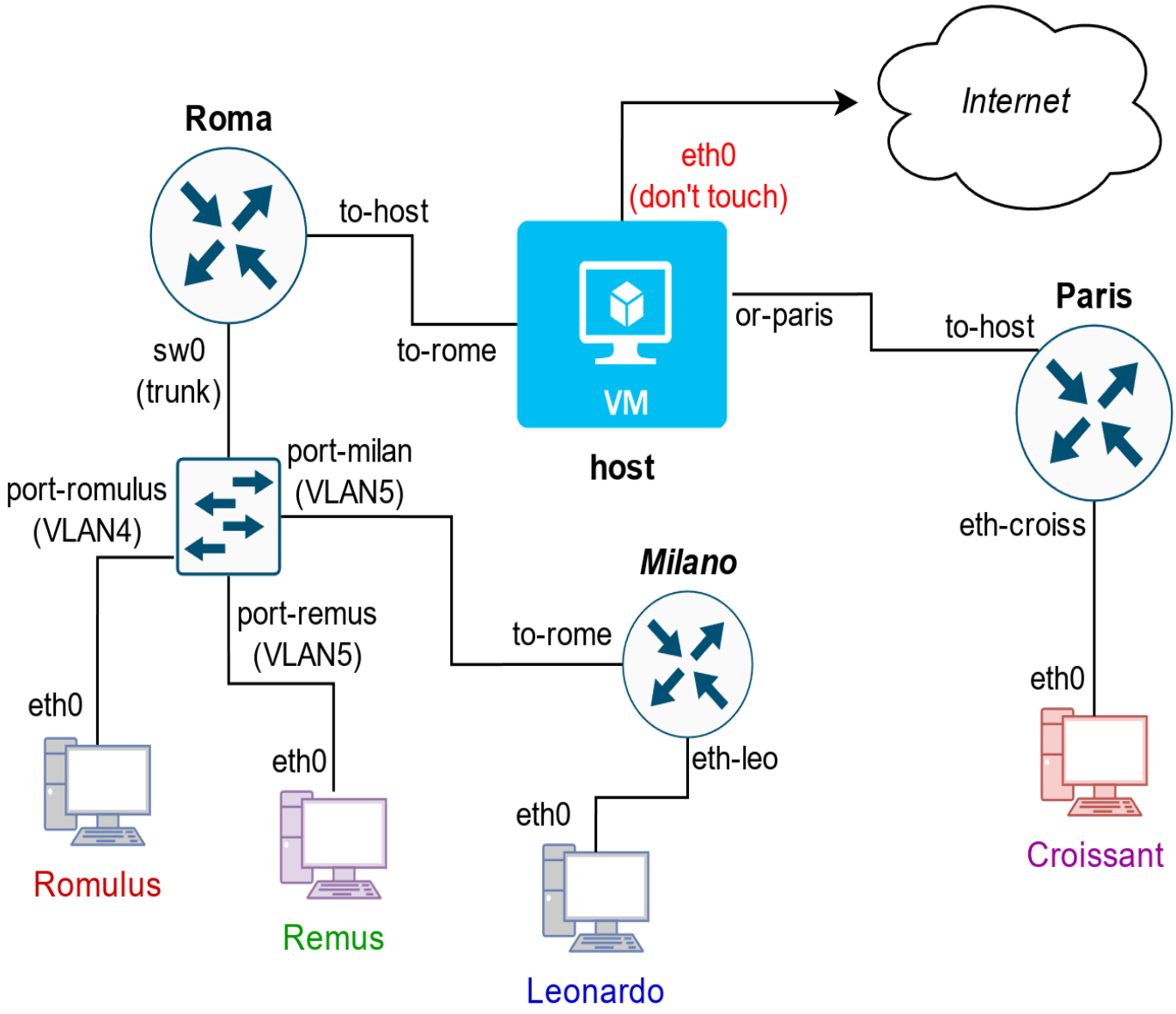
Rezolvare Tema 2 Retele Locale

Cuprins:

```
t2start bogdan.trifan2412
```

Topologie



Rutere:

- host
- Roma
- Milano
- Paris

Contine un singur switch (denumit **sw0**), cu doua VLAN-uri: VLAN4 si VLAN5.

End-device-uri:

- Romulus
- Remus

- Leonardo
- Croissant

Conexiuni:

- host/eth0 <-> Internet (**DON'T TOUCH**)
- host/to-rome <-> Roma/to-host
- host/or-paris <-> Paris/to-host
- Roma <-> sw0 (trunk)
- sw0/port-romulus <-> Romulus/eth0 (VLAN4)
- sw0/port-remus <-> Remus/eth0 (VLAN5)
- sw0/port-milan <-> Milano/to-rome (VLAN5)
- Milano/eth-leo <-> Lenoardo/eth0
- Paris/eth-croiss <-> Croissant/eth0

Conectarre prin SSH

```
eu@localhost$ ssh -J bogdan.trifan2412@fep.grid.pub.ro student@10.9.2.246
```

sau, in `~/.bashrc`:

```
ssh_open_stack ()
{
    if [[ $# != 1 ]]; then
        echo "[ERROR] Expected a single argument: the IP address of the
OpenStack VM!";
        return;
    fi;
    IP=$1;
    ssh -J bogdan.trifan2412@fep.grid.pub.ro student@"$IP"
}
```

```
eu@localhost$ ssh_open_stack 10.9.2.246
```

Sau

```
cat ~/.ssh/config
# RL Tema 2
# ssh -J bogdan.trifan2412@fep.grid.pub.ro student@10.9.2.246
Host rl_tema_2
    User student
    HostName 10.9.2.246
    ProxyJump bogdan.trifan2412@fep.grid.pub.ro
```

```
$ ssh rl_tema_2
```

Asignment

> Moodle Username: `bogdan.trifan2412`

```
root@host:/home/student# cat /root/assignment.txt
USERNAME=bogdan.trifan2412
A=179
B=7
C=106
D=171
E=158
F=73
G=13
H=27
I=46
J=214
K=80
```

Conectarea prin echipamente

```
root@host:/home/student# docker ps | awk '{ print $NF }'
NAMES
mn.Croissant
mn.Leonardo
mn.Remus
mn.Romulus
mn.Paris
mn.Milano
mn.Switch0
mn.Roma
root@host:/home/student# docker ps | awk 'NR>1 { print $NF }' | sed
"s/mn.//g"
Croissant
Leonardo
Remus
Romulus
Paris
Milano
Switch0
Roma
```

```
root@host: go Croissant
```

```
root@host: go Leonardo
```

```
root@host: go Remus
```

```
root@host: go Romulus
```

```
root@host: go Paris
```

```
root@host: go Milano
```

```
root@host: go Switch0
```

```
root@host: go Roma
```

Task 1

Task 1.1 | Subnetare FIXA

1. Subnetați FIX (i.e., dimensiuni egale, maximizare nr. de stații) spațiul 10.$A.$B.0/24 și configurați cu adrese IPv4 toate legăturile din topologie în ordinea cerută (începând cu PRIMA adresă asignabilă), astfel:

- prima subrețea alocată va fi VLAN4, asignare în ordinea: Roma, Romulus;
- a doua subrețea alocată va fi VLAN5, asignare în ordinea: Roma, Milano, Remus;
- a treia subrețea alocată va fi cea dintre Milano și Leonardo (asignare în această ordine);
- a patra subrețea alocată va fi cea dintre Paris și Croissant (la fel, în această ordine);

Solutie:

```
IP = 10.$A.$B.0/24
A = 179
B = 7
----

IP = 10.179.7.0/24
adresa IP este adresa de retea (/24 acopera in intregime primii 3 octeti)
```

```
Reteaua 1 (VLAN 4): Roma, Romulus
Reteaua 2 (VLAN 5): Roma, Milano, Remus
Reteaua 3: Milano, Leonardo
Reteaua 4: Paris, Croissant

Avem 4 subretele (4 <= 2^2) -> avem nevoie de 2 biti de subretea
/(24+2) = /26

Noile adrese vor avea (TOATE) mastile de /26.

O masca de /26 ofera 2^(32-26)=2^6=64 de host-uri (in total, atat
asignabile, cat si neasignabile)


Format: `Retea: prima adresa IP -> ultima adresa IP`

R1: 10.179.7.0/26 -> +2^(32-26)-1 = +63 -> 10.179.7.63/26
R2: 10.179.7.64/26 -> +63 -> 10.179.6.127/26
R3: 10.179.7.128/26 -> +63 -> 10.179.7.191/26
R4: 10.179.7.192/26 -> +63 -> 10.179.7.255/26



R1-Roma-sw04: R/26
R1-Romulus: 10.179.7.2/26

R2-Roma: 10.175.7.65/26
R2-Milano: 10.179.7.66/26
R2-Remus: 10.179.7.67/26
R3-Milano: 10.179.7.129/26
R3-Leonardo: 10.179.7.130/26

R4-Paris: 10.179.7.193/26
R4-Croissant: 10.179.7.194/26
```

Rezultat:

- Reteaua 1 (VLAN 4):
  - R1-Roma-sw04: 10.179.7.1/26
  - R1-Romulus: 10.179.7.2/26
- Reteaua 2 (VLAN 5):
  - R2-Roma: 10.179.7.65/26
  - R2-Milano: 10.179.7.66/26
  - R2-Remus: 10.179.7.67/26
- Reteaua 3:
  - R3-Milano: 10.179.7.129/26
  - R3-Leonardo: 10.179.7.130/26
- Reteaua 4:
  - R4-Paris: 10.179.7.193/26

- R4-Croissant: 10.179.7.194/26

```
# Pentru Reteaua 1 (VLAN 4)
root@Roma:~# ip addr add $IP_R1_Roma_sw04 dev sw0.4
root@Romulus:~# ip addr add $IP_R1_Romulus dev eth0

# Pentru Reteaua 2 (VLAN 5)
root@Roma:~# ip addr add $IP_R2_Roma_sw05 dev sw0.5
root@Milano:~# ip addr add $IP_R2_Milano_to_rome dev to-rome
root@Remus:~# ip addr add $IP_R2_Remus dev eth0

# Pentru Reteaua 3
root@Milano:~# ip addr add $IP_R3_Milano_eth_leo dev eth-leo
root@Leonardo:~# ip addr add $IP_R3_Leonardo dev eth0

# Pentru Reteaua 4
# Pe host-ul Paris:
root@Paris:~# ip addr add $IP_R4_Paris_eth_croiss dev eth-croiss
root@Croissant:~# ip addr add $IP_R4_Croissant dev eth0
```

Adica:

```
# Pentru Reteaua 1 (VLAN 4)
root@Roma:~# ip addr add 10.179.7.1/26 dev sw0.4
root@Romulus:~# ip addr add 10.179.7.2/26 dev eth0

# Pentru Reteaua 2 (VLAN 5)
root@Roma:~# ip addr add 10.179.7.65/26 dev sw0.5
root@Milano:~# ip addr add 10.179.7.66/26 dev to-rome
root@Remus:~# ip addr add 10.179.7.67/26 dev eth0

# Pentru Reteaua 3
root@Milano:~# ip addr add 10.179.7.129/26 dev eth-leo
root@Leonardo:~# ip addr add 10.179.7.130/26 dev eth0

# Pentru Reteaua 4
root@Paris:~# ip addr add 10.179.7.193/26 dev eth-croiss
root@Croissant:~# ip addr add 10.179.7.194/26 dev eth0
```

Task 1.2 | Subnetarea VARIABILA (VLSM)

2. Subnetați OPTIM spațiul 172.30.$C.240/28 + configurați echipamentele (host va avea mereu prima adresă asignabilă) astfel:

- o rețea între host și Roma;
- cealaltă (ultima rămasă): host și Paris.

Solutie:

```
IP = 172.30.$C.240/28
C = 126
----


IP = 172.30.106.240/28
adresa IP este adresa de retea

R5: 2H (Host si Roma) + 2H (IP retea si broadcast) = 4H <= 2^2 -> avem
nevoie de 2 biti de host -> /(32-2) = /30
R6: 2H (Host si Paris) + 2H (IP retea si broadcast) = 4H <= 2^2 -> avem
nevoie de 2 biti de host -> /(32-2) = /30

R5: /30 -> 2^(32-30) = 2^2 = 4 adrese IP (in total, atat asignabile, cat si
neasignabile)
R6: /30 -> 2^(32-30) = 2^2 = 4 adrese IP (in total, atat asignabile, cat si
neasignabile)


R5: 172.30.106.240/30 -> +2^(32-30)-1 = +3 > 172.30.106.243/30
R6: 172.30.106.244/30 -> +3 > 172.30.106.247/30


R5-Host_to-rome: 172.30.106.241/30
R5-Roma_to-host: 172.30.106.242/30

R6-Host_or-paris: 172.30.106.245/30
R6-Paris_to-host: 172.30.106.246/30
```

Rezultat:

- Reteaua 5:
  - R5-Host_to-rome: 172.30.106.241/30
  - R5-Roma_to-host: 172.30.106.242/30
- Reteaua 6:
  - R6-Host_or-paris: 172.30.106.245/30
  - R6-Paris_to_host: 172.30.106.246/30

```
# Pentru Reteaua 5
root@host:~# ip addr add $IP_R5_Host_to_rome dev to-rome
root@Roma:~# ip addr add $IP_R5_Roma_to_host to-host

# Pentru Reteaua 6
root@host:~# ip addr add $IP_R6_Host_or_paris dev or-paris
root@Paris:~# ip addr add $IP_R6_Paris_to_host dev to-host
```

Adica:

```
# Pentru Reteaua 5
root@host:~# ip addr add 172.30.106.241/30 dev to-rome
root@Roma:~# ip addr add 172.30.106.242/30 dev to-host

# Pentru Reteaua 6
root@host:~# ip addr add 172.30.106.245/30 dev or-paris
root@Paris:~# ip addr add 172.30.106.246/30 dev to-host
```

Task 1.3 | Rutare

3. Configurați rutarea IPv4 (default GWs și/sau rute statice) astfel încât toate stațiile să se poată accesa unele pe altele prin adresă IP!

