

4 Configuration of the System

4.1 Changing the DHCP configuration

- Open the DHCP server configuration file at `srv1DHCP.conf` using your favorite editor.
- Look at the settings and try to understand what they mean. Do you find something that might need to be improved? Do that change, save, and restart mininet using the `net.py`
- In the open mininet session, open an xterm on h1 again and ping 8.8.8.8. Can you reach it now?

4.2 DNS

- Lets now try to configure `h1` to use our custom DNS server
- Note: this can be a bit tricky. For best results, start the mininet session, and then
  - `sudo service network-manager stop` on your host machine (you will lose Internet)
  - `sudo nano /etc/resolv.conf` and replace the `127.0.0.1` IP with `8.8.8.8`
  - Now it should work until you restart network manager (with `sudo service network-manager start`) or you restart mininet
- On host `h1`, ping nils.net. Can you reach it? Why? Try using `dig` or `nslookup` to find out more. What is the IP of nils.net?

4.3 Observing NAT in action

- In the provided setup, one node provides NAT for the hosts with private IP address. Which node is this?
- Use wireshark on that host to inspect incoming and outgoing connections
- Have a look at the net.py script to see what is going on in the enableNAT() function. This is all it needs to configure a host to do NAT (under Linux).

4.4 Simple Firewalling

- The NAT was actually set up using `iptables`, which can be used as Linux firewall application
- In a nutshell, a firewall can prevent or allow incoming connections to a machine
  - In particular, specific rules can be added to **drop** (block) traffic from certain sources, or using certain protocols or ports
- Open an xterm on `intGW` and add a rule to block traffic from srv2 specifically. Test if it works, i.e. if you can still ping 8.8.8.8 from srv2 after the rule is effective. Ideally, you should not!
- Use either the Internet, the manpage, or the net.py script as reference. In net.py, a similar rule is used to emulate unroutable private IP-addresses.

  - A good example tutorial is found at
    `http://www.howtogeek.com/177621/the-beginners-guide-to-iptables-the-linux-firewall/`

- *Hint*: there are different arguments like `-I` and `-A`. They change the order in which rules are evaluated. `-I` puts the rule you insert to the front.
- *Hint2*: there are different *chains* like `INPUT` and `FORWARD`. The input chain applies to all packets direct to the host, while the forwarding chain applies to all packets forwarded by the host (e.g. on a router or NAT host).
- *Hint3*: there are different *commands* like `DROP`, `REJECT` and `ACCEPT`. The main difference between `DROP` and `REJECT` is that in the latter case, an error message is sent to the source.

4.5 (Optional)

- If you want, feel free to change/ extend the net.py script. Possible ideas:
  - Increase the number of desktops
  - Add a second switch and connect it
  - Introduce another router or firewall to separate a server subnetwork from the desktop subnetwork

5 What to Hand in

5.1 eDimension submission:

Please provide a writeup (in PDF format with your name) that includes the following information:

- A brief description of the networking setup provided
  - IP subnet for the desktops and server
  - Which server is acting as DHCP server
  - What you had to fix in the DHCP setup to reach 8.8.8.8
- Why can you resolv nils.net? Briefly describe where from `h1` knows about a DNS server, and what its IP is.
- Who is doing the NAT'ing? Which address ranges it is translating between?
- Did you manage to block srv1 from reaching the outside world?
- Did you do any of the optional tasks?

5.2 Checkoff:

- No checkoff required if you submitted your reply sheet