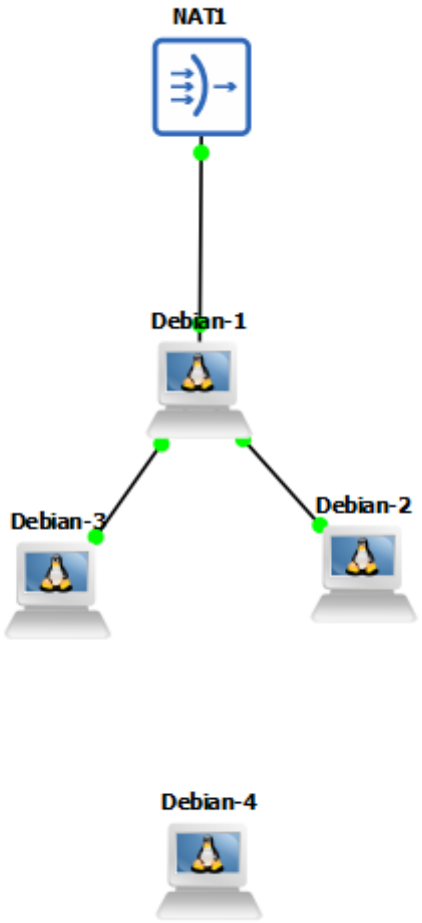
TD 3 : Adressage IP Dynamique

1 Accès internet

Avant de commencer toute manipulation, il est nécessaire de supprimer le cable entre la machine 1, qui joue le rôle de routeur, et la machine 4. A la place, cette interface va nous servir à relié le routeur à un NAT. Voici la configuration attendu :



Nous allons ensuite modifier l'interface de la machine Debian centrale connectée au Nat afin qu'il récupère son adresse IP en DHCP. Dans notre cas, il s'agit de l'interface **enp2s2**. Pour cela, nous allons modifier le fichier de configuration des interfaces pour configurer l'interface enp2s2 pour obtenir automatiquement une adresse IP à l'aide du protocole DHCP à l'aide de la commande : **sudo nano /etc/network/interfaces**

Il est ensuite nécessaire de commenter ou supprimer les lignes concernant l'adressage static de l'interface **enp2s2** écrite dans le précédent TP et de décommenter celles concernant le DHCP en indiquant l'interface voulue. Voici le résultat après avoir réalisé les modifications :

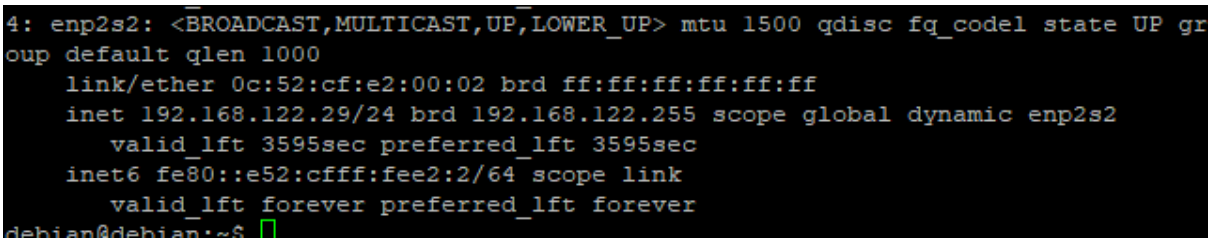
```
  GNU nano 7.2
# DHCP config for enp2s2
auto enp2s2
iface enp2s2 inet dhcp

# Static config for ens4
auto enp2s0
iface enp2s0 inet static
    address 192.168.0.62
    netmask 255.255.255.192

auto ens1
iface ens1 inet static
    address 192.168.0.126
    netmask 255.255.255.192

#auto enp2s2
#iface enp2s2 inet static
#    address 192.168.0.190
#    netmask 255.255.255.192
```

Il est nécessaire d'exécuter cette commande pour appliquer les modifications : **sudo systemctl restart networking**
puis : **sudo ifdown enp2s2 && sudo ifup enp2s2** pour se libérer de l'adresse static défini dans le précédant tp.

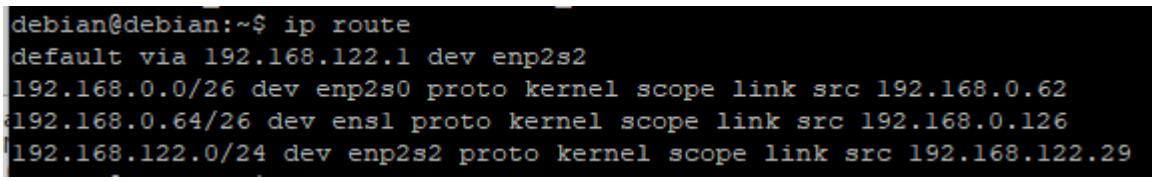
Vous devriez voir les échanges du protocole DHCP dans la console :

```
DHCPRELEASE of 192.168.122.29 on enp2s2 to 192.168.122.1 port 67
Internet Systems Consortium DHCP Client 4.4.3-P1
Copyright 2004-2022 Internet Systems Consortium.
All rights reserved.
For info, please visit https://www.isc.org/software/dhcp/

Listening on LPF/enp2s2/0c:52:cf:e2:00:02
Sending on   LPF/enp2s2/0c:52:cf:e2:00:02
Sending on   Socket/fallback
DHCPDISCOVER on enp2s2 to 255.255.255.255 port 67 interval 5
DHCPOFFER of 192.168.122.29 from 192.168.122.1
DHCPREQUEST for 192.168.122.29 on enp2s2 to 255.255.255.255 port 67
DHCPACK of 192.168.122.29 from 192.168.122.1
bound to 192.168.122.29 -- renewal in 1473 seconds.
```