> IP-urile se trec fara mastile de retea 😃.

Task 1.3 | Rutare | Default Gateways

Default Gateways:

```
# Pentru Reteaua 1 (VLAN 4)
root@Romulus:~# ip route add default via $IP_R1_Roma_sw04

# Pentru Reteaua 2 (VLAN 5)
root@Remus:~# ip route add default via $IP_R2_Roma_sw05
root@Milano:~# ip route add default via $IP_R2_Roma_sw05

# Pentru Reteaua 3
root@Leonardo:~# ip route add default $IP_R3_Milano_eth_leo


# Pentru Reteaua 4
root@Croissant:~# ip route add default via $IP_R4_Paris_eth_croiss

# Pentru Reteaua 5
root@Roma:~#ip route add default via $IP_R5_Host_to_rome

# Pentru Reteaua 6
root@Paris:~#ip route add default via $IP_R6_Host_or_paris
```

Adica:

Default Gateways:

```
# Pentru Reteaua 1 (VLAN 4)
root@Romulus:~# ip route add default via 10.179.7.1

# Pentru Reteaua 2 (VLAN 5)
root@Remus:~# ip route add default via 10.179.7.65
```

```
root@Milano:~# ip route add default via 10.179.7.65

# Pentru Reteaua 3
root@Leonardo:~# ip route add default via 10.179.7.129

# Pentru Reteaua 4
root@Croissant:~# ip route add default via 10.179.7.193

# Pentru Reteaua 5
root@Roma:~# ip route add default via 172.30.106.241

# Pentru Reteaua 6
root@Paris:~# ip route add default via 172.30.106.245
```

Task 1.3 | Rutare | Rute Statice

```
ip route add <adresa_retea>/<masca_subretea> via <gateway>
```

Reminder:

- Adresa IP retea R1: 10.179.7.0/26
- Adresa IP retea R2: 10.179.7.64/26
- Adresa IP retea R3: 10.179.7.128/26
- Adresa IP retea R4: 10.179.7.192/26
- Adresa IP retea R5: 172.30.106.240/30
- Adresa IP retea R6: 172.30.106.245/30

Reminder:

Nume interfata	Retea	IP
Roma/sw0.4	R1	10.179.7.1/26
Roma/sw0.5	R2	10.179.7.65/26
Milano/to-rome	R2	10.179.7.66/26
Milano/eth-leo	R3	10.179.7.129/26
Paris/eth-crois	R4	10.179.7.193/26
host/or-paris	R6	172.30.106.245/30
Paris/to-host	R6	172.30.106.246/30
host/to-rome	R5	172.30.106.241/30
Roma/to-host	R5	172.30.106.242/30

```

```

```
root@host:~# ip route add $IP_R4 via $IP_Paris_to_host

root@host:~# ip route add $IP_R1 via $IP_Roma_to_host
root@host:~# ip route add $IP_R2 via $IP_Roma_to_host
root@host:~# ip route add $IP_R3 via $IP_Roma_to_host


root@Roma:~# ip route add $IP_R3 via $IP_R2_Milano_to_rome
```

```
root@host:~# ip route add 10.179.7.192/26 via 172.30.106.246

root@host:~# ip route add 10.179.7.0/26 via 172.30.106.242
root@host:~# ip route add 10.179.7.64/26 via 172.30.106.242
root@host:~# ip route add 10.179.7.128/26 via 172.30.106.242

root@Roma:~# ip route add 10.179.7.128/26 via 10.179.7.66
```

Task 1.3 | Persistenta la restart

Comenzile precedent mentionate nu sunt persistente la restart; IP-urile si rutele trebuiesc configurate intr-un fisier.

Pentru fiecare router (inclusiv host), decomenteaza linia care contine `net.ipv4.ip_forward=1` (linia **28** ar trebui sa fie) din fisierul `/etc/sysctl.conf`.

Pe router **Roma**, decomenteaza linia care contine `net.ipv6.conf.all.forwarding=1` (linia **33**) din fisierul `etc/sysctl.conf`.

In rest, uita-te la fisierele din directorul configs/.

Task 2 | Adresare IPv6

Configurați adrese IPv6 pentru rețeaua VLAN4 și VLAN5 (notă: variabila $VLANID va avea valoarea 4, respectiv 5, cu zero-uri în față până la completarea segmentului de 16 biți): Folosiți spațiul 2024:baba:$B:$A:$VLANID::/96. Aceeași ordine de asignare ca la IPv4. Configurați conectivitate IPv6 între Roma și host: Folosiți spațiul fdee:dada:$C:$D::/64. Prima adresă asignabilă este pentru host, a doua a lui Roma. Configurați rutarea IPv6 pentru a permite comunicarea între toate sistemele cu adresă IPv6. Atenție: echipamentele Leonardo, Paris și Croissant NU vor avea adresă IPv6!

```
# VLAN 4
root@Roma:~# ip -6 addr add 2024:baba:07:179:004::1/96 dev sw0.4
root@Romulus:~# ip -6 addr add 2024:baba:07:179:004::2/96 dev eth0


# VLAN 5
root@Roma:~# ip -6 addr add 2024:baba:07:179:005::1/96 dev sw0.5
```

```
root@Milano:~# ip -6 addr add 2024:baba:07:179:005::2/96 dev to-rome
root@Remus:~# ip -6 addr add 2024:baba:07:179:005::3/96 dev eth0


# Host-Roma
root@host:~# ip -6 addr add fdee:dada:106:171::1/64 dev to-rome
root@Roma:~# ip -6 addr add fdee:dada:106:171::2/64 dev to-host
```

```
# incercare (nu merge rutarea IPV6)


root@Romulus:~# ip -6 route add default via 2024:baba:07:179:004::1

root@Milano:~# ip -6 route add default via 2024:baba:07:179:005::1
root@Remus:~# ip -6 route add default via 2024:baba:07:179:005::1

root@host:~# ip -6 route add default via fdee:dada:106:171::2
```

Task 3 | Accessing Hosts

Editeaza cu `nano`/`vim` textele printate cu **cat**.

