Networking exam

About this subject

Welcome to the networking (`TEAR` or `CLAN` depending on which class you're in) exam. In this exam, you will have to create three different topologies (that is, three separate GNS3 projects), each evaluating a particular concept studied in class. As each project is completely independent, you may of course do them in the order you like the most.

Warning: this exam is **hard**, you will probably not be able to do everything in only an hour and a half. That's okay, you don't have to do everything to get a great grade, just do your best.

Configuring GNS3

Just like the lab, you'll have to install and configure GNS3 at the start of the exam, but this shouldn't take you more than a few minutes:

`/afs/cri.epita.net/resources/teach/networking/GNS3/install_gns3.sh`

Don't forget to look at the lab document, still available here:

`/afs/cri.epita.net/resources/teach/networking/GNS3/lab.pdf`.

We will only use the following appliances:

- `Open vSwitch` (*not* the "Open vSwitch management" appliance!) for our switches.
- A custom host (aka "guest" or "end device") named `Host`, based on the `chewiebeardy/gns3-host:latest` docker image and with 1 network adapter.
- A custom router named `Router`, based on the `chewiebeardy/gns3-host:latest` docker image and with 4 network adapters. In the example images I included later, I changed the icon of the appliance template for clarity, but it's not mandatory to do so.

Persisting your work

Of course, I can't evaluate your work if it's wiped when you shut down your appliances! Just like the lab session, everything you do has to be stored in some configuration file.

The `Open vSwitch` appliance will by default persist all `ovs-vsctl` configuration you make.

The following files/folders are persisted on the appliances based on `chewiebeardy/gns3-host:latest`:

- The `/etc/network/interfaces` file, where you can put your network configuration. Pro tip: instead of editing it inside the container, you can also right click the node and select `Edit config` to edit this file directly.
- The `etc/iptables/rules` file, containing `iptables` rules that you might want to persist. You shouldn't create/edit this file by hand, but instead use the `/usr/sbin/iptables-save` and `/usr/sbin/iptables-restore` commands to manipulate it.

For example, the following `/etc/network/interfaces` configuration will setup two interfaces, add a custom route, and then apply the `iptables` rules:

```
auto eth0
iface eth0 inet static
        address X.X.X.X
        netmask X.X.X.X
        gateway X.X.X.X

auto eth1
iface eth1 inet static
        address X.X.X.X
        netmask X.X.X.X
        gateway X.X.X.X

up ip route add [...]
up /usr/sbin/iptables-restore < /etc/iptables/rules
```

Submitting your work

Each project will have to be exported with the `File -> Export portable project` menu option, excluding base images. The name is of course to be respected at all costs, because the correction is automatic! This will give you a `.gns3project` file.

Assemble those projects in a tarball named `login_x-networking.tar.gz` (where `login_x` is your login) with the following contents:

```
login_x-networking.tar.gz:
  login_x-networking/
    cantfindmywayhome.gns3project
    route66.gns3project
    stickyfingers.gns3project
```

Send this tarball to me via email at kevin.sztern@epita.fr with the following subject:

```
[NETWORKING][EXAM] login_x - submission
```

Where `login_x` is your login. Be careful to use this **exact** subject!

Be careful: Just like the project name, each machine, be it a host, a switch or a router, has to be named **exactly** like asked, otherwise I won't be able to find it! Do **not** keep the default names like "Host-1" and rename them explicitly by right clicking on them and selecting `Change hostname`.

Topology 1: Can't Find My Way Home

Instructions

The exported project has to be named `cantfindmywayhome.gns3project`.

Goal

Your goal is to create the following topology:

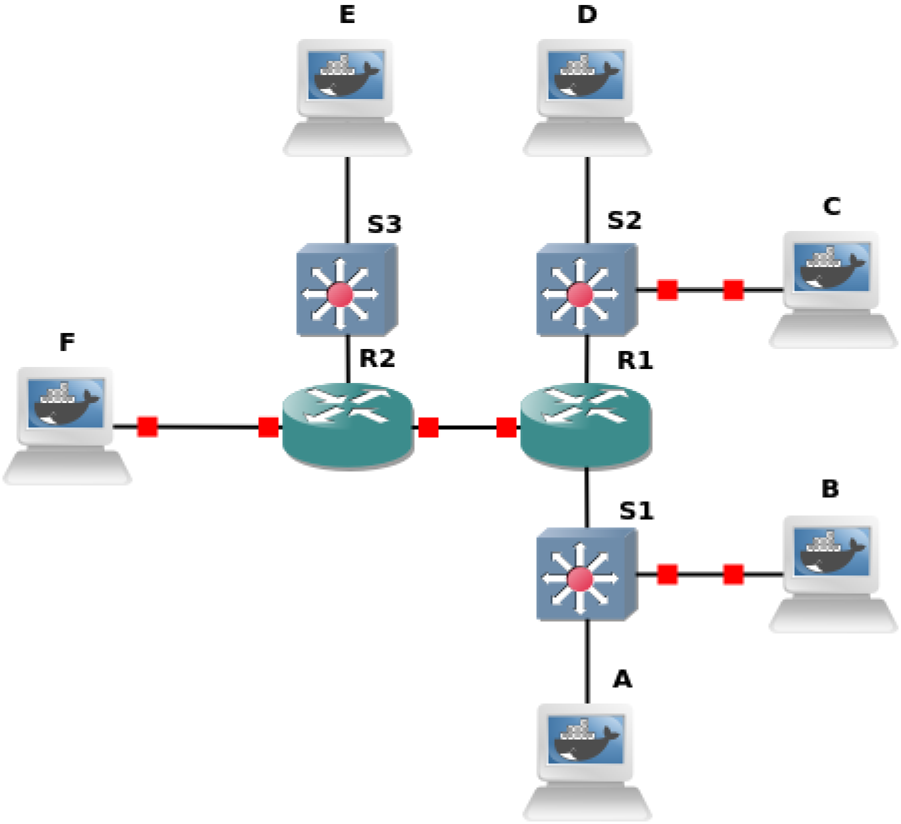

Figure 1:

In total, there are five local networks. Yes, some of those networks do not have a switch, they're just point-to-point networks.

The following configuration has to be applied:

Name	IP Addresses
A	• 192.168.1.2/24 (eth0)
B	• 192.168.1.3/24 (eth0)

Name	IP Addresses
C	• 192.168.2.2/24 (eth0)
D	• 192.168.2.3/24 (eth0)
E	• 10.25.58.2/24 (eth0)
F	• 10.25.87.2/24 (eth0)
R1	• 192.168.1.1/24 (eth0) • 192.168.2.1/24 (eth1) • 172.16.74.1/24 (eth2)
R2	• 10.25.58.1/24 (eth0) • 10.25.87.1/24 (eth1) • 172.16.74.2/24 (eth2)

I'll leave it to you to deduce on which adapter each cable should be plugged.

The subtlety of this exercise is that default routes are not enough: you *must* configure additional routes on your routers for everything to be connected! No need to get fancy with prefix aggregation, I suggest you hardcode the exact prefixes of every distant network on your routers.

How to test

Each host should be able to ping every other host.

Why this name?

"Can't Find My Way Home" is a song by Blind Faith, a blues rock supergroup featuring Steve Winwood, Eric Clapton, Ginger Baker and Ric Grech.

Topology 2: Route 66

Instructions

The exported project has to be named `route66.gns3project`.

Goal

Let's imagine the following scenario: you have a home network (think of your box at home as a combo switch+router), but the distant router you're connected to is RFC 1918 compliant: it will drop any incoming packet with source addresses having the prefixes `192.168/24`, `10/8` or `172.16/12`.

On the other side of this router is the machine we want to contact. You'll have to use NAT masquerading on your router to pretend that forwarded packets from your home network have the router's public IP as a source address.

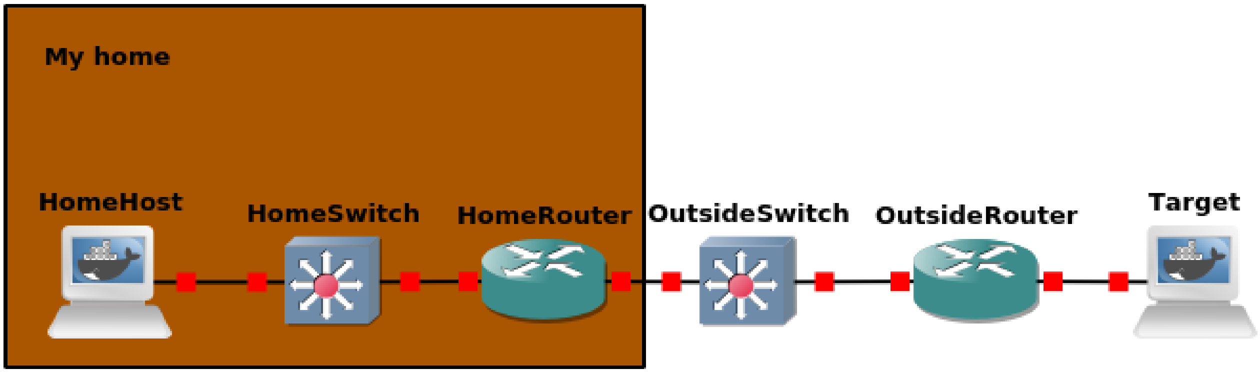

Figure 2:

The following configuration has to be applied:

Name	IP Addresses
HomeHost	• 192.168.42.2/24 (eth0)
HomeRouter	• 192.168.42.1/24 (eth0) • 82.232.189.74/22 (eth1)
OutsideRouter	• 82.232.188.1/22 (eth0) • 195.154.128.1/17 (eth1)
Target	• 195.154.165.95/17 (eth0)

In addition, some `iptables` magic must be done:

- On the `OutsideRouter`, add a rule to drop any incoming packet with source address beginning with `192.168.0.0/16` on the `FORWARD` chain.

