

UNIVERSITE D'ABOMEY CALAVI

INSTITUT DE FORMATION ET DE RECHERCHE EN
INFOMATIQUE (IFRI)

DISCIPLINE : SECURITE MATERIELLE ET LOGICIELLE
ANNEE : 2021-2022

**RAPPORT TP DE MISE EN PLACE D'UNE
ARCHITECTURE RESEAU AVEC LE
SIMULATEUR NETKIT**

APPRENANT :

PROFESSEUR :

SODOKIN Yao Marius

Mr AHOUANDJINOUE Arnaud

Partie 1 : SECURITE MATERIELLE

1- INSTALLATION DE L'OUTIL DE SIMULATION NETKIT

Dans un premier temps, j'ai fait l'installation de netkit qui s'est déroulé en deux étapes: j'ai créé un dossier nommé Travaux_Pratiques_Netkit avec la commande `mkdir Travaux_Pratiques_Netkit`

```
(sodyam@sodyam)-[~]  
$ mkdir Travaux_Pratiques_Netkit
```

❖ Après avoir copié les fichiers netkit dans le dossier, j'ai accédé au dossier avec la commande `cd Travaux_Pratiques_Netkit`

```
-rwxrwxrwx 1 sodyam sodyam 151591 13 août 2015 netkit-2.8.tar.bz2  
-rwxrwxrwx 1 sodyam sodyam 287214735 13 août 2015 netkit-filesystem-i386-F5.2.tar.bz2  
-rwxrwxrwx 1 sodyam sodyam 2872052 13 août 2015 netkit-kernel-i386-K2.8.tar.bz2  
  
(sodyam@sodyam)-[~/Bureau/Travaux_Pratique_Netkit]  
$
```

A- DECOMPRESSION DES FICHIERS

Ici, j'ai tapé les commandes ci après :

tar xjf netkit - 2.8.tar.bz2

tar xjf netkit- filesystem-filesystem - F5.2.tar.bz2

tar xjf netkit - kernel-kernel - K2.8.tar.bz

```
(sodyam@sodyam)-[~/Bureau/Travaux_Pratique_Netkit]  
$ tar xjf netkit-2.8.tar.bz2  
  
(sodyam@sodyam)-[~/Bureau/Travaux_Pratique_Netkit]  
$ tar xjf netkit-filesystem-i386-F5.2.tar.bz2  
  
(sodyam@sodyam)-[~/Bureau/Travaux_Pratique_Netkit]  
$ tar xjf netkit-kernel-i386-K2.8.tar.bz2  
  
(sodyam@sodyam)-[~/Bureau/Travaux_Pratique_Netkit]  
$
```

B- AJOUT DES FICHIERS AUX VARIABLES D'ENVIRONNEMENT SYSTEME

Après les décompression, un dossier nommé netkit est généré et qui comporte les fichiers d'installation du netkit. J'ai accédé à son contenu avec la commande `cd netkit`.

J'ai ensuite ajouté aux variables d'environnement système.

Ceci en les exportant avec les commandes suivantes: `cd netkit`

`export NETKIT_HOME=~/.Bureau/Travaux_Pratique_Netkit/netkit`

`export PATH=$PATH:$NETKIT_HOME/bin`

`export MANPATH=:$NETKIT_HOME/man`

Le premier export consiste à ajouter le chemin du fichier exécutable de netkit

```
(sodyam@sodyam)-[~/Bureau/Travaux_Pratique_Netkit/netkit]
$ export NETKIT_HOME=~/.Bureau/Travaux_Pratique_Netkit/netkit

(sodyam@sodyam)-[~/Bureau/Travaux_Pratique_Netkit/netkit]
$ export PATH=$PATH:$NETKIT_HOME/bin

(sodyam@sodyam)-[~/Bureau/Travaux_Pratique_Netkit/netkit]
$ export MANPATH=:$NETKIT_HOME/man

(sodyam@sodyam)-[~/Bureau/Travaux_Pratique_Netkit/netkit]
$
```

Le premier export consiste à ajouter le chemin du fichier exécutable de netkit.

B- TEST DE L'INSTALLATION

Pour s'assurer que netkit a été bien installé, j'ai tapé la commande `"check_configuration.sh"`

```
(sodyam@sodyam)-[~/Bureau/Travaux_Pratique_Netkit/netkit]
$ ./check_configuration.sh
> Checking path correctness... passed.
> Checking environment... passed.
> Checking for availability of man pages... passed.
> Checking for proper directories in the PATH... passed.
> Checking for availability of auxiliary tools:
    awk           : ok
    basename      : ok
    date          : ok
    dirname       : ok
    find          : ok
    getopt        : ok
    grep          : ok
    head          : ok
    id            : ok
    kill          : ok
    ls            : ok
    lsof          : ok
    ps            : ok
    readlink      : ok
    wc            : ok
    port-helper   : ok
    tuncctl       : ok
    uml_mconsole  : ok
    uml_switch    : ok
passed.
> Checking for availability of terminal emulator applications:
    xterm         : found
    konsole       : found
    gnome-terminal : found
passed.
> Checking filesystem type... passed.
> Checking whether 32-bit executables can run... passed.
[ READY ] Congratulations! Your Netkit setup is now complete!
          Enjoy Netkit!

(sodyam@sodyam)-[~/Bureau/Travaux_Pratique_Netkit/netkit]
$
```

2- CREATION DES DIFFERENTS RESSOURCES DE L'ARCHITECTURE ET CONFIGURATION DES INTERFACES

Il s'agit maintenant de commencer la création des machines virtuelles. Chaque machine virtuelle possède quatre (04) caractéristiques à savoir : son nom, son (ses) interface(s), le (s) domaine (s) de collision au (x) (s)quel(s) les interfaces sont connectés et la mémoire qui lui est allouée. Pour chaque machine créée, j'ai utilisé la commande ci après :

vstart — [interface]=[Domaine de collision] - M [mémoire allouée] [nom]

❖ CREATION DES RESSOURCES DU RESEAU LOCAL

Dans ce réseau, les machines sont tous connectés au domaines de collision nommé A. Etant donné qu'elles possèdent un seul interface chacun, j'ai juste créé pour chacun eth0. Pour ce qui concerne la mémoire allouée, j'ai tapé **vstart --eth0=A -M 64 pc1**.