Task 3 | Accessing Hosts | **host** (router)

```
root@host:~# touch /etc/hosts.orig
root@host:~# nano -l /etc/hosts.orig
root@host:~# cat /etc/hosts.orig
127.0.0.1 localhost
127.0.1.1 host

# IPv4 of Roma/to-host (router/interface)
172.30.106.242 Roma

# IPv4 of Milano/to-rome (router/interface)
10.179.7.66 Milano

# IPv4 of Paris/to-host (router/interface)
172.30.106.246 Paris

# IPv4 of end-devices
10.179.7.2 Romulus
10.179.7.67 Remus
10.179.7.130 Leonardo
10.179.7.194 Croissant


# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
```

```
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters


root@host:~# nano -l /etc/network/interfaces.d/rl.conf
root@host:~# cat /etc/network/interfaces.d/rl.conf
iface eth0
    up cat /etc/hosts.orig > /etc/hosts
```

Romulus:

```
root@Romulus:~# cat /etc/hosts
127.0.0.1 localhost Romulus



# IPv4 of Roma/sw0.4 (router/interface)
10.179.7.1 Roma

# IPv4 of Milano/to-rome (router/interface)
10.179.7.66 Milano

# IPv4 of Paris/to-host (router/interface)
172.30.106.246 Paris

# IPv4 of end-devices
10.179.7.67 Remus
10.179.7.130 Leonardo
10.179.7.194 Croissant

# IPv6
::1 localhost ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters

root@Romulus:~# cat /etc/network/interfaces.d/rl.conf
# Tema 2 RL - Interface configuration (on ifupdown-ng)
# RTFM: https://github.com/ifupdown-ng/ifupdown-
ng/blob/main/doc/interfaces.scd

# Example entry:
#auto eth0
#iface eth0
#   address 203.0.113.2/24
#   gateway 203.0.113.1

iface eth0
    up cat /etc/hosts.orig > /etc/hosts
```

Task 3 | Accessing Hosts | **Remus** (end-device)

```
root@Remus:~# nano -l /etc/hosts.orig
127.0.0.1 localhost Remus

# IPv4 of host/to-rome (router/interface)
172.30.106.241 host

# IPv4 of Roma/sw0.5 (router/interface)
10.179.7.65 Roma

# IPv4 of Milano/to-rome (router/interface)
10.179.7.66 Milano

# IPv4 of Paris/to-host (router/interface)
172.30.106.246 Paris

# IPv4 of end-devices
10.179.7.2 Romulus
10.179.7.130 Leonardo
10.179.7.194 Croissant

# IPv6
::1 localhost ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters


root@Remus:~# cat /etc/network/interfaces.d/rl.conf
# Tema 2 RL - Interface configuration (on ifupdown-ng)
# RTFM: https://github.com/ifupdown-ng/ifupdown-
ng/blob/main/doc/interfaces.scd

# Example entry:
#auto eth0
#iface eth0
#   address 203.0.113.2/24
#   gateway 203.0.113.1


iface eth0
    up cat /etc/hosts.orig > /etc/hosts
```

Task 3 | Accessing Hosts | **Leonardo** (end-device)

```
root@Leonardo:~# cat /etc/hosts.orig
127.0.0.1 localhost Leonardo
```

```
# IPv4 of host/to-rome (router/interface)
172.30.106.241 host

# IPv4 of Roma/sw0.5 (router/interface)
10.179.7.65 Roma

# IPv4 of Milano/eth-leo (router/interface)
10.179.7.129 Milano

# IPv4 of Paris/to-host (router/interface)
172.30.106.246 Paris

# IPv4 of end-devices
10.179.7.2 Romulus
10.179.7.67 Remus
10.179.7.194 Croissant


# IPv6
::1 localhost ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters

root@Leonardo:~# cat /etc/network/interfaces.d/rl.conf
# Tema 2 RL - Interface configuration (on ifupdown-ng)
# RTFM: https://github.com/ifupdown-ng/ifupdown-
ng/blob/main/doc/interfaces.scd

# Example entry:
#auto eth0
#iface eth0
#   address 203.0.113.2/24
#   gateway 203.0.113.1


iface eth0
    up cat /etc/hosts.orig > /etc/hosts
```

Task 3 | Accessing Hosts | **Croissant** (end-device)

```
root@Croissant:~# cat /etc/hosts.orig
127.0.0.1 localhost Croissant

# IPv4 of host/or-paris (router/interface)
172.30.106.245 host

# IPv4 of Paris/eth-croiss (router/interface)
```

```
10.179.7.193 Paris

# IPv4 of Roma/to-host (router/interface)
172.30.106.242 Roma

# IPv4 of Milano/to-rome (router/interface)
10.179.7.66 Milano


# IPv4 of end-devices
10.179.7.2 Romulus
10.179.7.67 Remus
10.179.7.130 Leonardo


# IPv6
::1 localhost ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters

root@Croissant:~# cat /etc/network/interfaces.d/rl.conf
# Tema 2 RL - Interface configuration (on ifupdown-ng)
# RTFM: https://github.com/ifupdown-ng/ifupdown-
ng/blob/main/doc/interfaces.scd

# Example entry:
#auto eth0
#iface eth0
#   address 203.0.113.2/24
#   gateway 203.0.113.1


iface eth0
    up cat /etc/hosts.orig > /etc/hosts
```

Task 3 | Accesing Hosts | **Roma** (router)

```
root@Roma:~# cat /etc/hosts.orig
127.0.0.1 localhost Roma


# IPv4 of host/to-rome (router/interface)
172.30.106.241 host

# IPv4 of Milano/to-rome (router/interface)
10.179.7.66 Milano

# IPv4 of Paris/to-host (router/interface)
172.30.106.246 Paris
```

```
# IPv4 of end-devices
10.179.7.2 Romulus
10.179.7.67 Remus
10.179.7.130 Leonardo
10.179.7.194 Croissant


# IPv6
::1 localhost ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters

root@Roma:~# cat /etc/network/interfaces.d/rl.conf
# Sample ifupdown network interfaces config

auto to-host
iface to-host
    #address <CIDR/prefix notation>

# VLAN4 sub-interface
auto sw0.4
iface sw0.4
    #address <CIDR/prefix notation>

# VLAN5 sub-interface
auto sw0.5
iface sw0.5
    #address <CIDR/prefix notation>

iface to-host
    up cat /etc/hosts.orig > /etc/hosts
```

Task 3 | Accesing Hosts | **Milano** (router)

Task 3 | Accesing Hosts | **Paris** (router)

```
root@Paris:~# cat /etc/hosts.orig
127.0.0.1 localhost Paris

# IPv4 of host/or-paris
172.30.106.245 host

# IPv4 of Roma/to-host
172.30.106.242 Roma

# IPv4 of Milano/to-rome
10.179.7.66 Milano
```

```
# IPv4 of end-devices
10.179.7.2 Romulus
10.179.7.67 Remus
10.179.7.130 Leonardo
10.179.7.194 Croissant

# IPv6
::1 localhost ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters

root@Paris:~# cat /etc/network/interfaces.d/rl.conf
# Sample ifupdown network interfaces config

auto to-host
iface to-host
    #address <CIDR/prefix notation>

auto eth-croiss
iface eth-croiss
    #address <CIDR/prefix notation>

iface to-host
    up cat /etc/hosts.orig > /etc/hosts
root@Paris:~#
```

Task 4 | Internet connectivity (**iptbables**)

```
root@host:~# iptables -A FORWARD -i eth0 -o to-rome -m state --state
RELATED,ESTABLISHED -j ACCEPT
root@host:~# iptables -A FORWARD -i eth0 -o or-paris -m state --state
RELATED,ESTABLISHED -j ACCEPT
root@host:~# iptables -A FORWARD -i to-rome -o eth0 -j ACCEPT
root@host:~# iptables -A FORWARD -i or-paris -o eth0 -j ACCEPT
root@host:~# iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
```

```
# In order to make them persistent
root@host:~# iptables-save
root@host:~# iptables-save > /etc/iptables/rules.v4
```

Task 5 | Network Address Translation (**NAT**)

Configurați reguli de DNAT pe sistemele Roma și/sau host (după caz), astfel:

- Conexiunile pe Roma la porturile (24000 + $E), (24000 + $F) și (24000 + $G) să conducă la conectarea ssh pe sistemele Romulus, Remus respectiv Leonardo.
- Conectarea pe host la portul (9000 + $K) să conducă la conectarea pe tracker-ul de pe sistemul Milano.