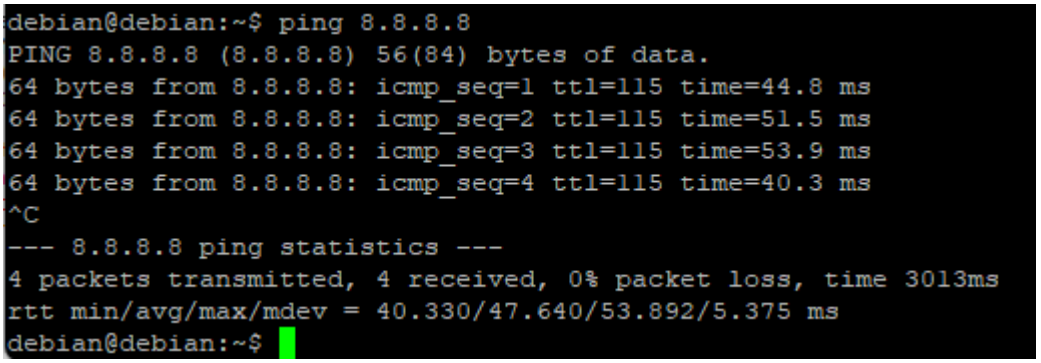
Pour voir la nouvelle adresse ip associée à l'interface, vous pouvez entrer la commande **ip a**. Normalement, vous verrez une mention "scope global dynamic" s'afficher en fin de ligne :



De plus, une route par défaut a normalement été ajoutée avec comme modèle : *default via <gateway_nat> dev <interface>* . Vous pouvez vérifier cette information avec la commande : **ip route**



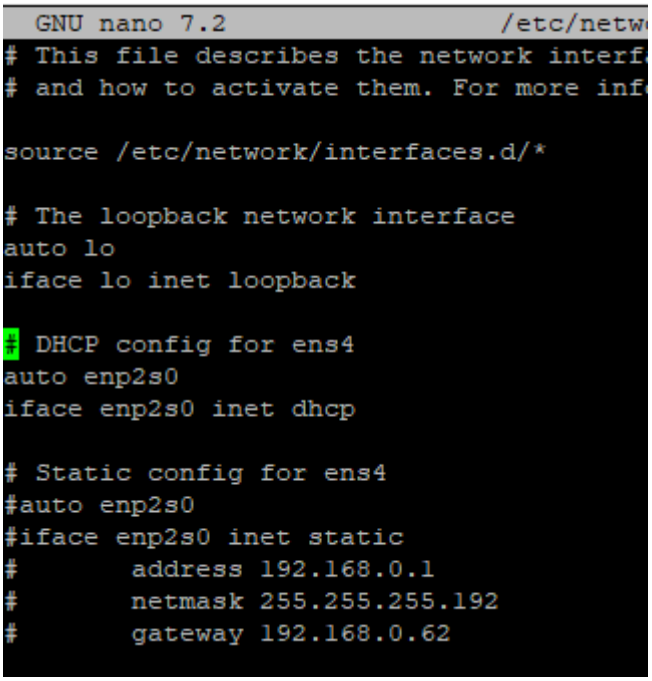
Pour tester que tout est opérationnel, vous pouvez réaliser un **ping 8.8.8.8** :



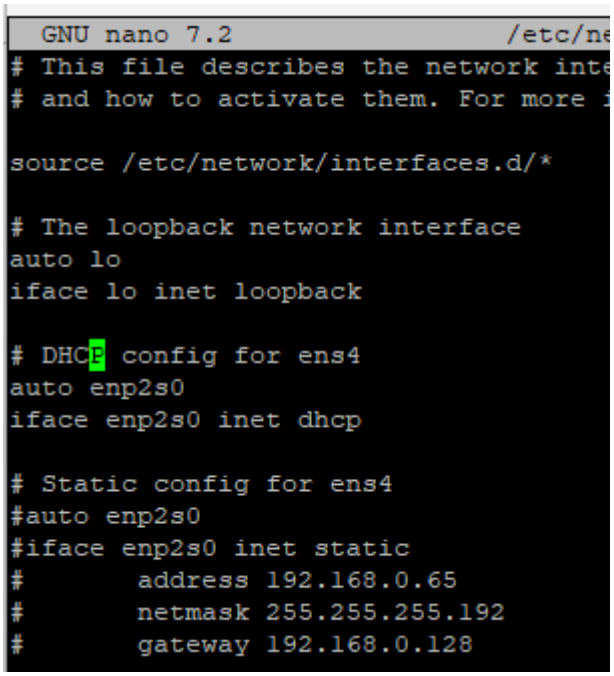
2 Serveur DHCP

Pour changer la configuration des deux machines Debian connectées au routeur principal afin qu'elles récupèrent leurs adresses IP en DHCP, il est nécessaire de reproduire la même procédure que pour le routeur. C'est à dire de modifier le fichier de configuration des interfaces avec la commande **sudo nano /etc/network/interfaces,** de commenter les lignes concernant l'adresse static des interfaces concernées, et de décommenter celles concernant le DHCP en indiquant l'interface voulue :

Machine 2 :



Machine 3 :



Il n'est pour l'instant pas nécessaire de redémarrer le service réseau car le routeur n'est pas encore capable de leur délivrer une nouvelle adresse ip.