Ici, j'ai créé une machine nommé pc1 ayant son interface eth0 (avec l'option --eth0) connecté au domaine de collision A (=A) et auquel allouée une mémoire de 64 MB (avec l'option -M) Après avoir lancé, une fenêtre est représentant le terminal de configuration.

```
(sodyam@sodyam)~[~/Bureau/Travaux_Pratique_Netkit/netkit]
$ vstart --eth0=A -M 64 pc1

===== Starting virtual machine "pc1" =====
Kernel:      /home/sodyam/Bureau/Travaux_Pratique_Netkit/netkit/kernel/netkit-kernel
Modules:     /home/sodyam/Bureau/Travaux_Pratique_Netkit/netkit/kernel/modules
Memory:      64 MB
Model fs:    /home/sodyam/Bureau/Travaux_Pratique_Netkit/netkit/fs/netkit-fs
Filesystem:  /home/sodyam/Bureau/Travaux_Pratique_Netkit/netkit/pc1.disk (new)
Interfaces:  eth0 @ A (/home/sodyam/.netkit/hubs/vhub_sodyam_A.cnct)
Hostfs at:   /home/sodyam

Running ==> /home/sodyam/Bureau/Travaux_Pratique_Netkit/netkit/bin/uml_switch -hub -unix /home/sodyam/.netkit/hubs/vhub_sodyam_A.cnct </dev/null 2>&1
Running ==> xterm -e /home/sodyam/Bureau/Travaux_Pratique_Netkit/netkit/kernel/netkit-kernel modules=/home/sodyam/Bureau/Travaux_Pratique_Netkit/netkit/kernel/modules name=pc1 title=pc1 umid=pc1 mem=68M ubd0=/home/sodyam/Bureau/Travaux_Pratique_Netkit/netkit/pc1.disk,/home/sodyam/Bureau/Travaux_Pratique_Netkit/netkit/fs/netkit-fs root=98:1 uml_dir=/home/sodyam/.netkit/m console eth0=daemon,,/home/sodyam/.netkit/hubs/vhub_sodyam_A.cnct hosthome=/home/sodyam quiet con0=fd:0,fd:1 con1=null SELINUX_INIT=0
```

Ici, j'ai

créé une machine nommé pc1 ayant son interface eth0 (avec l'option --eth0) connecté au domaine de collision A (=A) et auquel alloue une mémoire de 64 MB

```
pc1_internet

Starting system log daemon....
Starting kernel log daemon....

— Starting Netkit phase 2 init script —

=====
Lab directory (host):
Version: <none>
Author: <none>
Email: <none>
Web: <none>
Description:
<none>
=====

— Netkit phase 2 initialization terminated —

pc1_internet login: root (automatic login)
Last login: Sun Jan 23 20:51:56 UTC 2022 on tty1
pc1_internet:~#
```

(avec l'option -M) Après avoir lancé, une fenêtre est représentant le terminal de configuration.

Pour créer les autres machines du réseau local nous, j'ai utilisé la même commande sauf que le nom a changé.

J'ai donc crée selon l'architecture les machines pc2,pc3, printer et file_serv,pc2
Autrement dit, pour créer pc2 par exemple, nous avons executé la commande ***vstart --eth0=A -M 64 pc2***

```
(sodyam@sodyam)-[~/Bureau/Travaux_Pratique_Netkit/Lab]
$ vstart --eth0=A -M 64 pc2

===== Starting virtual machine "pc2" =====
Kernel:      /home/sodyam/Bureau/Travaux_Pratique_Netkit/netkit/kernel/netkit-kernel
Modules:     /home/sodyam/Bureau/Travaux_Pratique_Netkit/netkit/kernel/modules
Memory:      64 MB
Model fs:    /home/sodyam/Bureau/Travaux_Pratique_Netkit/netkit/fs/netkit-fs
Filesystem:  /home/sodyam/Bureau/Travaux_Pratique_Netkit/Lab/pc2.disk
Interfaces:  eth0 @ A (/home/sodyam/.netkit/hubs/vhub_sodyam_A.cnct)
Hostfs at:   /home/sodyam
```

Interface :



```
pc2
Starting system log daemon....
Starting kernel log daemon....

— Starting Netkit phase 2 init script —

#####
Lab directory (host):
Version: <none>
Author: <none>
Email: <none>
Web: <none>
Description:
<none>
#####

— Netkit phase 2 initialization terminated —

pc2 login: root (automatic login)
Last login: Sun Jan 23 20:51:56 UTC 2022 on tty1
pc2:~#
```

❖ CREATION DES RESSOURCES DU RESEAU DMZ

DMZ est aussi un réseau mais qui est dans un autre domaine de collision, ici nommé B. Dans ce réseau, nous avons créé deux machines serveurs: le serveur mail/web et le serveur DNS,FTP/SSH. serveur mail/web

vstart --eth0=B -M 64 mail_web_serv

```
(sodyam@sodyam)-[~/Bureau/Travaux_Pratique_Netkit/Lab]
$ vstart --eth0=B -M 64 mail_web_serv
ci nommé B
web et le serveur DNS,FTP/SSH.
===== Starting virtual machine "mail_web_serv" =====
```

Le konsole d'execution est:

```
mail_web_serv

Starting system log daemon....
Starting kernel log daemon....

-- Starting Netkit phase 2 init script --

=====
Lab directory (host):
Version: <none>
Author: <none>
Email: <none>
Web: <none>
Description:
<none>
=====

-- Netkit phase 2 initialization terminated --

mail_web_serv login: root (automatic login)
Last login: Mon Jan 24 17:28:17 UTC 2022 on ttyd
mail_web_serv:~#
```

➤ serveur dns/ftp

vstart --eth0=B -M 64 dns_serv

```
(sodyam@sodyam)-[~/Bureau/Travaux_Pratique_Netkit/Lab]
$ vstart --eth0=B -M 64 dns_serv
l/netkit-kernel
l/modules
===== Starting virtual machine "dns_serv" =====
```

Interface du konsole d'execution

```
dns_serv

Starting system log daemon....
Starting kernel log daemon....

-- Starting Netkit phase 2 init script --

=====
Lab directory (host):
Version: <none>
Author: <none>
Email: <none>
Web: <none>
Description:
<none>
=====

-- Netkit phase 2 initialization terminated --

dns_serv login: root (automatic login)
Last login: Mon Jan 24 16:13:47 UTC 2022 on tty0
dns_serv:~#
```

❖ CREATION DES RESSOURCES DU RESEAU INTERNET

Pour pouvoir faire des test de connectivité entre les machines du réseau, nous avons crée deux machines dans le réseau internet à savoir **pc1_internet** et **pc2_internet**.

Ici, ces machines sont connectées au domaine de collision C.

➤ pc1_internet

vstart --eth0=C -M 64 pc1_internet

```
(sodyam@sodyam)-[~/Bureau/Travaux_Pratique_Netkit/Lab]
$ vstart --eth0=C -M 64 pc1_internet

===== Starting virtual machine "pc1_internet" =====
```

Interface d'exécution

```
pc1_internet

Starting system log daemon....
Starting kernel log daemon....

— Starting Netkit phase 2 init script —

=====

Lab directory (host):
Version: <none>
Author: <none>
Email: <none>
Web: <none>
Description:
<none>

=====

— Netkit phase 2 initialization terminated —

pc1_internet login: root (automatic login)
Last login: Sun Jan 23 20:51:56 UTC 2022 on tty1
pc1_internet:~#
```

➤ pc2_internet

vstart --eth0=C -M 64 pc2_internet

```
(sodyam@sodyam)-[~/Bureau/Travaux_Pratique_Netkit/Lab]
$ vstart --eth0=C -M 64 pc2_internet

Starting kernel log daemon....

— Starting Netkit phase 2 init script —

===== Starting virtual machine "pc2_internet" =====
```

❖ CREATION DES FIREWALL

Les firewall ici sont considérés comme des machines faisant office de routeur. Ils serviront à protéger les serveurs (de la DMZ). Le premier firewall (firewall1) lit le réseau local et le réseau DMZ.