Sfat: aveți grijă cum testați: DNAT-ul va funcționa DOAR dacă veniți dintr-o rețea externă ruterului (e.g., host sau Paris vs Roma)!

Task 5 | NAT | DNAT pentru SSH

```
E = 158
F = 73
G = 13
```

```
root@Roma:~# iptables -t nat -A POSTROUTING -o sw0.4 -j MASQUERADE


# Romulus (port 24158)
root@Roma:~# iptables -t nat -A PREROUTING -p tcp --dport 24158 -j DNAT --
to-destination 10.179.7.2:22

# Remus (port 24073)
root@Roma:~# iptables -t nat -A PREROUTING -p tcp --dport 24073 -j DNAT --
to-destination 10.179.7.67:22

# Leonardo (port 24013)
root@Roma:~# iptables -t nat -A PREROUTING -p tcp --dport 24013 -j DNAT --
to-destination 10.179.7.130:22

root@Roma:~# iptables -A INPUT -p tcp --dport 24158 -j ACCEPT
root@Roma:~# iptables -A INPUT -p tcp --dport 24073 -j ACCEPT
root@Roma:~# iptables -A INPUT -p tcp --dport 24013 -j ACCEPT

# Persistenta configuratiei
root@Roma:~# iptables-save
root@Roma:~# iptables-save > /etc/iptables/rules.v4
```

Task 5 | NAT | DNAT pentru tracker-ul de pe Milano

```
K = 80
```

Task 5 | NAT | DNAT pentru tracker-ul de pe Milano | Router-ul host

```
root@host~# iptables -t nat -A PREROUTING -i to-rome -p udp --dport 9080 -j
DNAT --to-destination <IP_Roma/to-rome>:9080
```

```
root@host~# iptables -t nat -A PREROUTING -i or-paris -p udp --dport 9080 -
j DNAT --to-destination <IP_Roma/to-rome>:9080
```

```
IP_Roma/to-rome = 172.30.106.242
```

Deci:

```
root@host:~# iptables -t nat -A PREROUTING -i to-rome -p udp --dport 9080 -
j DNAT --to-destination 172.30.106.242:9080
root@host:~# iptables -t nat -A PREROUTING -i or-paris -p udp --dport 9080
-j DNAT --to-destination 172.30.106.242:9080
root@host:~# iptables-save > /etc/iptables/rules.v4
```

Task 5 | NAT | DNAT pentru tracker-ul de pe Milano | Router-ul Roma

```
root@Roma:~# iptables -t nat -A PREROUTING -i to-host -p udp --dport
<port_on_Rome>  -j DNAT --to-destination <IP_Milano/to-rome>:
<port_on_Milano>
root@Roma:~# iptables -t nat -A POSTROUTING -o sw0.5 -p udp --dport
<port_on_Milano> -d <IP_Milano/to-rome> -j SNAT --to-source
<IP_default_gateway_of_Roma>
root@Roma:~# iptables-save > /etc/iptables/rules.v4
```

```
IP_Milano/to-rome = 10.179.7.66

Rome's default gateway = host/to-rome = 172.30.106.241
```

```
root@Milano:~$ netstat -tuln
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:23              0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:25              0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:53              0.0.0.0:*               LISTEN
tcp6       0      0 :::143                  :::*                    LISTEN
tcp6       0      0 :::22                   :::*                    LISTEN
tcp6       0      0 :::21                   :::*                    LISTEN
tcp6       0      0 :::25                   :::*                    LISTEN
tcp6       0      0 :::53                   :::*                    LISTEN
tcp6       0      0 :::993                  :::*                    LISTEN
udp        0      0 0.0.0.0:53              0.0.0.0:*
```

```
udp        0      0 0.0.0.0:9123            0.0.0.0:*
udp6       0      0 :::53                   :::*
```

Pentru mai multe detalii:

```
# Spune si procesul care l-a creat
root@Milano:~# netstat -tulpn
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
PID/Program name
tcp        0      0 0.0.0.0:25              0.0.0.0:*               LISTEN
872/master
tcp        0      0 0.0.0.0:23              0.0.0.0:*               LISTEN
245/inetd
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
255/sshd: /usr/sbin
tcp        0      0 0.0.0.0:53              0.0.0.0:*               LISTEN
264/dnsmasq
tcp6       0      0 :::143                  :::*                    LISTEN
87/couriertcpd
tcp6       0      0 :::25                   :::*                    LISTEN
872/master
tcp6       0      0 :::21                   :::*                    LISTEN
250/vsftpd
tcp6       0      0 :::22                   :::*                    LISTEN
255/sshd: /usr/sbin
tcp6       0      0 :::53                   :::*                    LISTEN
264/dnsmasq
tcp6       0      0 :::993                  :::*                    LISTEN
88/couriertcpd
udp        0      0 0.0.0.0:53              0.0.0.0:*
264/dnsmasq
udp        0      0 0.0.0.0:9123            0.0.0.0:*
1022/python3
udp6       0      0 :::53                   :::*
264/dnsmasq
```

Tracker-ul de pe Milan:

- Port pe care primeste: 9123
- Port pe care trimite: 9123

Portul de interes (al tracker-ului) este port-ul **9123**, care este un port **UDP**.

Pentru a ne conecta la portul **UDP 9123**:

```
$ netcat -u Milano 9123
```

> Orice scriem la **stdin** trebuie sa primim exact la **stdout**.

```
root@Roma:~# iptables -t nat -A PREROUTING -i to-host -p udp -m udp --dport
9080 -j DNAT --to-destination 10.179.7.66:9123
root@Roma:~# iptables -t nat -A POSTROUTING -d 10.179.7.66/32 -o sw0.5 -p
udp -m udp --dport 9123 -j SNAT --to-source 172.30.106.241
root@Roma:~# iptables-save > /etc/iptables/rules.v4
```

Task 6 | Filtare pachete (iptables)

Configurați filtrarea de pachete pe Roma / Paris / Milano (după caz) astfel încât:

- conexiunile SMTP și Telnet inițiate de pe sistemul Remus în afara rețelei lui să fie blocate (inclusiv către alte rutere!);
- conexiunile către tracker-ul ce rulează pe sistemul Milano să nu fie permise de la Croissant.
- blocați TOATE conexiunile externe (i.e., de pe IP-urile din afara stației) către Leonardo, mai puțin protocoalele icmp și ssh.
- atenție: NU blocați conexiunile inițiate de Leonardo și nici răspunsurile de la acestea! folosiți reguli stateful (i.e. connection tracking)!