Nous allons maintenant installer et configurer le serveur DHCP isc-dhcp-server sur le routeur Debian, afin qu'il délivre les adresses IP des deux sous-réseaux. Pour ce faire, il est nécessaire de suivre les étapes suivantes :

Avant de réaliser toute modification, il est important de modifier le contenu du fichier **/etc/apt/sources.list** permettant d'indiquer à Debian où chercher les paquets logiciels. Voici le contenu à mettre dans ce fichier :

deb http://deb.debian.org/debian/ bookworm main contrib non-free non-free-firmware
deb-src http://deb.debian.org/debian/ bookworm main contrib non-free non-free-firmware

deb http://security.debian.org/debian-security bookworm-security main contrib non-free non-free-firmware
deb-src http://security.debian.org/debian-security bookworm-security main contrib non-free non-free-firmware

deb http://deb.debian.org/debian/ bookworm-updates main contrib non-free non-free-firmware
deb-src http://deb.debian.org/debian/ bookworm-updates main contrib non-free non-free-firmware

Il est ensuite nécessaire de mettre à jour les paquets : **sudo apt update**

Procédons maintenant à la mise en place du serveur DHCP :

Tout d'abord, nous devons installer le package isc-dhcp-server : **sudo apt install isc-dhcp-server**

```
debian@debian:~$ sudo apt install isc-dhcp-server
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  isc-dhcp-common policycoreutils selinux-utils
Suggested packages:
  policykit-1 isc-dhcp-server-ldap ieee-data
The following NEW packages will be installed:
  isc-dhcp-common isc-dhcp-server policycoreutils selinux-utils
0 upgraded, 4 newly installed, 0 to remove and 106 not upgraded.
Need to get 1884 kB of archives.
After this operation, 7943 kB of additional disk space will be used.
Do you want to continue? [Y/n]
```

ATTENTION : Il est normal d'avoir une erreur lors du démarrage du service car le serveur n'est pas encore configuré :

```
apparmor_parser"
Job for isc-dhcp-server.service failed because the control process exited with error code.
See "systemctl status isc-dhcp-server.service" and "journalctl -xeu isc-dhcp-server.service" for details.
invoke-rc.d: initscript isc-dhcp-server, action "start" failed.
× isc-dhcp-server.service - LSB: DHCP server
     Loaded: loaded (/etc/init.d/isc-dhcp-server; generated)
     Active: failed (Result: exit-code) since Sat 2026-01-31 16:01:45 UTC; 290ms ago
       Docs: man:systemd-sysv-generator(8)
    Process: 1686 ExecStart=/etc/init.d/isc-dhcp-server start (code=exited, status=1/FAILURE)
        CPU: 845ms

Jan 31 16:01:43 debian dhcpd[1698]: bugs on either our web page at www.isc.org or in the README file
Jan 31 16:01:43 debian dhcpd[1698]: before submitting a bug.  These pages explain the proper
Jan 31 16:01:43 debian dhcpd[1698]: process and the information we find helpful for debugging.
Jan 31 16:01:43 debian dhcpd[1698]:
Jan 31 16:01:43 debian dhcpd[1698]: exiting.
Jan 31 16:01:45 debian isc-dhcp-server[1686]: Starting ISC DHCPv4 server: dhcpdcheck syslog for diagnostics. ... failed!
Jan 31 16:01:45 debian isc-dhcp-server[1686]:  failed!
Jan 31 16:01:45 debian systemd[1]: isc-dhcp-server.service: Control process exited, code=exited, status=1/FAILURE
Jan 31 16:01:45 debian systemd[1]: isc-dhcp-server.service: Failed with result 'exit-code'.
Jan 31 16:01:45 debian systemd[1]: Failed to start isc-dhcp-server.service - LSB: DHCP server.
Setting up isc-dhcp-common (4.4.3-P1-2) ...
Processing triggers for man-db (2.11.2-2) ...
```

Ensuite, le serveur DHCP doit uniquement répondre sur les interfaces reliées aux **réseaux internes**, et non sur l'interface connectée au NAT. Pour ce faire, nous allons modifier le fichier **/etc/default/isc-dhcp-server** en indiquant les interfaces du routeur reliées à nos deux machines au niveau de la ligne : **INTERFACESv4= "" .**

Dans notre cas, cela donnera :

```
# On what interfaces should the DHCP server (dhcpd) serve DHCP requests?
#       Separate multiple interfaces with spaces, e.g. "eth0 eth1".
INTERFACESv4="enp2s0 ens1"
INTERFACESv6=""
```

Enfin, nous allons configurer les pools DHCP pour les deux sous réseaux. Pour ce faire, nous devons modifier le fichier **/etc/dhcp/dhcpd.conf** et rentrer les différentes informations suivantes :

```
# Durée pour les baux DHCP en secondes (default 4 jours et 8 jours maximum)
default-lease-time 345600;
max-lease-time 691200;

# Serveur DHCP principal sur ce réseau local
authoritative;

# Sous-réseau 1 : 192.168.0.0/26
subnet 192.168.0.0 netmask 255.255.255.192 {
    range 192.168.0.10 192.168.0.50;
    option routers 192.168.0.62;
}

# Sous-réseau 2 : 192.168.0.64/26
subnet 192.168.0.64 netmask 255.255.255.192 {
    range 192.168.0.70 192.168.0.100;
    option routers 192.168.0.126;
}
```

Nous pouvons maintenant redémarrer le service DHCP avec la commande : **sudo systemctl restart isc-dhcp-server**