Donc, ils possèdent deux interfaces que je nomme eth1 connecté au réseau Local et eth2 connecté au réseau DMZ. Le second firewall (firewall2) a aussi deux interfaces connectés respectivement au réseau DMZ et à l'internet.

➤ Creation du Firewall1

vstart --eth1=A --eth2=B -M 64 firewall1

```
(sodyam@sodyam)-[~/Bureau/Travaux_Pratique_Netkit/Lab]
$ vstart --eth1=A --eth2=B -M 64 firewall1

===== Starting virtual machine "firewall1" =====
```

Son Interface d'exécution

```
Virtual machine "firewall1" booting up...
Activating swap...done.
Cleaning up ifupdown...
Mounting kernel modules directory (/home/sodyam/Bureau/Travaux_Pratique_Netkit/netkit/kernel/modules/lib/modules) on /lib/modules/ ...
Loading kernel modules...done.
Setting kernel variables (/etc/sysctl.conf)...done.
Setting up networking...
Configuring network interfaces...done.
Starting portmap daemon...
INIT: Entering runlevel: 2

--- Starting Netkit phase 1 init script ---
Mounting /home/sodyam on /hosthome...
--- Netkit phase 1 initialization terminated ---

Starting system log daemon...
Starting kernel log daemon...

--- Starting Netkit phase 2 init script ---
=====
Lab directory (host):
Version: <none>
Author: <none>
Email: <none>
Web: <none>
Description:
<none>
=====
--- Netkit phase 2 initialization terminated ---

firewall1 login: root (automatic login)
Last login: Mon Jan 24 16:13:47 UTC 2022 on tty0
firewall1:~#
```

➤ Creation du Firewall2

vstart --eth1=B --eth2=C -M 64 firewall2

```
(sodyam@sodyam)-[~/Bureau/Travaux_Pratique_Netkit/Lab]
$ vstart --eth1=B --eth2=C -M 64 firewall2

===== Starting virtual machine "firewall2" =====
```


Interface d'execution

```
firewall2
loading kernel modules...done.
Setting kernel variables (/etc/sysctl.conf)...done.
Setting up networking....
Configuring network interfaces...done.
Starting portmap daemon....
INIT: Entering runlevel: 2

— Starting Netkit phase 1 init script —
Mounting /home/sodgiam on /hosthome...
— Netkit phase 1 initialization terminated —

Starting system log daemon....
Starting kernel log daemon....

— Starting Netkit phase 2 init script —
*****

Lab directory (host):
Version: <none>
Author: <none>
Email: <none>
Web: <none>
Description:
```

3. CONFIGURATION DES INTERFACES DES RESSOURCES

a. CONFIGURATION DES INTERFACES DES DE CHAQUE MACHINE

Ici, nous allons configurer les interfaces de chacun des machines des différents réseaux. Il s'agit de de configurer leur adresses d'interface.

Pour le faire, nous allons utiliser la commande **ifconfig** qui a pour syntaxe **ifconfig [interface] [adresse d'interface IP] netmask [masque réseau] broadcast [adresse de diffusion] up**

L'option **netmask** nous permet de spécifier le masque du réseau et broadcast l'adresse de diffusion. Nous utilisons l'option **up** pour rendre toujours actif l'interface.

1. MACHINES DU RESEAU LOCAL

Selon l'architecture, le réseau local a pour adresse réseau 192.168.0.0 avec un masque de 255.255.255.0. L'adresse de diffusion est de 192.168.0.255.

Toutes les machines de ce réseau auront pour passerelle par défaut l'adresse d'interface du firewall1. Ainsi, pour configurer la passerelle par défaut au niveau de ces machines, nous avons tapé la commande **route add default gw 192.168.0.5 dev eth0**

- pc1

La commande :

ifconfig eth0 192.168.0.1 netmask 255.255.255.0 broadcast 192.168.0.255 up Configuration de la passerelle par défaut : **route add default gw 192.168.0.5 dev eth0.**

Cette commande veut dire que le pc1 va confier le paquet à l'équipement d'adresse d'interface 192.168.0.5. Ce paquet sortira de son interface eth0.

```
pc1:~# ifconfig eth0 192.168.0.1 netmask 255.255.255.0 broadcast 192.168.0.5 up
pc1:~# █
```

Configuration de la passerelle par défaut :

route add default gw 192.168.0.5 dev eth0

Cette commande veut dire que le pc1 va confier le paquet à l'équipement d'adresse d'interface 192.168.0.5. Ce paquet sortira de son interface eth0.

```
pc1:~# route add default dev 192.168.0.5 dev eth0
Usage: inet_route [-vF] del {-host|-net} Target[/prefix] [gw Gw] [metric M] [[dev If]
inet_route [-vF] add {-host|-net} Target[/prefix] [gw Gw] [metric M]
                        [netmask N] [mss Mss] [window W] [irtt I]
                        [mod] [dyn] [reinststate] [[dev] If]
inet_route [-vF] add {-host|-net} Target[/prefix] [metric M] reject
inet_route [-FC] flush      NOT supported
pc1:~# █
```

- **printer**

C'est la même commande que nous avons tapé ici sauf que l'adresse IP de l'imprimante est de 192.168.0.2

```
printer:~# ifconfig eth0 192.168.0.2 netmask 255.255.255.0 broadcast 192.168.0.5 up
printer:~# route add default gw 192.168.0.5 █
```

- **file_serv**

C'est le serveur de fichier ayant pour adresse 192.168.0.4

```
file_serv:~# ifconfig eth0 192.168.0.4 netmask 255.255.255.0 broadcast 192.168.0.255 up
file_serv:~# route add default gw 192.168.0.5 dev eth0 █
```

➤ pc2

Adresse IP : 192.168.0.3

```
pc2
pc2:~# ifconfig eth0 192.168.0.3 netmask 255.255.255.0 broadcast 192.168.0.5 up
pc2:~# route add default gw 192.168.0.5
```

➤ pc3

Ici, son adresse est la même chose, alors nous avons attribuer l' Adresse IP : 192.168.0.6

```
— Netkit phase 2 initialization terminated —

pc3 login: root (automatic login)
pc3:~# ifconfig eth0 192.168.0.6 netmask 255.255.255.0 broadcast 192.168.0.255
up
pc3:~# route add default gw 192.168.0.5 dev eth0
```

2. MACHINES DU RESEAU DMZ

L'adresse réseau de ce réseau est le 10.0.0.0 avec un masque de 255.0.0.0 et une adresse de diffusion de 10.255.255.255. Dans ce réseau se trouve machines serveurs.

