Interfaces configuration

Network Infrastructures labs

Marco Spaziani Brunella

SAPIENZA
UNIVERSITÀ DI ROMA

Lecture details

- **Readings:**
  - Linux Network Administrators Guide
- **Lecture outline:**
  - Netkit startup scripts
  - Networking daemon
  - TAP Interface
  - DHCP

Do I really have to reconfigure ifaces every time I restart the VMs???

The Netkit way

Netkit allows us to execute commands at startup via specific files. Taking the example of lab_0, we can configure the ifaces on startup creating the 3 startup files for pc1, pc2 and r1 named "VM_NAME.startup". So, for each VM, we create a stratup file wich:

- Brings up ifaces
- Assigns ip addresses to ifaces
- Populates the routing tables (if needed)

/etc/network/interfaces

In real linux enviroments, the ifaces configuration is carried out by the "/etc/network/interfaces" configuration file.
In this file, you can specify the way in wich ifaces are brought up and assign ip addresses, netmasks and gateway (and a lot more!).
For pc1:

```
auto eth0
iface eth0 inet static
        address 10.0.0.100
        netmask 255.255.255.0
        gateway 10.0.0.1
```

We need to create the /etc/network/interfaces file inside every host directory, in order to be mounted at boot up of the VMs.

Networking daemon

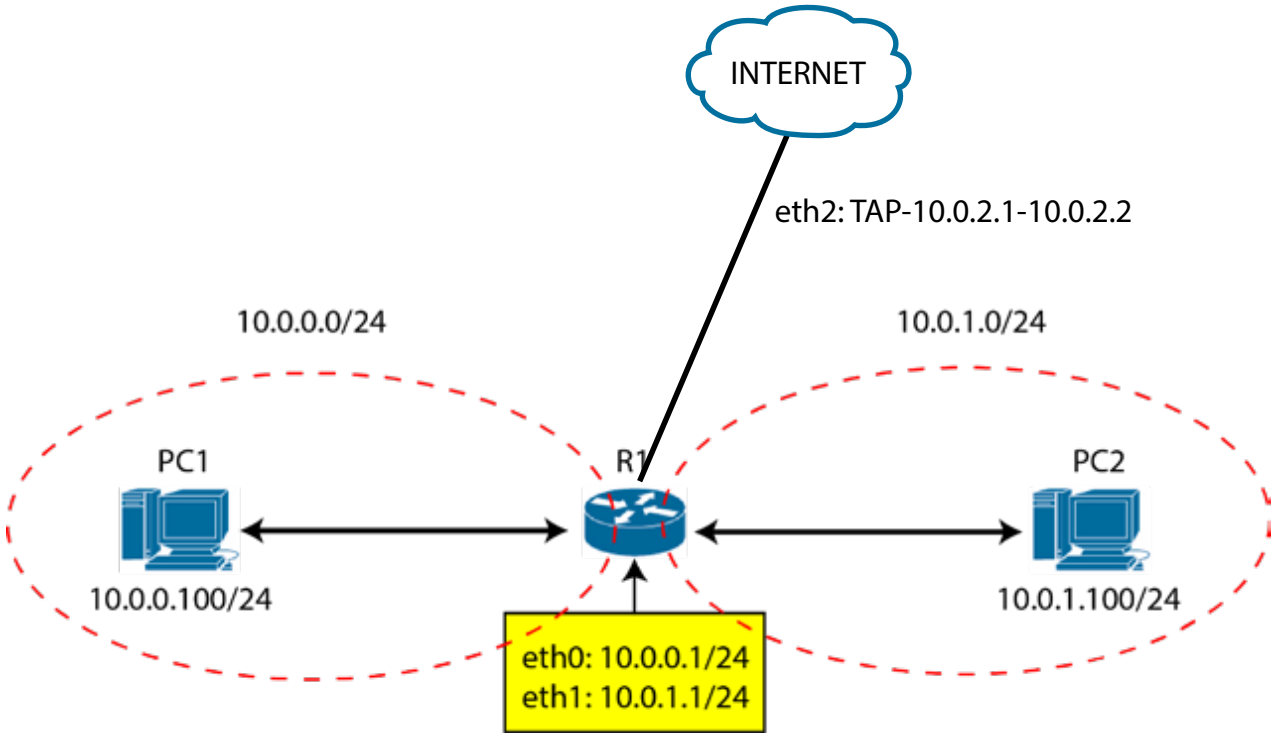
In linux, network, as many other things, are managed by a "daemon".
A daemon is a background process which is configured via configuration files like "/etc/network/interfaces".
Since the boot process of a netkit VM first launches the networking daemon and after mounts the files and folders inside the VM folders in the lab, we need to restart the daemon via:

```
user@localhost:$/etc/init.d/networking restart
```

In order to do that automatically, we can put the command inside the startup script of every host.
Every daemon has a control script inside /etc/init.d/.