On peut ensuite vérifier son état avec : **sudo systemctl status isc-dhcp-server**
Le service doit apparaître comme **active (running)**.

```
debian@debian:~$ sudo systemctl status isc-dhcp-server
● isc-dhcp-server.service - LSB: DHCP server
     Loaded: loaded (/etc/init.d/isc-dhcp-server; generated)
     Active: active (running) since Sat 2026-01-31 16:21:44 UTC; 4s ago
       Docs: man:systemd-sysv-generator(8)
    Process: 1723 ExecStart=/etc/init.d/isc-dhcp-server start (code=exited, sta>
      Tasks: 1 (limit: 537)
     Memory: 4.6M
        CPU: 1.087s
     CGroup: /system.slice/isc-dhcp-server.service
             └─1735 /usr/sbin/dhcpd -4 -q -cf /etc/dhcp/dhcpd.conf enp2s0 ens1

Jan 31 16:21:41 debian systemd[1]: Starting isc-dhcp-server.service - LSB: DHCP>
Jan 31 16:21:42 debian isc-dhcp-server[1723]: Launching IPv4 server only.
Jan 31 16:21:42 debian dhcpd[1735]: Wrote 0 leases to leases file.
Jan 31 16:21:42 debian dhcpd[1735]: Server starting service.
Jan 31 16:21:44 debian isc-dhcp-server[1723]: Starting ISC DHCPv4 server: dhcpd.
Jan 31 16:21:44 debian systemd[1]: Started isc-dhcp-server.service - LSB: DHCP >
lines 1-17/17 (END)
```

Il est maintenant nécessaire d'exécuter les commandes **sudo ifdown enp2s0** et **sudo ifup enp2s0** permettent de désactiver puis réactiver l'interface réseau, ce qui force la prise en compte des nouvelles configurations IP ou DHCP sans redémarrer la machine.

Pour voir la nouvelle adresse ip associée à l'interface, vous pouvez entrer la commande **ip a**. Normalement, vous verrez une mention "scope global dynamic" s'afficher de la ligne affichant l'ip :

Machine 2 :

```
debian@debian:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group defaul
t qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: enp2s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP gr
oup default qlen 1000
    link/ether 0c:1b:a5:6e:00:00 brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.10/26 brd 192.168.0.63 scope global dynamic enp2s0
       valid_lft 345293sec preferred_lft 345293sec
    inet6 fe80::e1b:a5ff:fe6e:0/64 scope link
       valid_lft forever preferred_lft forever
debian@debian:~$
```

Machine 3 :

```
debian@debian:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group defaul
t qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: enp2s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP gr
oup default qlen 1000
    link/ether 0c:00:26:35:00:00 brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.70/26 brd 192.168.0.127 scope global dynamic enp2s0
       valid_lft 345598sec preferred_lft 345598sec
    inet6 fe80::e00:26ff:fe35:0/64 scope link
       valid_lft forever preferred_lft forever
debian@debian:~$
```

Pour activer le forwarding IPv4 si pas déjà fait, il suffit de décommenter la ligne : *net.ipv4.ip_forward=1* dans le fichier **/etc/sysctl.conf** :

```
# Uncomment the next line to enable packet forwarding for IPv4
net.ipv4.ip_forward=1
```

Appliquer ensuite les modifications avec la ligne : **sudo sysctl -p**

Pour finir, nous devons **configurer le NAT du routeur en mode masquerade.** Pour ce faire, voici le modèle de la commande à utiliser : *sudo iptables -t nat -A POSTROUTING -o <interface> -j MASQUERADE*
Dans notre cas, l'interface du routeur relié au NAT est enp2s2, donc la commande sera : **sudo iptables -t nat -A POSTROUTING -o enp2s2 -j MASQUERADE**

Pour plus d'info : https://eric-wurbel.pedaweb.univ-amu.fr/extranet/Enseignement/SAE203/tuto-nat.html

Pour vérifier que la règle a bien été prise en compte, vous pouvez taper la commande suivante : **sudo iptables -t nat -n -L -v**
Vous devriez voir afficher la règle dans la partie "Chain POSTROUTING"

```
Chain POSTROUTING (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

    1    84 MASQUERADE  0    --  *      enp2s2  0.0.0.0/0            0.0.0.0/0
```

Il est également **important de rendre les règles iptables persistantes**. Pour ce faire, il est nécessaire d'installer iptables-persistent avec la commande : **sudo apt install iptables-persistent**

```
debian@debian:~$ sudo apt install iptables-persistent
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  netfilter-persistent
The following NEW packages will be installed:
  iptables-persistent netfilter-persistent
```

Un message apparaitra nous indiquant si l'on souhaite sauvegarder les règles de iptables actuelles. Répondre à "*Yes*" pour les règles IPv4 et IPv6.

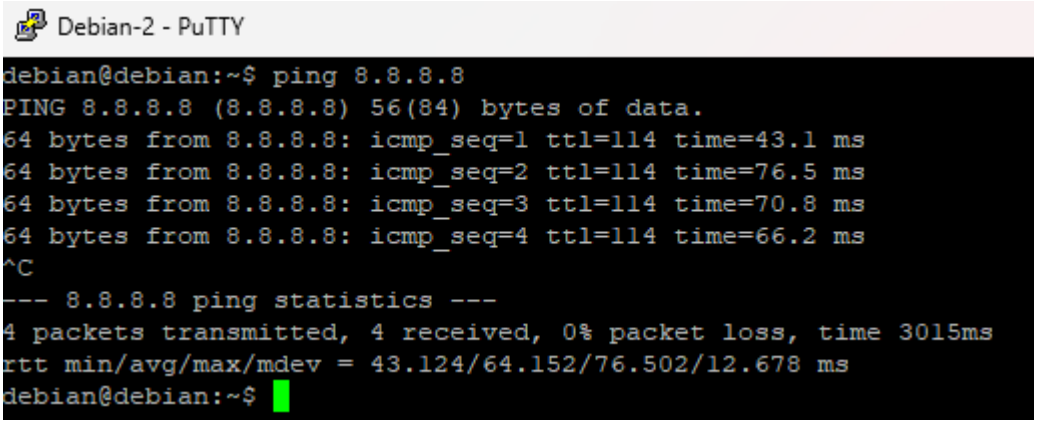
```
┤ Configuring iptables-persistent ├

Current iptables rules can be saved to the configuration file
/etc/iptables/rules.v4. These rules will then be loaded automatically
during system startup.

Rules are only saved automatically during package installation. See the
manual page of iptables-save(8) for instructions on keeping the rules
file up-to-date.

Save current IPv4 rules?

                <Yes>                    <No>
```