Etant donné que ce réseau se trouve au milieu de deux réseaux différents, nous avons choisi l'adresse du firewall1 (10.0.0.3) comme le gateway par défaut.

```
mail_web_serv
mail_web_serv:~# ifconfig eth0 10.0.0.1 netmask 255.0.0.0 broadcast 10.255.255.255 up
mail_web_serv:~# route add default gw 10.0.0.3 dev eth0
```

Pour ce qui concerne le serveur DNS, c'est la même commande sauf que l'adresse IP de son interface est 10.0.0.4.

```
dns_serv
dns_serv:~# ifconfig eth0 10.0.0.4 netmask 255.0.0.0 broadcast 10.255.255.255 up
dns_serv:~# route add default gw 10.0.0.3 dev eth0
dns_serv:~#
```

3. MACHINES DU RESEAU DMZ

D'après l'architecture, l'adresse de ce réseau est de 139.124.0.0, le masque associé est de 255.255.0.0 et le broadcast est 139.124.255.255.

L'adresse d'interface du routeur qui est de 139.124.44.223. C'est le gateway par défaut des machines du reseau internet.

➤ pc1_internet

```
pc1_internet
pc1_internet:~# ifconfig eth0 139.124.44.223 netmask 255.255.0.0 broadcast 139.124.255.255 up
pc1_internet:~# route add default 139.124.44.223 dev eth0
Usage: inet_route [-vF] del {-host|-net} Target[/prefix] [gw Gw] [metric M] [[dev] If]
       inet_route [-vF] add {-host|-net} Target[/prefix] [gw Gw] [metric M]
       [netmask N] [mss Mss] [window W] [irtt I]
       [mod] [dyn] [reinststate] [[dev] If]
       inet_route [-vF] add {-host|-net} Target[/prefix] [metric M] reject
       inet_route [-vF] flush
pc1_internet:~#
```

ifconfig eth0
139.124.0.1
netmask
255.255.0.0
broadcast

139.124.44.223 up

➤ pc2_internet

L'adresse associé à cet interface est : 192.168.0.2, on a donc :

```
pc2_internet:~# ifconfig eth0 139.124.0.2 netmask 255.255.0.0 broadcast 139.124.255.255 up
pc2_internet:~# route add default 139.124.44.223 dev eth0
Usage: inet_route [-vF] del {-host|-net} Target[/prefix] [gw Gw] [metric M] [[dev] If]
       inet_route [-vF] add {-host|-net} Target[/prefix] [gw Gw] [metric M]
       [netmask N] [mss Mss] [window W] [irtt I]
       [mod] [dyn] [reinststate] [[dev] If]
       inet_route [-vF] add {-host|-net} Target[/prefix] [metric M] reject
       inet_route [-vF] flush
pc2_internet:~#
```

4. CONFIGURATION DES FIREWALL

Dans un premier temps, nous avons configuré l'adresse d'interface des firewall.

Voilà comment nous avons configuré le premier firewall :

#configuration des adresses des interfaces

ifconfig eth1 192.168.0.5 netmask 255.255.255.0 broadcast

192.168.0.255 up

ifconfig eth2 10.0.0.3 netmask 255.0.0.0 broadcast 10.255.255.255 up

#Ajout du reseau internet à la table de routage du firewall

ip route add 139.124.0.0/16 via 10.0.0.2 dev eth2

Configuration de la route par défaut :

route add default gw 10.0.0.2 dev eth2 pour le firewall1 **et route add default gw 10.0.0.3 dev eth1** pour le firewall2.

```
firewall1
firewall1:~# ifconfig eth1 192.168.0.5 netmask 255.255.255.0 broadcast 192.168.0.255 up
firewall1:~# ifconfig eth2 10.0.0.3 netmask 255.0.0.0 broadcast 10.255.255.255 up
firewall1:~# ip route add 139.124.0.0/16 via 10.0.0.2 eth1
```

Afin de

pouvoir faire sortir des paquets sur l'internet, nous avons configuré le NAT.

La seule règle que nous avons paramétrer est la commande :

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

```
firewall1
/hostlab/firewall1.startup: line 14: sysctl-w: command not found
net.ipv4.ip_forward = 1
>>> End of firewall1 specific startup script.

=====
Lab directory (host): /home/sodyam/Bureau/Travaux_Pratique_Netkit/Lab
Version: 1.0
Author: Yao Marius SODOKIN
firewall1:~# iptables -t nat -A POSTROUTING -o eth2 -j MASQUERADE
firewall1:~#
```

➤ **Firewall1**

➤ **Firewall2**

```
firewall2
firewall2:~# iptables -t nat -A POSTROUTING -o eth2 -j MASQUERADE
firewall2:~#
```

-t nat : indique la table du noyau sur laquelle on intervient

eth2: représente l'interface via lequel les paquets vont partir sur l'internet.

-o : précise qu'il s'agit d'une interface de sortir

-A POSTROUTING indique que le NAT intervient après le routage des paquets. Et juste avant qu'ils ne soient renvoyé sur l'interface sélectionné (ici eth2)

-j MASQUERADE précise le masquage de l'origine des paquets

❖ ACTIVATION DU ROUTAGE STATIQUE

Le routage statique va permettre aux firewall de relayer le paquet vers le réseau cible. Pour cela, il faut que le contenu du **fichier /proc/sys/net/ipv4/ip_forward** **doit valoir 1**.

Ainsi, nous avons l'éditer avec la commande :

echo 1 > /proc/sys/net/ipv4/ip_forward

```
firewall1
firewall1:~# echo 1 > /proc/sys/net/ipv4/ip_forward
firewall1:~#
```

```
firewall2
firewall2:~# echo 1 > /proc/sys/net/ipv4/ip_forward
firewall2:~#
```

❖ CONFIGURATION DU ROUTAGE STATIQUE

Ici, nous avons ajouter à la table de routage du firewall1 l'adresse du réseau côté internet, c'est à dire 193.124.0.0/16 avec la ligne de commande ci après :
ip route add 139.124.0.0/16 via 10.0.0.2 dev eth2.

De même pour le firewall2, nous avons ajouter le réseau 192.168.0.0 à sa table de routage puisqu'il n'est pas dans ce réseau.

ip route add 192.168.0.0/24 via 10.0.0.3 dev eth1.

Cette commande a été déjà taper précédemment..

❖ TABLE DE ROUTAGE DES FIREWALL

Avec la commande route -n sur chaque firewall, nous obtenons les captures d'écran suivant :

➤ **firewall1**

```
firewall1
firewall1:~# route -n
Kernel IP routing table
Destination Gateway Genmask Flags Metric Ref Use Iface
192.168.0.0 0.0.0.0 255.255.255.0 U 0 0 0 eth1
139.124.0.0 10.0.0.2 255.255.0.0 UG 0 0 0 eth2
10.0.0.0 0.0.0.0 255.0.0.0 U 0 0 0 eth2
0.0.0.0 10.0.0.2 0.0.0.0 UG 0 0 0 eth2
firewall1:~#
```

Nous avons constaté que ce firewall se trouve dans deux réseaux. Il arrive à joindre le troisième réseau.