Task 6 | Blocare initieare conexiuni **SMTP** si **Telnet** de pe **Remus**

```
# Blocare SMTP (port 25) de la Remus catre exterior
root@Roma:~# iptables -A FORWARD -s Remus -p tcp --dport 25 -j DROP

# Blocare Telnet (port 23) de la Remus catre exterior
root@Roma:~# iptables -A FORWARD -s Remus -p tcp --dport 23 -j DROP
root@Roma:~# iptables-save > /etc/iptables/rules.v4
```

Task 6 | Blocarea conexiunilor catre tracker-ul de pe **Milano** de la **Croissant**

> REMINDER: Tracker-ul de pe Milano se afla pe portul **9123**, care este port **UDP**.

```
root@Paris:~# iptables -A FORWARD -s Croissant -d Milano -p udp --dport
9123 -j DROP
root@Paris:~# iptables-save > /etc/iptables/rules.v4
```

Task 6 | Blocarea conexiunilor catre **Leonardo**, in afara de **icmp** si **ssh**

```
# A se rula comenzile in aceasta ordine (ordinea conteaza)
root@Milano:~# iptables -P INPUT ACCEPT
root@Milano:~# iptables -P FORWARD ACCEPT
root@Milano:~# iptables -P OUTPUT ACCEPT
root@Milano:~# iptables -A FORWARD -d Milano -p icmp -j ACCEPT
```

```
root@Milano:~# iptables -A FORWARD -d Milano -p tcp -m tcp --dport 22 -j
ACCEPT
root@Milano:~# iptables -A FORWARD -d Milano -m state --state
RELATED,ESTABLISHED -j ACCEPT
root@Milano:~# iptables -A FORWARD -d Milano -j DROP
```

Task 7 | SSH Keys

> In caz sa trebuie sa refaci de la 0 generarea cheilor SSH, uita-te in directorul `/configs/` dupa perechile de chei SSH, da paste la ele si doar `ssh-add`.

Task 7 | SSH Keys | From **Host** to others (Romulus, Remus, Leonardo)

```
# Generating SSH key-pairs for other devices, for HOST to connect to
student@host:~$ ssh-keygen -t ed25519 -f /home/student/.ssh/romu -N ""
student@host:~$ ssh-keygen -t ed25519 -f /home/student/.ssh/remu -N ""
student@host:~$ ssh-keygen -t ed25519 -f /home/student/.ssh/leo -N ""
```

```
student@host:~$ ssh student@Romulus "mkdir -p ~/.ssh && cat >>
~/.ssh/authorized_keys" < /home/student/.ssh/romu.pub
student@host:~$ ssh student@Remus "mkdir -p ~/.ssh && cat >>
~/.ssh/authorized_keys" < /home/student/.ssh/remu.pub
student@host:~$ ssh student@Leonardo "mkdir -p ~/.ssh && cat >>
~/.ssh/authorized_keys" < /home/student/.ssh/leo.pub
```

Task 7 | SSH Keys | From **Romulus** to others (Host, Remus, Leonardo)

```
# Generating SSH key-pairs for other devices, for HOST to connect to
student@Romulus:~$ ssh-keygen -t ed25519 -f /home/student/.ssh/host -N ""
student@Romulus:~$ ssh-keygen -t ed25519 -f /home/student/.ssh/remu -N ""
student@Romulus:~$ ssh-keygen -t ed25519 -f /home/student/.ssh/leo -N ""
```

> Copiaza manual `student@Remus:~$ /home/student/.ssh/host.pub` la `student@host:~$ ~/.ssh/authorized_keys`

```
student@Romulus:~$ ssh student@Remus "mkdir -p ~/.ssh && cat >>
~/.ssh/authorized_keys" < /home/student/.ssh/remu.pub
student@Romulus:~$ ssh student@Leonardo "mkdir -p ~/.ssh && cat >>
~/.ssh/authorized_keys" < /home/student/.ssh/leo.pub
```

Task 7 | SSH Keys | From **Remus** to others (Host, Romulus, Leonardo)

```
# Generating SSH key-pairs for other devices, for HOST to connect to
student@Remus:~$ ssh-keygen -t ed25519 -f /home/student/.ssh/host -N ""
student@Remus:~$ ssh-keygen -t ed25519 -f /home/student/.ssh/romu -N ""
student@Remus:~$ ssh-keygen -t ed25519 -f /home/student/.ssh/leo -N ""
```

> Copiaza manual `student@Remus:~$ /home/student/.ssh/host.pub` la `student@host:~$ ~/.ssh/authorized_keys`

```
student@Remus:~$ ssh student@Romulus "mkdir -p ~/.ssh && cat >>
~/.ssh/authorized_keys" < /home/student/.ssh/romu.pub
student@Remus:~$ ssh student@Leonardo "mkdir -p ~/.ssh && cat >>
~/.ssh/authorized_keys" < /home/student/.ssh/leo.pub
```

Task 7 | SSH Keys | From **Leonardo** to others (Host, Romulus, Remu)

```
# Generating SSH key-pairs for other devices, for HOST to connect to
student@Leonardo:~$ ssh-keygen -t ed25519 -f /home/student/.ssh/host -N ""
student@Leonardo:~$ ssh-keygen -t ed25519 -f /home/student/.ssh/romu -N ""
student@Leonardo:~$ ssh-keygen -t ed25519 -f /home/student/.ssh/remu -N ""
```

> Copiaza manual `student@Leonardo:~$ /home/student/.ssh/host.pub` la `student@host:~$ ~/.ssh/authorized_keys`

```
student@Leonardo:~$ ssh student@Romulus "mkdir -p ~/.ssh && cat >>
~/.ssh/authorized_keys" < /home/student/.ssh/romu.pub
student@Leonardo:~$ ssh student@Remus "mkdir -p ~/.ssh && cat >>
~/.ssh/authorized_keys" < /home/student/.ssh/remu.pub
```

Task 8 | Scanarea **pe Romulus** a **serverului mail** (SMTP si IMAP) **de pe Milano**

Pe **Romulus**, am scris urmatorul script de **python**:

> NOTA: Daca rulez checker-ul de doua ori consecutiv, imi spune *"Mailbox is broken"*.

