

# NAT et Filtrage sous linux

Adressage: PC2: 10.203.0.116 Routeur, réseau de la salle: 10.203.0.194 Routeur,  
réseau privé: 192.168.1.254 Serveur Web: PC1:

## 1. Le NAT

### 1.1 SNAT

1. Le SNAT doit être sur le hook PostRouting, ceci est une action sur la sorti d'interface.
2. Voici les commandes

```
nft add table nat
nft add chain nat natexterne {type nat hook postrouting priority 0 \;}
nft add rule nat natexterne ip saddr 192.168.1.0/24 oif ens4 snat 10.203.0.194
```

3. On peut effectuer un ping, voici le résultat:

The screenshot displays a network simulation environment. On the left, a network diagram shows a router (R401-TP4-NAT) connected to a switch (Switch1) and a web server (WEB). The router's configuration is shown in the console window, including the NAT rule setup. The right pane shows two Wireshark captures. The top capture shows ICMP echo requests and replies between 10.203.0.116 and 10.203.0.194. The bottom capture shows ICMP echo requests and replies between 10.203.0.194 and 10.203.0.116. The bottom pane also shows a console window with the command 'nft list table nat' and its output.

### 1.2 MASQUERADE

4. Voici les commandes à mettre, et la preuve avec le ping:

```
nft -n -a list table nat #permet de lister la table de nat avec le numéro de handle, c'est mieux que nft list ruleset.
nft delete rule nat natexterne handle 11 #handle est la position de la règle.
nft add rule nat natexterne masquerade
```

The screenshot displays a GNS3 network simulation. The topology includes a Cloud1 connected to a ROUTER (ens4), which is connected to a Switch1 (ens5). Switch1 is connected to two PCs: UbuntuDesktopGuest16.04.6-1 (eth0) and a WEB server (ens4). The router's configuration shows a default route to the cloud and a specific route for the web server. The terminal window shows a successful ping from the PC to the web server. The Wireshark captures show ICMP echo requests and replies between the router and the web server.

## 1.3 DNAT

5. La règle de DNAT se place sur le hook du prerouting.
6. Voici les règles utilisées:

```
nft nft add chain nat dstnat {type nat hook prerouting priority 0 \;}
nft add rule ip nat dstnat iif ens4 tcp dport 80 dnat 192.168.1.80:80
```

7. Voici la page du serveur web depuis le PC2:

**Thibault GARCIA et Mathys DOMERGUE**

**debian**

**C'EST POUR LE TP4 DE R401 !**

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Debian systems. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at `/var/www/html/index.html`) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

**Configuration Overview**

Debian's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Debian tools. The configuration system is **fully documented in `/usr/share/doc/apache2/README.Debian.gz`**. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the **manual** if the `apache2-doc` package was installed on this server.

The configuration layout for an Apache2 web server installation on Debian systems is as follows:

```
/etc/apache2/
|-- apache2.conf
    |-- ports.conf
    |-- mods-enabled
        |-- *.load
        |-- *.conf
    |-- conf-enabled
        |-- *.conf
    |-- sites-enabled
        |-- *.conf
```

- `apache2.conf` is the main configuration file. It puts the pieces together by including all remaining configuration files when starting up the web server.
- `ports.conf` is always included from the main configuration file. It is used to determine the listening ports for incoming connections, and this file can be customized anytime.
- Configuration files in the `mods-enabled/`, `conf-enabled/` and `sites-enabled/` directories contain particular configuration snippets which manage modules, global configuration fragments, or virtual host configurations, respectively.
- They are activated by symlinking available configuration files from their respective `*-available/` counterparts. These should be managed by using our helpers `a2enmod`, `a2dismod`, `a2ensite`, `a2dissite`, and `a2enconf`, `a2disconf`. See their respective man pages for detailed information.
- The binary is called `apache2`. Due to the use of environment variables, in the default configuration, `apache2` needs to be started/stopped with `/etc/init.d/apache2` or `apache2ctl`. **Calling `/usr/bin/apache2` directly will not work with the default configuration.**

**Network Diagram:**

```
graph TD
    Cloud1((Cloud1)) --- ens4[ROUTER ens4]
    ens4 --- ens5[ROUTER ens5]
    ens5 --- 254[254]
    254 --- Switch1[Switch1]
    Switch1 --- eth0[eth0]
    eth0 --- Ubuntu[UbuntuDesktopGuest16.04.6-1]
    Switch1 --- ens4[Switch1 ens4]
    ens4 --- WEB[WEB]
```