Dans le futur, il suffira d'entrer cette commande pour sauvegarder les règles : **sudo netfilter-persistent save**

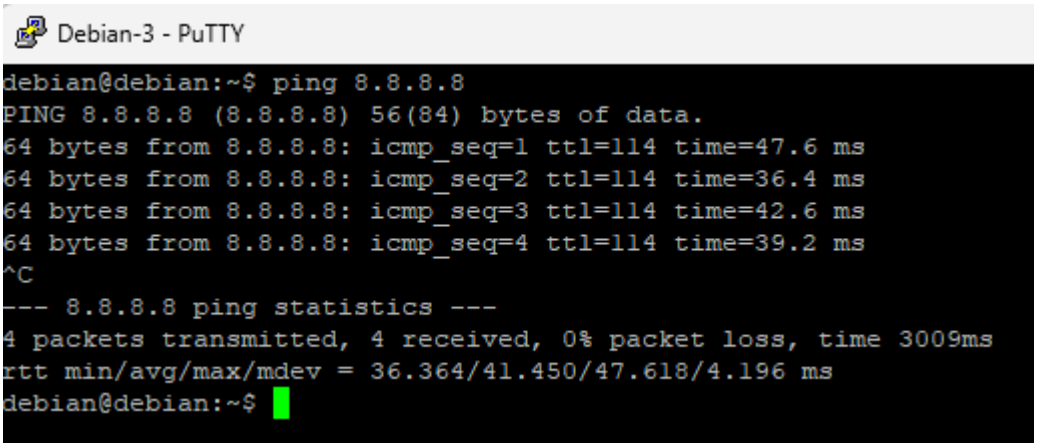
```
debian@debian:~$ sudo netfilter-persistent save
run-parts: executing /usr/share/netfilter-persistent/plugins.d/15-ip4tables save
run-parts: executing /usr/share/netfilter-persistent/plugins.d/25-ip6tables save
debian@debian:~$
```

Pour vérifier que tout est opérationnel, nous pouvons réaliser un **ping 8.8.8.8** depuis la machine 3 et 4. Si une réponse est reçue, c'est que le serveur DHCP est fonctionnel 🎉🎉

Machine 2 :



Machine 3 :



Nous pouvons maintenant nous intéresser à la communication IPv6. Nous allons utiliser SLAAC afin de permettre l'auto configuration des adresses.

Avant toute chose, il est nécessaire d'activer le forwarding IPv6. Pour ce faire, nous devons modifier le fichier /etc/sysctl.conf avec la commande : **sudo nano /etc/sysctl.conf**
Dans ce fichier, vérifier la présence ou ajouter la ligne suivante :
net.ipv6.conf.all.forwarding=1

```
# Uncomment the next line to enable packet forwarding for IPv6
#  Enabling this option disables Stateless Address Autoconfiguration
#  based on Router Advertisements for this host
net.ipv6.conf.all.forwarding=1
```

Appliquer ensuite les modifications avec : **sudo sysctl -p**

Nous allons maintenant installer radvd avec la commande : **sudo apt install radvd**

```
debian@debian:~$ sudo apt install radvd
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following NEW packages will be installed:
  radvd
```

Nous allons maintenant configurer radvd avec la commande : **sudo nano /etc/radvd.conf**
Voici le contenu à ajouter :

```
interface enp2s0 {
   AdvSendAdvert on;
   AdvManagedFlag off;
   AdvOtherConfigFlag off;

   prefix 2001:db8:1::/64 {
      AdvOnLink on;
      AdvAutonomous on;
   };
};

interface ens1 {
   AdvSendAdvert on;
   AdvManagedFlag off;
   AdvOtherConfigFlag off;

   prefix 2001:db8:2::/64 {
      AdvOnLink on;
      AdvAutonomous on;
   };
};
```

```
interface enp2s0 {
    AdvSendAdvert on;
    AdvManagedFlag off;
    AdvOtherConfigFlag off;

    prefix 2001:db8:1::/64 {
        AdvOnLink on;
        AdvAutonomous on;
    };
};

interface ens1 {
    AdvSendAdvert on;
    AdvManagedFlag off;
    AdvOtherConfigFlag off;

    prefix 2001:db8:2::/64 {
        AdvOnLink on;
        AdvAutonomous on;
    };
```

Explication :
Le fichier de configuration radvd permet au routeur d'envoyer des annonces IPv6 aux machines du réseau.
L'option **AdvSendAdvert on** active l'envoi de ces annonces, indispensables au fonctionnement de SLAAC.
Les options **AdvManagedFlag off** et **AdvOtherConfigFlag off** indiquent aux clients qu'ils ne doivent pas utiliser DHCPv6 et qu'ils peuvent s'auto-configurer uniquement à partir des annonces du routeur.
Le bloc **prefix** définit le préfixe IPv6 annoncé aux machines. Celui-ci est obligatoirement en **/64**, condition nécessaire pour que l'auto-configuration IPv6 fonctionne.
Enfin, **AdvOnLink on** précise que le préfixe correspond au lien local, et **AdvAutonomous on** autorise les machines à générer automatiquement leur adresse IPv6.