> Inainte de rula checker-ul, deconecteaza-te de pe VM si conecteaza-te din nou (prin **ssh**).

```
student@Romulus:~$ touch scan-support-tickets
student@Romulus:~$ chmod +x scan-support-tickets
student@Romulus:~$ nano -l scan-support-tickets
```

```
#!/usr/bin/env python3

import imaplib
import smtplib
from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText
import email


# IP Milano
IMAP_SERVER = '10.179.7.66'
SMTP_SERVER = '10.179.7.66'

USERNAME = 'support'
PASSWORD = 'Pierdut$Cont1337'

ticket_counter = 1

def fetch_support_tickets():
  global ticket_counter
  mail = imaplib.IMAP4_SSL(IMAP_SERVER)
  mail.login(USERNAME, PASSWORD)
  mail.select('inbox')

  result, data = mail.search(None, '(SUBJECT "Support Ticket")')
  email_ids = data[0].split()

  for email_id in email_ids:
      result, msg_data = mail.fetch(email_id, '(RFC822)')
      raw_email = msg_data[0][1]

      msg = email.message_from_bytes(raw_email)

      sender_email = msg['From']
      subject_text = msg['Subject']

      if 'Support Ticket' in subject_text:
          send_reply(sender_email, subject_text)
          ticket_counter += 1

  mail.logout()

def send_reply(to_email, subject):
  body = f"Hi,\n" \
         f"\n" \
         f"Your ticket named '{subject}' was registered as #
{ticket_counter}.\n" \
         f"\n" \
         f"Thank you for your patience!\n"


  msg = MIMEMultipart()
  msg['From'] = USERNAME
```

```
    msg['To'] = to_email
    msg['Subject'] = f'Re: {subject}'
    msg.attach(MIMEText(body, 'plain'))

    try:
        with smtplib.SMTP(SMTP_SERVER, 25) as server:
          server.send_message(msg)
          return
    except Exception as e:
      return

if __name__ == "__main__":
    fetch_support_tickets()
```

Task 9 | Sincronizarea automata de pe **Remus** pe **Roma**

Ne uitam pe **Roma**:

```
student@Roma:~$ cat /etc/passwd
```

Observ **user**-ul:

```
remus-backup:x:1001:1001::/var/remus-backup:/bin/bash
```

NU merge:

```
student@Roma:~$ su remus-backup
```

Dar merge:

```
student@Roma:~$ sudo su remus-backup
remus-backup@Roma:/home/student$
```

Sau (merge si):

```
student@Roma:~$ sudo su
root@Roma:student# su remus-backup
remus-backup@Roma:/home/student$
```

Prin urmare, vreu sa ma conectez cu **user**-ul *remus-backup*, pe **host**-ul *Roma*.

```
student@Remus:~$ ssh-keygen -t ed25519 -f /home/student/.ssh/roma -N ""
student@Remus:~$ ssh student@Roma "mkdir -p ~/.ssh && cat >>
~/.ssh/authorized_keys" < /home/remus-backup/.ssh/roma.pub
```

> Daca nu merge a doua comanda, copiaza manual **cheia publica** (**roma.pub**) de pe **Romus** la **/home/remus-backup/.ssh/authorized_keys** pe **Roma**.

Adauga o noua intrare in fisierul de configuratie SSH a lui **Romus**:

```
student@Remus:~$ nano -l ~/.ssh/config
```

```
Host Roma
    HostName Roma
    IdentityFile /home/student/.ssh/roma
```

Acum ne putem conecta fara parola:

```
student@Remus:~$ ssh remus-backup@Roma
```

```
student@Remus:~$ mkdir ~/scripts
student@Remus:~$ touch ~/scripts/auto-backup
student@Remus:~$ chmod +x ~/scripts/auto-backup
```

```
student@Remus:~$ nano -l ~/scripts/auto-backup
```

```
#!/bin/bash

USER="remus-backup"
IP="Roma"

terminate() {
    echo "Se opreste rularea script-ului!"
    exit 0
}

trap terminate SIGINT SIGTERM

while true; do
    rsync -av --exclude='.*' --exclude='*VIRUS*' \
```

```
        --chown=$USER:$USER --chmod=ug=rw,o=r \
        /home/student/Documents/* $USER@$IP:/var/remus-backup \
        > /dev/null 2>&1 &

    sleep 1
done
```

Task 10 | **Wireguard tunnel**

Dorim să conectăm rețelele Milano-Leonardo și Paris-Croissant împreună printr-un tunel Wireguard: Pentru rețeaua din tunel, folosiți spațiul 10.$H.$J.96/30; prima adresă asignabilă este a lui Milano, iar ce-a de-a doua a lui Paris; Va trebui să puteți accesa adresele IP ale capetelor de tunel și de pe Leonardo și Croissant! Denumiți interfețele de tunel wg-rl la ambele capete. Tunelul wireguard trebuie să fie persistent! (folosiți hook-uri în interfaces).

> Un link foarte bun de inspiratie (si exercitiu de practica) este **Laboratorul 10 de ISC**: https://ocw.cs.pub.ro/courses/isc/labs/10

De aici, am aflat:

```
# Generare pereche chei publica-privata pentru WireGuard (va afisa cheia publica)
wg genkey | tee wg-priv.key | wg pubkey | tee wg-pub.key
# Q: what does `tee` do? (`man` it!)
```

Template fisier de configuratie interfata wireguard:

```
[Interface]
PrivateKey = <paste-your-private-key>
ListenPort = 55820

[Peer]
PublicKey = <paste-your-colleagues's-pub-key>
Endpoint = <colleague-VM-IP>:55820
AllowedIPs = <your-tunnel-subnet>/<mask>
```

> NOTA: **WireGuard** functioneaza by default pe portul `51820`.

WireGuard se configureaza oarecum ca **SSH**: are nevoie de o pereche de chei publica-privata.

```
# Comanda nu va afisa nimic, va scri cheile direct in fisiere
root@Milano:~# wg genkey | tee wg-privatekey | wg pubkey > wg-publickey
```

```
root@Paris:~# wg genkey | tee wg-privatekey | wg pubkey > wg-publickey
```

Pentru rețeaua din tunel, folosiți spațiul 10.$H.$J.96/30; prima adresă asignabilă este a lui Milano, iar ce-a de-a doua a lui Paris;

```
IP retea tunnel = 10.$H.$J.96/30

H=27
J=214

IP retea tunnel  = 10.27.214.96/30
IP privat Milano = IP Milano/wg-rl = 10.27.214.97/30
IP privat Paris  = IP Paris/wg-rl  = 10.27.214.98/30
```

Pe **Milano**:

```
root@Milano:~# nano -l /etc/wireguard/wg-rl.conf
```

```
[Interface]
Address = <IP-privat-Milano/wg-rl>
SaveConfig = true
ListenPort = 51820
PrivateKey = <cheie-privata-Milano>

# Paris/to-host
[Peer]
PublicKey = <cheie-publica-Paris>
# AllowedIPs = <tunnel-subnet>/<mask>
AllowedIPs = <IP-retea-tunel>
# Endpoint = Paris/to-host
Endpoint = <IP-Paris/to-host>:51820
PersistentKeepalive = 60
```