**Wireshark Packet Capture 1 (ROUTER ens4):**

No.	Time	Source	Destination	Protocol	Length	Info
258	0.759493	10.203.0.194	10.203.0.116	TCP	1514	80 → 53108 [ACK] Seq=2897 Ack=77
259	0.759569	10.203.0.194	10.203.0.116	TCP	1514	80 → 53108 [ACK] Seq=4345 Ack=77
260	0.759637	10.203.0.194	10.203.0.116	TCP	1514	80 → 53108 [FIN, ACK] Seq=5793
261	0.759663	10.203.0.194	10.203.0.116	TCP	1514	80 → 53108 [ACK] Seq=7241 Ack=77
262	0.759719	10.203.0.194	10.203.0.116	TCP	1514	80 → 53108 [ACK] Seq=8689 Ack=77
263	0.759750	10.203.0.194	10.203.0.116	HTTP	911	HTTP/1.1 200 OK (text/html)
264	0.760970	10.203.0.116	10.203.0.194	TCP	66	53108 → 80 [ACK] Seq=77 Ack=7241
265	0.760995	10.203.0.116	10.203.0.194	TCP	66	53108 → 80 [ACK] Seq=77 Ack=1098
266	0.761287	10.203.0.116	10.203.0.194	TCP	66	53108 → 80 [FIN, ACK] Seq=77 Ack=1098
267	0.761545	10.203.0.194	10.203.0.116	TCP	66	80 → 53108 [FIN, ACK] Seq=1098

**Wireshark Packet Capture 2 (Switch1 ens4):**

No.	Time	Source	Destination	Protocol	Length	Info
8	0.003770	192.168.1.80	192.168.1.254	TCP	1514	80 → 53108 [ACK] Seq=1449 Ack=77
9	0.003871	192.168.1.80	192.168.1.254	TCP	1514	80 → 53108 [ACK] Seq=4345 Ack=77
10	0.004016	192.168.1.80	192.168.1.254	TCP	1514	80 → 53108 [FIN, ACK] Seq=5793
11	0.004083	192.168.1.80	192.168.1.254	TCP	1514	80 → 53108 [ACK] Seq=7241 Ack=77
12	0.004168	192.168.1.80	192.168.1.254	TCP	1514	80 → 53108 [ACK] Seq=8689 Ack=77
13	0.004210	192.168.1.80	192.168.1.254	HTTP	911	HTTP/1.1 200 OK (text/html)
14	0.005295	192.168.1.254	192.168.1.80	TCP	66	53108 → 80 [ACK] Seq=77 Ack=7241
15	0.005335	192.168.1.254	192.168.1.80	TCP	66	53108 → 80 [ACK] Seq=77 Ack=1098
16	0.006032	192.168.1.254	192.168.1.80	TCP	66	53108 → 80 [FIN, ACK] Seq=77 Ack=1098
17	0.006241	192.168.1.80	192.168.1.254	TCP	66	80 → 53108 [FIN, ACK] Seq=1098

## 2. Filtrage

### 2.1 Sans état: Stateless

8. Voici la règle utilisée:

```
nft add rule filter refus ip daddr 141.94.78.218 drop
```

La limite des cette méthode est les grosses structures comme Youtube ou Facebook, avec beaucoup de serveurs.

9. Voici le filtre

```
nft add rule filter refus iif ens5 tcp dport 22 reject
```

10. Par default, il vaut mieux de mettre une table en drop puis d'ajouter les règles une par une.

11. Voici les commandes passés:

```
nft add rule filter OUTPUT icmp type echo-request reject
nft add rule filter OUTPUT icmp type echo-request drop
```

12. Voici les commandes:

```
nft add rule filter FORWARD icmp type echo-request accept // Pour ping
l'Internet
nft add rule filter FORWARD icmp type echo-reply accept // Pour ping
l'Internet
nft add rule filter INPUT icmp type echo-request accept // Pour ping le
Router
nft add rule filter INPUT icmp type echo-reply accept // Pour ping le
Router
nft add rule filter INPUT tcp dport 22 ip saddr 192.168.1.0/24 accept // accept
le SSH du LAN
nft add rule filter FORWARD tcp dport 80 accept // accept les acces WEB http
nft add rule filter FORWARD tcp dport 443 accept // accept les acces WEB https
(On va plus loin que la question)
drop
```

## 2.2 Avec état: Satefull

13. Voici l'option à utiliser :

```
ct state
```

14. Voici les commandes :

```
nft add rule filter FORWARD tcp dport 22 daddr 192.168.1.0/24 ct state
established accept // Autorise les réponses ssh
( tcp port 22 )

nft add rule filter FORWARD tcp dport 22 saddr 192.168.1.0/24 ct state new
accept // Pour qu'il y est established, il faut
un new (donc que le LAN puisse initier une connexion)
```

15.

```
nft add rule filter FORWARD tcp dport 22 daddr 192.168.1.0/24 ct state
established accept // Autorise les réponses ssh
( tcp port 22 )

nft add rule filter FORWARD tcp dport 22 saddr 192.168.1.0/24 ct state new
accept // Pour qu'il y est established, il faut
un new (donc que le LAN puisse initier une connexion)
```

16.

```
nft add rule filter FORWARD counter tcp dport 80 ct state new accept
```

17.

```
nft list counters
```