Il est ensuite nécessaire de démarrer et d'activer radvd avec :
sudo systemctl enable radvd
sudo systemctl restart radvd

Pour vérifier que le service est opérationnel, on entre : **sudo systemctl status radvd**

```
debian@debian:~$ sudo systemctl status radvd
● radvd.service - Router advertisement daemon for IPv6
     Loaded: loaded (/lib/systemd/system/radvd.service; enabled; preset: enable>
     Active: active (running) since Sun 2026-02-01 14:31:16 UTC; 7s ago
       Docs: man:radvd(8)
    Process: 730 ExecStartPre=/usr/sbin/radvd --logmethod stderr_clean --config>
    Process: 731 ExecStart=/usr/sbin/radvd --logmethod stderr_clean (code=exite>
   Main PID: 732 (radvd)
      Tasks: 2 (limit: 537)
     Memory: 472.0K
        CPU: 277ms
     CGroup: /system.slice/radvd.service
             ├─732 /usr/sbin/radvd --logmethod stderr_clean
             └─733 /usr/sbin/radvd --logmethod stderr_clean

Feb 01 14:31:15 debian systemd[1]: Starting radvd.service - Router advertisemen>
Feb 01 14:31:15 debian radvd[730]: config file, /etc/radvd.conf, syntax ok
Feb 01 14:31:16 debian radvd[731]: version 2.19 started
Feb 01 14:31:16 debian systemd[1]: Started radvd.service - Router advertisement>
```

Nous allons maintenant passer à la configuration côté client. Pour ce faire, nous allons modifier le fichier /etc/network/interfaces avec la commande :
sudo nano /etc/network/interfaces
et nous allons ajouter cette ligne : **iface enp2s0 inet6 auto**

```
# DHCP config for ens4
auto enp2s0
iface enp2s0 inet dhcp
iface enp2s0 inet6 auto
```

Il est maintenant nécessaire d'exécuter les commandes **sudo ifdown enp2s0** et **sudo ifup enp2s0** permettent de désactiver puis réactiver l'interface réseau, ce qui force la prise en compte des nouvelles configurations IP ou DHCP sans redémarrer la machine.

Pour vérifier que tout fonctionne correctement, nous devrions voir une adresse IPv6 en 2001:db8:1::/64 sur la machine 2 et une adresse IPv6 en 2001:db8:2::/64 sur la machine 3, avec la mention "scope global dynamic" en fin de ligne.
Pour vérifier, faire : **ip a**

Machine 2 :

```
2: enp2s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 0c:1b:a5:6e:00:00 brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.10/26 brd 192.168.0.63 scope global dynamic enp2s0
       valid_lft 345310sec preferred_lft 345310sec
    inet6 2001:db8:1:0:e1b:a5ff:fe6e:0/64 scope global dynamic mngtmpaddr
       valid_lft 86364sec preferred_lft 14364sec
    inet6 fe80::e1b:a5ff:fe6e:0/64 scope link
       valid_lft forever preferred_lft forever
```

Machine 3 :

```
2: enp2s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 0c:00:26:35:00:00 brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.70/26 brd 192.168.0.127 scope global dynamic enp2s0
       valid_lft 345592sec preferred_lft 345592sec
    inet6 2001:db8:2:0:e00:26ff:fe35:0/64 scope global dynamic mngtmpaddr
       valid_lft 86399sec preferred_lft 14399sec
    inet6 fe80::e00:26ff:fe35:0/64 scope link
       valid_lft forever preferred_lft forever
```

De plus, une route par défaut passant par l'interface du routeur devrait apparaître avec la commande : **ip -6 route**

Machine 2 :

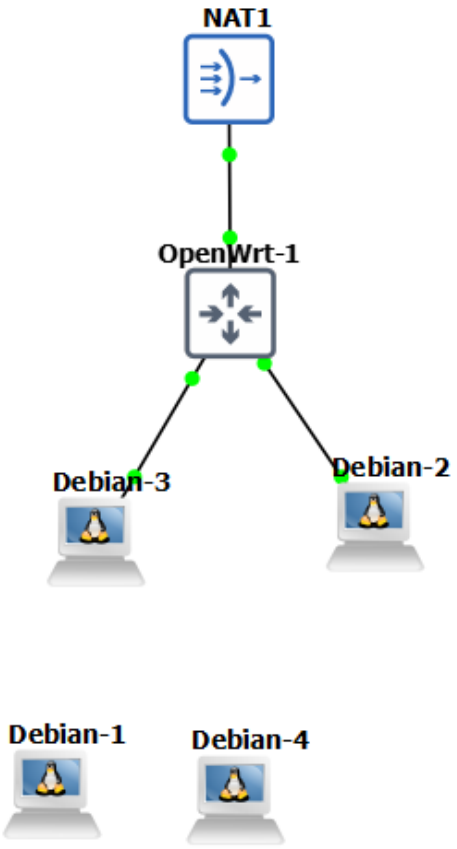
```
debian@debian:~$ ip -6 route
2001:db8:1::/64 dev enp2s0 proto kernel metric 256 expires 86395sec pref medium
fe80::/64 dev enp2s0 proto kernel metric 256 pref medium
default via fe80::e52:cfff:fee2:0 dev enp2s0 proto ra metric 1024 expires 1795sec hoplimit 64 pref medium
```

Machine 3 :

```
debian@debian:~$ ip -6 route
2001:db8:2::/64 dev enp2s0 proto kernel metric 256 expires 86278sec pref medium
fe80::/64 dev enp2s0 proto kernel metric 256 pref medium
default via fe80::e52:cfff:fee2:1 dev enp2s0 proto ra metric 1024 expires 1678sec hoplimit 64 pref medium
```

3 Ça, c'est un routeur ?