Party goes to internet

Netkit allows us to create a layer 2 interface called TAP between a VM and the host machine. With this interface, we can send the entire virtual network to internet. Consider the following topology:



TAP interface

To create a TAP interface on r1, put in lab.conf:

```
r1[2]=tap,10.0.2.1,10.0.2.2
```

This will:

- Create a eth2 interface on r1 with address 10.0.2.2
- Crete a nk_tap_user interface on the host with address 10.0.2.1

Even tough we correctly setup the TAP interface and the gateways on pc1 and pc2, if we try to ping the outside world, we don't get any response.... Why???

Network Address Translation (more on firewall lectures)

Packets coming from pc1 and pc2 are correctly forwarded to the host machine, which forwards them to the outside world (use wireshark on your host machine). But when they come back, the host machine does not have any entry on it's routing table to forward the packets to pc1 and pc2 (use ip route on the host). We have 2 solutions:

- Add manually the route for pc1 and pc2 on the host (boring)
- Every packet coming from pc1 and pc2 passes trough r1 tap interface, but the host machine knows the address of the tap interface!

r1 should change the source ip address of the packets coming from pc1 and pc2 before forwarding them to the host machine. Now, the host machine knows how to forward back them, because to her is if its like they are directed to r1. Once they are on r1, he changes back the correct destination ip addresses and finally pc1 and pc2 take the reply.

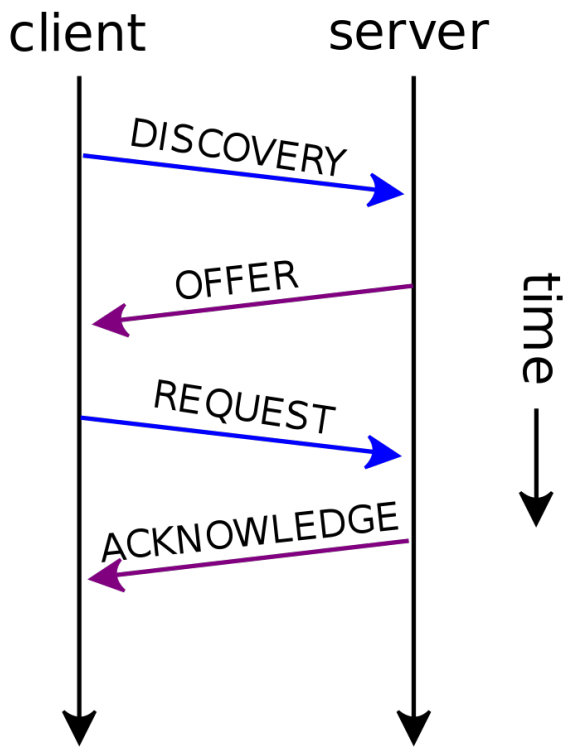
Masquerading

The aforementioned scheme falls under the wide Network Address Translation techniques. In particular, we are Masquerading the source ip address (Source NAT) with the one of r1. In order to accomplish the task, we use *iptables*, which is the Linux firewall. On r1:

```
iptables -t nat -A POSTROUTING -o eth2 -j MASQUERADE
```

Dynamic Host Configuration Protocol

We can also setup r1 to be a DHCP server, distributing IP addresses to pc1 and pc2. Also pc1 and pc2 need to be configured to search for a DHCP server to request for configuration.



DHCP in Linux

In Linux, DHCP is implemented with *isc-dhcp*, which is the most used open-source DHCP implementation. It provides:

- DHCP Client (*dhclient*)
- DHCP Server (*dhcp3-server*)
- DHCP Relay (*dhcrelay*)

ISC DHCP client and sever are already in the NETKIT VM filesystem. To configure pc1 and pc2, just replace the static configuration in /etc/network/interfaces with:

```
iface eth0 inet dhcp
```

DHCP server configuration

To configure r1 as DHCP server, lets create a file dhcpd.conf in /etc/dhcp3:

```
default-lease-time 3600;
option domain-name-servers 8.8.8.8;

subnet 10.0.0.0 netmask 255.255.255.0 {
                range 10.0.0.100 10.0.0.254;

                option routers 10.0.0.1;

}

subnet 10.0.1.0 netmask 255.255.255.0 {

                range 10.0.1.100 10.0.1.254;

                option routers 10.0.1.1;
}
```

To start the DHCP server on r1 boot-up, add this to r1.startup:

```
/etc/init.d/dhcp3-server start
```