Adica:

```
[Interface]
Address = 10.27.214.97/30
SaveConfig = true
ListenPort = 51820
PrivateKey = SKff+08t1TVayJj3Ob2lemtSG0G9fXqGCvyUPYDFCUc=

# Paris/to-host
[Peer]
PublicKey = ki3M91uPAU/ooKr9dogvEq7R0vHS0gQNkx83MQp7Xyo=
```

```
# AllowedIPs = <tunnel-subnet>/<mask>
AllowedIPs = 10.27.214.96/30
# Endpoint = Paris/to-host
Endpoint = 172.30.106.246:51820
PersistentKeepalive = 60
```

Pe **Paris**:

```
root@Paris:~# nano -l /etc/wireguard/wg-rl.conf
```

```
[Interface]
Address = <IP-privat-Paris/wg-rl>
SaveConfig = true
ListenPort = 51820
PrivateKey = <cheie-privata-Paris>

# Milano/wg-rl
[Peer]
PublicKey = <cheie-publica-Milano>
# AllowedIPs = <tunnel-subnet>/<mask>
AllowedIPs = <IP-retea-tunel>
# Endpoint = Milano/to-rome
Endpoint = <IP-Milano/to-rome>:51820
PersistentKeepalive = 60
```

Adica:

```
[Interface]
Address = 10.27.214.98/30
SaveConfig = true
ListenPort = 51820
PrivateKey = cMxJNvd5rErTXyecAg4rlCmRKHohyaaz6KzYBF/qVG8=

# Milano/wg-rl
[Peer]
PublicKey = 8Tsqi0T1DWijG7Zb0QfnWH7zcA7NnlUsGRuaUzqzR2Q=
# AllowedIPs = <tunnel-subnet>/<mask>
AllowedIPs = 10.27.214.96/30
# Endpoint = Milano/to-rome
Endpoint = 10.179.7.66:51820
PersistentKeepalive = 60
```

Dupa am configurat cum ar trebui sa arate interfetele, ele trebuiesc pornite:

```
root@Milano:~# wg-quick up wg-rl
```

```
root@Paris:~# wg-quick up wg-rl
```

> Foarte important!
>
> Pentru a modifica fisierul de configuratie al interfetei de **WireGuard**, interfata trebuie mai intai orpita cu `wg-quick down wg-rl` (altfel nu va salva nimica) si repornita cu `wg-quick down wg-rl`.

> NOTA: Aparent, fisierele de configuratie nu salveaza comentariile 😦.

Comanda `wg-quick up wg-rl` nu este persistenta la restart, de aceea trebuie adaugata intr-un **hook** de **up** in *"interfata principala"* a router-ului:

Pentru **Milano**:

```
root@Milano:~# nano -l /etc/network/interfaces.d/rl.conf
```

```
auto to-rome

iface to-rome inet static
  ....
    # Interfata virtuala pentru WireGuard (VPN)
    up wg-quick up wg-rl
....
```

La fel si pe **Paris**:

```
root@Paris:~# nano -l /etc/network/interfaces.d/rl.conf
```

```
auto to-host
iface to-host
  ....
    # Interfata virtuala pentru WireGuard (VPN)
    up wg-quick up wg-rl
....
```

In acest moment, tunelul functioneaza de la up capat la altul. Putem da **ping** de la o adresa privata la alta. Totusi, in cerinta se vrea ca end-device-urile **Leonardo** si **Croissant** (direct conectate la cele doua rutere) sa poate accesa si ele ambele capete de tunel.

Pentru asta, imi doresc ca oricarui pachet ce pleaca de la **Leonardo** si vrea sa ajunga la capatul celalalt al tunelului privat (**Paris/wg-rl**), sa i se modifice adresa IP sursa (publica) cu adresa IP privata a interfetei `wg-rl` de pe **Milano**.

```
root@Milano:~# iptables -t nat -A POSTROUTING -s <IP-retea-Milano-
Leonardo>/<mask> -d <IP-Paris/wg-rl>/<mask> -o wg-rl -j SNAT --to-source
<IP-Milano/wg-rl>
```

Adica:

```
root@Milano:~# iptables -t nat -A POSTROUTING -s 10.179.7.128/26 -d
10.27.214.98/30 -o wg-rl -j SNAT --to-source 10.27.214.97
```

```
# Pentru persistenta la restart
root@Milano:~# iptables-save > /etc/iptables/rules.v4
```

Leonardo poate accesa acum ambele capete ale tunelului.

Repetam procesul si pentru **Croissant**, schimband doar niste adrese IP: oricarui pachet ce pleaca de la **Croissant** si are ca destinatie capatul opus al tunelului (**Milano/wg-rl**), sa i se modifice adresa IP sursa (publica) cu adresa IP privata a interfetei `wg-rl` de pe **Paris**.

```
root@Pris:~# iptables -t nat -A POSTROUTING -s <IP-retea-Paris-
Croissant>/<mask> -d <IP-Milano/wg-rl>/<>mask -o wg-rl -j SNAT --to-source
<IP-Milano/wg-rl>
```

Adica:

```
root@Pris:~# iptables -t nat -A POSTROUTING -s 10.179.7.192/26 -d
10.27.214.97/30 -o wg0 -j SNAT --to-source 10.27.214.98
```

```
# Pentru persistenta la restart
root@Paris:~# iptables-save > /etc/iptables/rules.v4
```

Croissant poate accesa acum ambele capete ale tunelului.

Task 11 | Pin your hair

Pe stația Milano va rula un serviciu securizat pe portul TCP 1000+$K care va asculta DOAR pe adresa wireguard creată la task-ul anterior (checkerul îl va porni automat; pentru testare puteți folosi nc cu argumentul -l și IP-ul WireGuard (+ desigur, portul), trebuie să puteți trimite mesaje bidirecționale prin portul forwardat). Configurați DNAT pe Paris astfel încât să poată primi conexiuni pe portul 1000+$K și să le forwardeze prin tunelul Wireguard către Milano, același port. Accesul la serviciu prin Paris:1000+$K va trebui să fie funcțional din orice rețea! Restricție: este obligatoriu să folosiți DOAR iptables și/sau rutare (care ar trebui să fie deja configurată la ex. anterior) pentru a rezolva acest exercițiu, e.g., nu e voie să folosiți un serviciu auxiliar care să asculte pe portul 1000+$K și să redirecționeze pachetele la Milano! Hint: folosiți tcpdump cu încredere când nu funcționează ceva 😉

K=80

```
root@Paris:~# iptables -A FORWARD -d 10.27.214.97/32 -p tcp -m tcp --dport
1080 -j ACCEPT
root@Paris:~# iptables -t nat -A PREROUTING -s 10.179.7.192/26 -p tcp -m
tcp --dport 1080 -j DNAT --to-destination 10.27.214.97:1080
root@Paris:~# iptables -t nat -A PREROUTING -p tcp -m tcp --dport 1080 -j
DNAT --to-destination 10.27.214.97:1080
root@Paris:~# iptables -t nat -A POSTROUTING -o wg-rl -j MASQUERADE
root@Paris:~# iptables -t nat -A POSTROUTING -s 10.179.7.192/26 -d
10.27.214.96/30 -o wg-rl -j SNAT --to-source 10.27.214.98

root@Paris:~# iptables-save > /etc/iptables/rules.v4
```