➤ Firewall2

```
firewall2
firewall2:~# route add default gw 139.124.44.223
firewall2:~# route -n
Kernel IP routing table
Destination      Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.0.0      10.0.0.3        255.255.255.0   UG    0      0        0 eth1
139.124.0.0      0.0.0.0         255.255.0.0     U      0      0        0 eth2
10.0.0.0         0.0.0.0         255.0.0.0       U      0      0        0 eth1
0.0.0.0          139.124.44.223 0.0.0.0         UG    0      0        0 eth2
firewall2:~#
```

3 -c) TEST DE COMMUNICATION ENTRE LES MACHINES DU RESEAU DMZ

Nous avons ensuite tester la connectivité entre les machines d u réseau DMZ. Nous avons constaté que la communication marche bien entre les deux machines serveurs du réseau.

➤ Test du serveur DNS vers Serveur Mail

```
dns_serv
-----
--- Netkit phase 2 initialization terminated ---
dns_serv:~# ping 10.0.0.1
PING 10.0.0.1 (10.0.0.1) 56(84) bytes of data.
64 bytes from 10.0.0.1: icmp_seq=1 ttl=64 time=0.235 ms
64 bytes from 10.0.0.1: icmp_seq=2 ttl=64 time=0.129 ms
64 bytes from 10.0.0.1: icmp_seq=3 ttl=64 time=0.219 ms
64 bytes from 10.0.0.1: icmp_seq=4 ttl=64 time=0.248 ms
64 bytes from 10.0.0.1: icmp_seq=5 ttl=64 time=0.242 ms
^C
--- 10.0.0.1 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 3996ms
rtt min/avg/max/mdev = 0.129/0.214/0.248/0.046 ms
dns_serv:~#
```

➤ Test de connexivité du mail_web_serv vers le serveur DNS (sens inverse)

```
mail_web_serv
mail_web_serv:~# ping 10.0.0.4
PING 10.0.0.4 (10.0.0.4) 56(84) bytes of data.
64 bytes from 10.0.0.4: icmp_seq=1 ttl=64 time=43.5 ms
64 bytes from 10.0.0.4: icmp_seq=2 ttl=64 time=0.756 ms
64 bytes from 10.0.0.4: icmp_seq=3 ttl=64 time=0.807 ms
64 bytes from 10.0.0.4: icmp_seq=4 ttl=64 time=0.754 ms
64 bytes from 10.0.0.4: icmp_seq=5 ttl=64 time=0.703 ms
^C
--- 10.0.0.4 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4035ms
rtt min/avg/max/mdev = 0.703/9.314/43.550/17.118 ms
mail_web_serv:~#
```

3 -d) TEST DE COMMUNICATION ENTRE LES MACHINES DU RESEAU INTERNET

Nous avons crée deux machines dans ce réseau pour pouvoir faire les tests.

- **Test de connectivité du pc1_internet vers pc2_internet (d'adresse 139.124.0.2)**

```
pc1_internet
Author: 50003CN Yao Harius
Email: yao50003cn@gmail.com
Web:
Description:
PROJECT: NETKIT IPFI

*****
--- Netkit phase 2 initialization terminated ---

pc1_internet login: root (automatic login)
pc1_internet:~# ping 139.124.0.2
PING 139.124.0.2 (139.124.0.2) 56(84) bytes of data.
64 bytes from 139.124.0.2: icmp_seq=1 ttl=64 time=0.146 ms
64 bytes from 139.124.0.2: icmp_seq=2 ttl=64 time=0.652 ms
64 bytes from 139.124.0.2: icmp_seq=3 ttl=64 time=0.655 ms
64 bytes from 139.124.0.2: icmp_seq=4 ttl=64 time=0.658 ms
64 bytes from 139.124.0.2: icmp_seq=5 ttl=64 time=0.640 ms
^C
--- 139.124.0.2 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4012ms
rtt min/avg/max/mdev = 0.146/0.544/0.658/0.199 ms
pc1_internet:~#
```

- **Test de connectivité du PC2_internet vers pc1_internet (le retour)**

```
pc2_internet
pc2_internet:~# ping 139.124.0.1
PING 139.124.0.1 (139.124.0.1) 56(84) bytes of data.
64 bytes from 139.124.0.1: icmp_seq=1 ttl=64 time=9.86 ms
64 bytes from 139.124.0.1: icmp_seq=2 ttl=64 time=0.670 ms
64 bytes from 139.124.0.1: icmp_seq=3 ttl=64 time=0.776 ms
64 bytes from 139.124.0.1: icmp_seq=4 ttl=64 time=0.702 ms
^C
--- 139.124.0.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3047ms
rtt min/avg/max/mdev = 0.670/3.002/9.863/3.961 ms
pc2_internet:~#
```

3 -e) TEST DE COMMUNICATION ENTRE LES MACHINES DU RESEAU LOCAL, ET RESEAU DMZ .

- **Test de connectivité du Server de fichier (file_serv) du reseau Local vers le server DNS du réseau DM**


```

*****
--- Netkit phase 2 initialization terminated ---

file_serv login: root (automatic login)
file_serv:~# ping 10.0.0.4
PING 10.0.0.4 (10.0.0.4) 56(84) bytes of data.
64 bytes from 10.0.0.4: icmp_seq=1 ttl=63 time=0.516 ms
64 bytes from 10.0.0.4: icmp_seq=2 ttl=63 time=0.371 ms
64 bytes from 10.0.0.4: icmp_seq=3 ttl=63 time=0.369 ms
64 bytes from 10.0.0.4: icmp_seq=4 ttl=63 time=0.249 ms
64 bytes from 10.0.0.4: icmp_seq=5 ttl=63 time=0.666 ms
64 bytes from 10.0.0.4: icmp_seq=6 ttl=63 time=0.895 ms
^C
--- 10.0.0.4 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 4995ms
rtt min/avg/max/mdev = 0.249/0.511/0.895/0.215 ms
file_serv:~#

```

La connexion s'est bien passée

- Test de connectivité du réseau du serveur mail (mail_web_serv) du réseau DMZ vers le PC2 du réseau local et d'adresse 192.168.0.3

```

mail_web_serv
Author: SODOKIN Yao Marius
Email: yasomeriussodokin@gmail.com
Web: <none>
Description:
PROJET NETKIT IFRI
*****
--- Netkit phase 2 initialization terminated ---

mail_web_serv login: root (automatic login)
mail_web_serv:~# Ping 10.0.0.4
PING 10.0.0.4 (10.0.0.4) 56(84) bytes of data.
64 bytes from 10.0.0.4: icmp_seq=1 ttl=64 time=6.65 ms
64 bytes from 10.0.0.4: icmp_seq=2 ttl=64 time=0.607 ms
64 bytes from 10.0.0.4: icmp_seq=3 ttl=64 time=0.580 ms
64 bytes from 10.0.0.4: icmp_seq=4 ttl=64 time=0.617 ms
64 bytes from 10.0.0.4: icmp_seq=5 ttl=64 time=0.606 ms
^C
--- 10.0.0.4 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4013ms
rtt min/avg/max/mdev = 0.580/1.812/6.650/2.419 ms
mail_web_serv:~#