1. Télécharger l'appliance OpenWRT sur : https://gns3.com/marketplace/appliances/openwrt-2
2. L'installer sur le server puis remplacer la machine debian 1 tel que ci dessous :



Attention à suivre la même configuration des interfaces que nous pour faciliter la suite du tp :
(OpenWrt) **eth0=== nat0** (NAT)
(OpenWrt) **eth2=== ens4** (Machine 2)
(OpenWrt) **eth3=== ens4** (Machine 3)

3. Ouvrir la machine pour les configuration
4. lancer : **vi /etc/config/network**
5. configurer l'interface eth0 sur le WAN (NAT) et les autres ports sur le LAN comme ci-dessous :

```
config interface 'loopback'
        option device 'lo'
        option proto 'static'
        option ipaddr '127.0.0.1'
        option netmask '255.0.0.0'

config globals 'globals'
        option ula_prefix 'fdbd:3567:6e9f::/48'

config device
        option name 'br-lan'
        option type 'bridge'
        list ports 'eth1'
        list ports 'eth2'
        list ports 'eth3'

config interface 'lan1'
        option device 'eth2'
        option proto 'static'
        option ipaddr '192.168.0.1'
        option netmask '255.255.255.192'
        option ip6assign '64'

config interface 'lan2'
        option device 'eth3'
        option proto 'static'
        option ipaddr '192.168.0.65'
        option netmask '255.255.255.192'
        option ip6assign '64'

config interface 'wan'
        option device 'eth0'
        option proto 'dhcp'

config interface 'wan6'
        option device 'eth0'
        option proto 'dhcpv6'
```

Les contraintes de temps ne nous ont pas permis de rendre l'IPv6 fonctionnel alors nous nous contentons de l'IPv4.

Tips vi :

- touche "i" pour entré en mode d'édition
- utiliser les flèches pour se déplacer
- touche "échap" pour sortir du mode d'édition
- commande :q! pour quitter
- commande :wq pour sauvegarder et quitter

Nous devons maintenant configurer le serveur DHCP. Pour ce faire, modifier le fichier avec cette commande : **vi /etc/config/dhcp**
Nous devons ajouter une configuration pour nos 2 Lan :
(ATTENTION au nom des lan, ils doivent être identique à ceux mis dans le fichier network)

On débute la plage d'ip du Lan 1 à .10
avec une limite de 41 adresses
(pour couvrir jusqu'à .50)

de la plage à .6 (soit 65 + 5 = 70)
avec une limite de 36 adresses
(pour couvrir jusqu'à .100)

```
config dhcp 'lan1'
        option interface 'lan1'
        option start '10'
        option limit '41'
        option leasetime '12h'
        option dhcpv4 'server'

config dhcp 'lan2'
        option interface 'lan2'
        option start '6'
        option limit '36'
        option leasetime '12h'
        option dhcpv4 'server'
```

Nous devons aussi configurer le pare-feu, en autorisant les requêtes venant des 2 Lan, permettant aux machines d'obtenir une adresse IP. Pour ce faire, nous devons modifier un fichier à l'aide de la commande : **vi /etc/config/firewall**

Nous devons ajouter dans la section dédiée au lan nos deux nouveaux lan :

```
config zone
        option name             lan
        list   network          'lan'
        list   network          'lan1'
        list   network          'lan2'
        option input            ACCEPT
        option output           ACCEPT
        option forward          ACCEPT
```

Nous devons entrer les commandes suivantes pour redémarrer les services nécessaires pour prendre en compte les modifications apportées :
/etc/init.d/network restart
/etc/init.d/dnsmasq restart
/etc/init.d/odhcpd restart

On peut tester cette nouvelle installation avec différents ping :

Sur OpenWrt :

```
root@OpenWrt:~# ping 192.168.10.1
PING 192.168.10.1 (192.168.10.1): 56 data bytes
64 bytes from 192.168.10.1: seq=0 ttl=64 time=1.647 ms
64 bytes from 192.168.10.1: seq=1 ttl=64 time=0.680 ms
^C
--- 192.168.10.1 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max = 0.680/1.163/1.647 ms
root@OpenWrt:~# ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8): 56 data bytes
64 bytes from 8.8.8.8: seq=0 ttl=112 time=55.258 ms
64 bytes from 8.8.8.8: seq=1 ttl=112 time=44.714 ms
64 bytes from 8.8.8.8: seq=2 ttl=112 time=57.598 ms
^C
--- 8.8.8.8 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 44.714/52.523/57.598 ms
```

```
root@OpenWrt:~# ping google.com
PING google.com (142.250.179.110): 56 data bytes
64 bytes from 142.250.179.110: seq=0 ttl=113 time=47.839 ms
64 bytes from 142.250.179.110: seq=1 ttl=113 time=48.905 ms
64 bytes from 142.250.179.110: seq=2 ttl=113 time=38.051 ms
^C
--- google.com ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 38.051/44.931/48.905 ms
root@OpenWrt:~#
```