- On the `HomeRouter`, add a rule to masquerade outgoing packets from the home network to the outside.

How to test

`HomeHost` should be able to ping `Target`. Using `wireshark` on the outside should show the source IP to be the `HomeRouter`'s public IP.

Of course, since this is just masquerading, `Target` cannot initiate a connection to `HomeHost`. This will obviously not be tested.

Why this name?

You might wonder why I chose the name “Route 66” for this exercise instead of the first. It’s simple: “Route 66” is a song by Nat King Cole. Get it?

Topology 3: Sticky Fingers

Instructions

The exported project has to be named `stickyfingers.gns3project`.

Goal

You guessed it, you prepared for it: here comes the router on a stick!

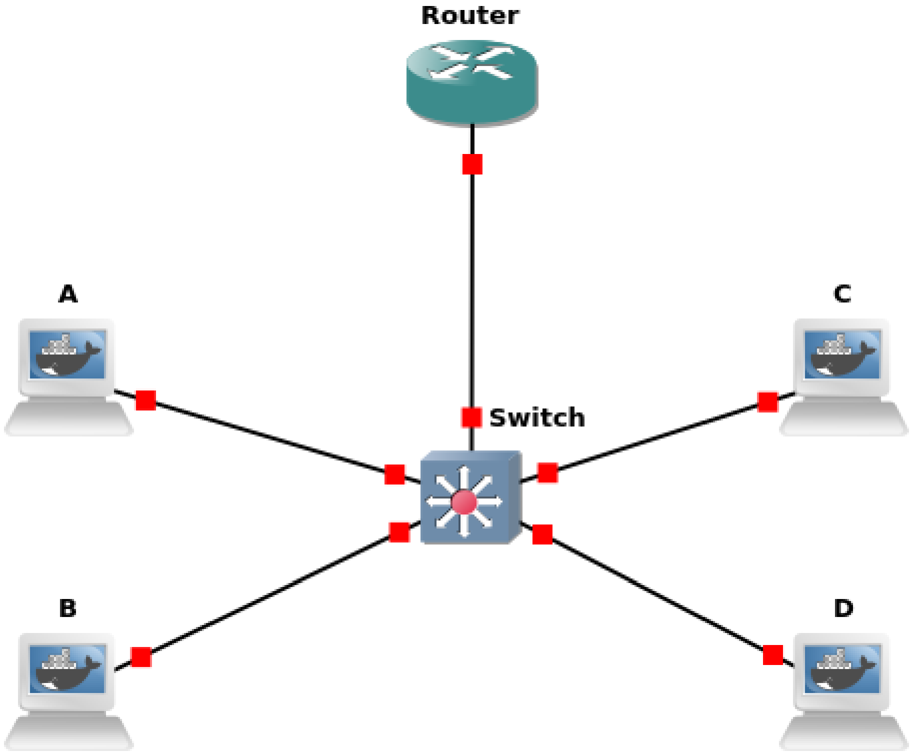

Figure 3:

`A` and `B` should be isolated from `C` and `D` by putting each pair in a separate VLAN:

- `A` and `B` should be in VLAN 1
- `C` and `D` should be in VLAN 18

Why those numbers? Because January 18th is my birthday.

This time, how you plug the machines to the switch is important. Follow this table:

Machine	Port
Router	eth0
A	eth1
B	eth2
C	eth3
D	eth4

The following configuration has to be applied:

Name	IP Addresses
A	- 192.168.1.2/24 (eth0)
B	- 192.168.1.3/24 (eth0)
C	- 192.168.18.2/24 (eth0)
D	- 192.168.18.3/24 (eth0)
Router	- 192.168.1.1/24 (eth0.1) - 192.168.18.1/24 (eth0.18)

Don't forget that a judicious use of the `pre-up` field in your `etc/network/interfaces` can help you create `eth0.1` and `eth0.18` when needed.

How to test

Wireshark is your friend: you should be able to see the tags on the link between the switch and the routers, but not anywhere else.

Why this name?

The namesake for this exercise is of course the 1971 Rolling Stones album.