```

- Test de connectivité du pc1_internet (du réseau internet) vers le serveur DNS (du réseau DMZ) d'adresse 10.0.0.4

```

pc1_internet
pc1_internet:~# ping 10.0.0.4
PING 10.0.0.4 (10.0.0.4) 56(84) bytes of data.
64 bytes from 10.0.0.4: icmp_seq=1 ttl=62 time=17.8 ms
64 bytes from 10.0.0.4: icmp_seq=2 ttl=62 time=1.54 ms
64 bytes from 10.0.0.4: icmp_seq=3 ttl=62 time=1.21 ms
64 bytes from 10.0.0.4: icmp_seq=4 ttl=62 time=1.37 ms
64 bytes from 10.0.0.4: icmp_seq=5 ttl=62 time=1.33 ms
64 bytes from 10.0.0.4: icmp_seq=6 ttl=62 time=1.42 ms
^C
--- 10.0.0.4 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5051ms
rtt min/avg/max/mdev = 1.219/4.128/17.874/6.148 ms
pc1_internet:~#

```

- Test de connectivité du pc2_internet (de réseau internet) vers le le printer du réseau local (d'adresse 192.168.0.2)

```
pc2_internet
pc2_internet:~# ping 192.168.0.2
PING 192.168.0.2 (192.168.0.2) 56(84) bytes of data.
64 bytes from 192.168.0.2: icmp_seq=1 ttl=62 time=21.6 ms
64 bytes from 192.168.0.2: icmp_seq=2 ttl=62 time=1.45 ms
64 bytes from 192.168.0.2: icmp_seq=3 ttl=62 time=1.57 ms
64 bytes from 192.168.0.2: icmp_seq=4 ttl=62 time=1.75 ms
64 bytes from 192.168.0.2: icmp_seq=5 ttl=62 time=1.69 ms
^C
--- 192.168.0.2 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4033ms
rtt min/avg/max/mdev = 1.456/5.617/21.607/7.995 ms
pc2_internet:~#
```

- Test de connectivité du printer (du réseau local) vers le pc2_internet du réseau internet (d'adresse 139.124.0.2)

```
printer
Web: <none>
Description:
PROJET NETKIT IFRI
*****
--- Netkit phase 2 initialization terminated ---

printer login: root (automatic login)
printer:~# ping 139.124.0.2
PING 139.124.0.2 (139.124.0.2) 56(84) bytes of data.
64 bytes from 139.124.0.2: icmp_seq=1 ttl=62 time=28.4 ms
64 bytes from 139.124.0.2: icmp_seq=2 ttl=62 time=1.63 ms
64 bytes from 139.124.0.2: icmp_seq=3 ttl=62 time=1.83 ms
64 bytes from 139.124.0.2: icmp_seq=4 ttl=62 time=1.45 ms
64 bytes from 139.124.0.2: icmp_seq=5 ttl=62 time=1.80 ms
64 bytes from 139.124.0.2: icmp_seq=6 ttl=62 time=1.60 ms
64 bytes from 139.124.0.2: icmp_seq=7 ttl=62 time=1.79 ms
^C
--- 139.124.0.2 ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 6065ms
rtt min/avg/max/mdev = 1.450/5.504/28.416/9.354 ms
printer:~#
```

- Test de pc1_internet du réseau internet vers le pc2 du réseau local (d'adresse 192.168.0.3)

```
pc1_internet
pc1_internet:~# ping 192.168.0.3
PING 192.168.0.3 (192.168.0.3) 56(84) bytes of data.
64 bytes from 192.168.0.3: icmp_seq=1 ttl=62 time=27.0 ms
64 bytes from 192.168.0.3: icmp_seq=2 ttl=62 time=1.79 ms
64 bytes from 192.168.0.3: icmp_seq=3 ttl=62 time=1.64 ms
64 bytes from 192.168.0.3: icmp_seq=4 ttl=62 time=1.64 ms
64 bytes from 192.168.0.3: icmp_seq=5 ttl=62 time=0.997 ms
64 bytes from 192.168.0.3: icmp_seq=6 ttl=62 time=0.631 ms
^C
--- 192.168.0.3 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5052ms
rtt min/avg/max/mdev = 0.631/5.621/27.017/9.577 ms
pc1_internet:~#
```

CONCLUSION

Nous constatons que chaque machine arrive à joindre avec succès toutes les autres machines des autres réseaux.

5. AUTOMATISATION DES TACHES AVEC LA CREATION D'UN FICHIER LAB

Nous avons profiter à la creation d'un fichier lab.conf.

Le fichier lab.conf est un fichier qui contient toutes les créations et leurs configurations de l'architecture du réseau. Ceci pour ne pas ne pas perdre les configurations après le redémarrage de la machine ou du terminale. Nous l'avons créer en trois étapes ci-après :

1. CREATION DE L'ARBORESCENCE

Ici, nous avons créer les dossiers et des fichiers.

a. CREATION ET ACCES AU DOSSIER DE LAB

mkdir Lab && cd Lab

b. CREATION D'UN DOSSIER POUR CHAQUE MACHINE

Ces dossiers serviront pour l'execution des machines virtuelles du lab.

***mkdir pc1 pc2 pc3 printer file_serv mail_web_serv dns_serv
pc1_internet pc2_internet firewall1 firewall2***

```
(sodyam@sodyam)-[~/Bureau/Travaux_Pratique_Netkit]
$ mkdir pc1 pc2 pc3 printer switch1 switch2 file_serv mail_web_serv dns_serv firewall1 firewall2 pc1_internet pc2_internet
(sodyam@sodyam)-[~/Bureau/Travaux_Pratique_Netkit]
$
```

c. CREATION DES FICHIERS .startup

Nous avons créer ensuite, un fichier .startup pour chaque machine .

Ces fichiers serviront pour la configuration de chaque machine virtuelle.

Le fichier porte le nom de la machine. La commande tapée est la suivante :

***touch pc1.startup pc2.startup pc3.startup printer.startup
file_serv.startup mail_web_serv.startup dns.startup***

```
(sodyam@sodyam)-[~/Bureau/Travaux_Pratique_Netkit]
$ touch pc1.startup pc2.startup pc3.startup printer.startup file_serv.startup mail_web_serv.startup dns.startup firewall1.startup firewall2.startup pc1_internet.startup pc2_internet.startup
touch lab.conf
(sodyam@sodyam)-[~/Bureau/Travaux_Pratique_Netkit]
(sodyam@sodyam)-[~/Bureau/Travaux_Pratique_Netkit]
```

***pc1_internet.startup pc2_internet.startup firewall1.startup
firewall2.startup***

D. CREATION DU FICHIER LAB.CONF

Nous avons par la suite, crée le fichier lab.conf.