Sur la machine 2 :

```
debian@debian:~$ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=111 time=43.3 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=111 time=57.0 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=111 time=55.1 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=111 time=51.3 ms
64 bytes from 8.8.8.8: icmp_seq=5 ttl=111 time=97.7 ms
^C
--- 8.8.8.8 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4011ms
rtt min/avg/max/mdev = 43.290/60.891/97.686/18.992 ms
debian@debian:~$
```

```
debian@debian:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: enp2s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 0c:1b:a5:6e:00:00 brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.10/26 brd 192.168.0.63 scope global dynamic enp2s0
       valid_lft 42591sec preferred_lft 42591sec
    inet6 fe80::e1b:a5ff:fe6e:0/64 scope link
       valid_lft forever preferred_lft forever
debian@debian:~$
```

Sur la machine 3 :

```
debian@debian:~$ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=111 time=53.9 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=111 time=40.7 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=111 time=39.1 ms

--- 8.8.8.8 ping statistics ---
^C3 packets transmitted, 3 received, 0% packet loss, time 2005ms
rtt min/avg/max/mdev = 39.121/44.580/53.879/6.608 ms
debian@debian:~$ ping 192.168.10.212
PING 192.168.10.212 (192.168.10.212) 56(84) bytes of data.
64 bytes from 192.168.10.212: icmp_seq=1 ttl=64 time=13.5 ms
64 bytes from 192.168.10.212: icmp_seq=2 ttl=64 time=4.98 ms
64 bytes from 192.168.10.212: icmp_seq=3 ttl=64 time=3.49 ms
^C
--- 192.168.10.212 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2011ms
rtt min/avg/max/mdev = 3.485/7.331/13.524/4.421 ms
debian@debian:~$
```

```
debian@debian:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: enp2s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 0c:00:26:35:00:00 brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.70/26 brd 192.168.0.127 scope global dynamic enp2s0
       valid_lft 43174sec preferred_lft 43174sec
    inet6 fe80::e00:26ff:fe35:0/64 scope link
       valid_lft forever preferred_lft forever
debian@debian:~$
```

Pour mettre en place le SLAAC IPv6, il suffit de modifier un fichier avec la commande : **vi /etc/config/dhcp** et d'ajouter les lignes suivantes dans la configuration de lan1 et lan2 :

```
option ra 'server'
option ra_default '1'
option dhcpv6 'server'
option ndp 'server'
```

Ce qui donnera :

```
config dhcp 'lan1'
        option interface 'lan1'
        option start '10'
        option limit '41'
        option leasetime '12h'
        option dhcpv4 'server'
        option ra 'server'
        option ra_default '1'
        option dhcpv6 'server'
        option ndp 'server'

config dhcp 'lan2'
        option interface 'lan2'
        option start '6'
        option limit '36'
        option leasetime '12h'
        option dhcpv4 'server'
        option ra 'server'
        option ra_default '1'
        option dhcpv6 'server'
        option ndp 'server'
```

Il faut ensuite redémarrer le service avec : **/etc/init.d/odhcpd restart .**

Pour prendre en compte les modifications sur les machines clientes, entrez la commande : **sudo systemctl restart networking**

Il suffit ensuite d'entrer : **ip a** pour vérifier si les modifications ont bien été prises en compte, avec l'apparition d'une adresse ipv6 avec la mention "scope global dynamic" en fin de ligne.

Machine 2 :

```
2: enp2s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 0c:1b:a5:6e:00:00 brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.10/26 brd 192.168.0.63 scope global dynamic enp2s0
       valid_lft 43034sec preferred_lft 43034sec
    inet6 fdbd:3567:6e9f:0:e1b:a5ff:fe6e:0/64 scope global dynamic mngtmpaddr
       valid_lft forever preferred_lft forever
    inet6 fe80::e1b:a5ff:fe6e:0/64 scope link
       valid_lft forever preferred_lft forever
debian@debian:~$
```

Machine 3 :

```
2: enp2s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 0c:00:26:35:00:00 brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.70/26 brd 192.168.0.127 scope global dynamic enp2s0
       valid_lft 41872sec preferred_lft 41872sec
    inet6 fdbd:3567:6e9f:1:e00:26ff:fe35:0/64 scope global dynamic mngtmpaddr
       valid_lft forever preferred_lft forever
    inet6 fe80::e00:26ff:fe35:0/64 scope link
       valid_lft forever preferred_lft forever
debian@debian:~$
```

Une route par défaut a normalement également été ajouté. Pour vérifier : **ip -6 route**

Machine 2 :

```
debian@debian:~$ ip -6 route
fdbd:3567:6e9f::/64 dev enp2s0 proto kernel metric 256 pref medium
fe80::/64 dev enp2s0 proto kernel metric 256 pref medium
default via fe80::e0a:b8ff:fe67:2 dev enp2s0 proto ra metric 1024 expires 1762sec hoplimit 64 pref medium
debian@debian:~$
```

Machine 3 :

```
debian@debian:~$ ip -6 route
fdbd:3567:6e9f:1::/64 dev enp2s0 proto kernel metric 256 pref medium
fe80::/64 dev enp2s0 proto kernel metric 256 pref medium
default via fe80::e0a:b8ff:fe67:3 dev enp2s0 proto ra metric 1024 expires 1479sec hoplimit 64 pref medium
debian@debian:~$
```

Après quelques recherches, nous sommes arrivés à cette conclusion :

Bien que l'adressage dynamique par SLAAC soit opérationnel au sein des réseaux locaux (LAN), l'accès à l'Internet IPv6 n'est pas possible dans cette maquette. Cela s'explique par le fait que le nœud NAT de GNS3 ne fournit qu'une connectivité IPv4