C'est de fichier de configuration du lab. Dans ce fichier, nous allons créer les machines et leurs interfaces.

touch lab.conf

```
(sodyam@sodyam)-[~/Bureau/Travaux_Pratique_Netkit]
$ touch lab.conf
(sodyam@sodyam)-[~/Bureau/Travaux_Pratique_Netkit]
```

2. CREATION DU CONTENU

❖ Fichier lab.conf

C'est dans cette fichier se trouve la création et la configuration des machines. C'est aussi dans cette fichier que nous avons renseigné les informations sur le lab.

```
LAB_DESCRIPTION="PROJET NETKIT IFRI"
LAB_VERSION=1.0 LAB_AUTHOR="SODOKIN Yao Marius"
LAB_EMAIL="yaomariussodokin@gmail.com"
```

#Creation des machines du réseau LOCAL

```
pc1[0]=A
pc1[mem]=64
pc2[0]=A
```

```
pc2[mem]=64
pc3[0]=A
pc3[mem]=64
printer[0]=A
printer[mem]=64
file_serv[0]=A
file_serv[mem]=64
```

#Création des firewall

```
firewall1[1]=A
firewall1[2]=B
firewall1[mem]=64
```

```
firewall2[1]=B
firewall2[2]=C
firewall2[mem]=64
```

#Creation des machines du réseau DMZ

```
mail_web_serv[0]=B
mail_web_serv[mem]=64
```

```
dns_serv[0]=B
dns_serv[mem]=64
```

#Création des machines du réseau Internet

```
pc1_internet[0]=C pc1_
internet[mem]=64
```

```
pc2_internet[0]=C
pc2_internet[mem]=64
```

Ici pour créer une machine, nous utilisons la commande :

nom_machine[interface]=[domaine de collision]

et pour préciser la mémoire allouée, nous suivons la syntaxe

nom_machine[mem]=[taille]

mem est une option permettant d'indiquer qu'il s'agit de mémoire.

Exemple : comme dans la liste des commandes :

pc1[0]=A

pc1[mem]=64

Ces deux lignes signifient respectivement que la machine pc1 est créée avec son interface eth0 relié au domaine de collision A. Et auquel une mémoire de 64MB est allouée.

➤ **Fichier pc1.startup**

```
ifconfig eth0 192.168.0.1 netmask 255.255.255.0 broadcast  
up 192.168.0.255 up  
route add default gw 192.168.0.5 dev eth0 Fichier
```

➤ **pc2.startup**

```
ifconfig eth0 192.168.0.3 netmask 255.255.255.0 broadcast  
192.168.0.255 up  
route add default gw 192.168.0.5 dev eth0
```

➤ **Fichier pc3.startup**

```
ifconfig eth0 192.168.0.6 netmask 255.255.255.0 broadcast 192.168.0.255  
up  
route add default gw 192.168.0.5 dev eth0
```

➤ **Fichier printer.startup**

```
ifconfig eth0 192.168.0.2 netmask 255.255.255.0 broadcast  
192.168.0.255 up  
route add default gw 192.168.0.5 dev eth0
```

➤ **Fichier file_serv.startup**

```
ifconfig eth0 192.168.0.4 netmask 255.255.255.0 broadcast 192.168.0.255  
up  
route add default gw 192.168.0.5 dev eth0
```

#Mise en place du service FTP

/etc/init.d/proftpd start

➤ **Fichier firewall1.startup**

```
ifconfig eth1 192.168.0.5 netmask 255.255.255.0 broadcast 192.168.0.255 up
ifconfig eth2 10.0.0.3 netmask 255.0.0.0 broadcast 10.255.255.255 up
```

```
#Ajout du réseau internet à la table de routage
ip route add 139.124.0.0/16 via 10.0.0.2 dev eth2
```

```
#Route par défaut route add default gw 10.0.0.2 dev eth2
```

```
#Configuration du NAT
```

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

```
#Activation du NAT
```

```
echo 1 > /proc/sys/net/ipv4/ip_forward
```

```
#Autorisation des connexions SSH en entrée et en sortie avec le réseau
```

```
localiptables -A FORWARD -p tcp --dport 22 -j ACCEPT
```

```
# Règle à appliquer sur le firewall1
```

```
iptables -A FORWARD -m state --state ESTABLISHED,RELATED -j ACCEPT
```

```
iptables -A FORWARD -m state --state NEW -p tcp --dport 80 -j ACCEPT
```

```
#Autorisation des trafics
```

```
iptables -A FORWARD -p icmp -j ACCEPT
```

```
iptables -A FORWARD -p tcp --dport 110 -j ACCEPT
```

```
iptables -A FORWARD -p tcp --dport 25 -j ACCEPT
```

```
iptables -A FORWARD -p tcp --dport 21 -j ACCEPT
```

```
# REDIRECTION DES PAQUETS
```

```
iptables -t nat -A PREROUTING -i eth1 -p tcp --dport 25 -j DNAT --to-destination 10.0.0.1:25
```

```
iptables -t nat -A PREROUTING -i eth1 -p tcp --dport 110 -j DNAT --to-destination 10.0.0.1:110
```

```
iptables -t nat -A PREROUTING -i eth1 -p tcp --dport 21 -j DNAT --to-destination 10.0.0.4:21
```

```
iptables -t nat -A PREROUTING -i eth1 -p tcp --dport 22 -j DNAT --to-destination 10.0.0.4:22
```

➤ **Fichier firewall2.startup**

```
ifconfig eth1 10.0.0.2 netmask 255.0.0.0 broadcast 10.255.255.255 up
ifconfig eth2 139.124.44.223 netmask 255.255.0.0 broadcast 139.124.255.255
up
```

```
#Ajout du réseau internet à la table de routage
ip route add 192.168.0.0/24 via 10.0.0.3 dev eth1
```

```
#Route par défaut route add default gw 10.0.0.3 dev eth1
```

```
#Configuration du NAT iptables -t nat -A POSTROUTING -o eth2 -j
MASQUERADE
```

```
#Activation du NAT echo 1 > /proc/sys/net/ipv4/ip_forward # Règle à appliquer
sur le firewall2
```

```
iptables -A FORWARD -p tcp --dport 80 -m state --state NEW -j ACCEPT
iptables -A FORWARD -m state --state ESTABLISHED,RELATED -j ACCEPT
iptables -t nat -A PREROUTING -i eth1 -p tcp --dport 80 -j DNAT --to- destination
10.0.0.1:80
iptables -A OUTPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
iptables -A OUTPUT -m state --state NEW -p udp --dport 53 -j ACCEPT
iptables -t nat -A POSTROUTING -o eth1 -j MASQUERADE
```

```
#Autorisation des traffics
```

```
iptables -A FORWARD -p icmp -j ACCEPT iptables -A FORWARD -p tcp --dport
110 -j ACCEPT
```

```
iptables -A FORWARD -p tcp --dport 25 -j ACCEPT iptables -A FORWARD -p
tcp --dport 21 -j ACCEPT
```

```
#Configuration du firewall2 et autorisation des trafics sur les ports 25,110,21,22
```

```
iptables -A FORWARD -p tcp --dport 25 -j ACCEPT
iptables -A FORWARD -p tcp --dport 110 -j ACCEPT
iptables -A FORWARD -p tcp --dport 21 -j ACCEPT
iptables -A FORWARD -p tcp --dport 22 -j ACCEPT
```

```
# REDIRECTION DES PAQUETS
```

```
iptables -t nat -A PREROUTING -i eth1 -p tcp --dport 25 -j DNAT --to-
destination 10.0.0.1:25
```


iptables -t nat -A PREROUTING -i eth1 -p tcp --dport 110 -j DNAT --to-destination 10.0.0.1:110

iptables -t nat -A PREROUTING -i eth1 -p tcp --dport 21 -j DNAT --to-destination 10.0.0.4:21

iptables -t nat -A PREROUTING -i eth1 -p tcp --dport 22 -j DNAT --to-destination 10.0.0.4.22

➤ **Fichier dns_serv.startup**

*ifconfig eth0 10.0.0.4 netmask 255.0.0.0 broadcast 10.255.255.255 up
route add default gw 10.0.0.3 dev eth0*

#Mise en place du service TELNET :

/etc/init.d/inetd start

#Mise en place du service SSH :

/etc/init.d/ssh start

#Mise en place du service FTP :

/etc/init.d/proftpd start2

#Mise en place du service DNS :

/etc/init.d/bind9 start

➤ **Fichier mail_web_serv.startup**

*ifconfig eth0 10.0.0.1 netmask 255.0.0.0 broadcast 10.255.255.255 up
route add default gw 10.0.0.3 dev eth0*

#Commande de mise en place du service MAIL

/etc/init.d/exim4 start

#Commande pour mettre en place le service WEB

/etc/init.d/apache2 start

➤ **Fichier pc1_internet.startup**

*ifconfig eth0 139.124.0.1 netmask 255.255.0.0 broadcast 139.124.255.255 up
route add default gw 139.124.44.223 dev eth0*

➤ **Fichier pc2_internet.startup**

*ifconfig eth0 139.124.0.2 netmask 255.255.0.0 broadcast 139.124.255.255 up
route add default gw 139.124.44.223 dev eth0*

❖ DEMARRAGE DU LAB

```
(sodyam@sodyam)-[~/Bureau/Travaux_Pratique_Netkit/Lab]
$ lstart -f

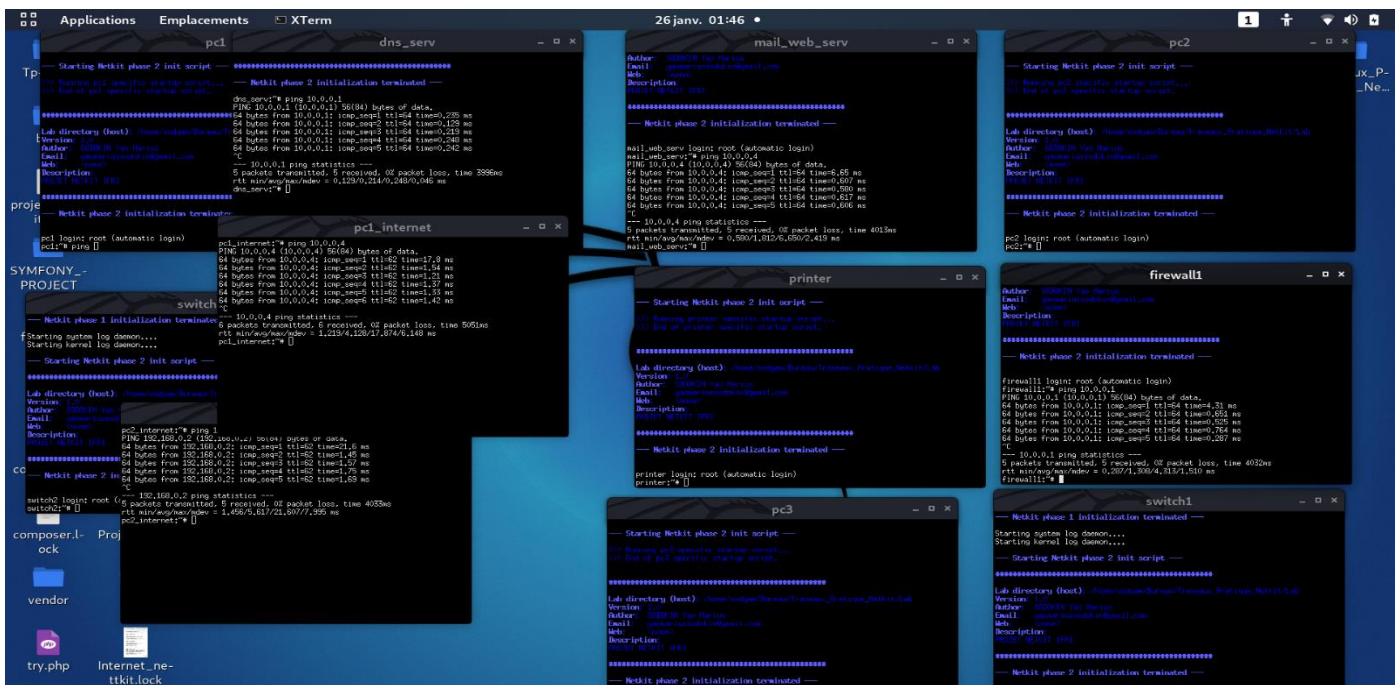
===== Starting lab =====
Lab directory: /home/sodyam/Bureau/Travaux_Pratique_Netkit/Lab
Version: 1.0
Author: SODOKIN Yao Marius
Email: yaomariussodokin@gmail.com
Web: <unknown>
Description:
PROJET NETKIT IFRI
=====
Starting "dns_serv"...
Starting "file_serv"...
Starting "firewall1"...
Starting "firewall2"...
Starting "mail_web_serv"...
Starting "pc1"...
Starting "pc1_internet"...
Starting "pc2"...
Starting "pc2_internet"...
Starting "pc3"...
Starting "printer"...
Starting "switch1"...
Starting "switch2"...

The lab has been started.
=====

(sodyam@sodyam)-[~/Bureau/Travaux_Pratique_Netkit/Lab]
```

Nous avons demarrer le lab avec la commande ***lstart -f***

❖ LES MACHINES EN EXECUTION



5. MISE EN PLACE DES SERVICES AU NIVEAU DES SERVEURS

Nous allons mettre en place les services sur les différents serveurs.

Il s'agit d'activer juste les services sur les différents serveurs. Les services serveurs se trouvent dans les fichiers, nous tapons juste la commande

/etc/init.d/[service] start

❖ SERVEURS DE FICHIERS (file_serv)

Ici, nous allons activer le service ftp du nom de **proftpd** par la Commande **/etc/init.d/proftpd start**

```

file_serv
file_serv:~# /etc/init.d/proftpd start
Starting ftp server: proftpd.
file_serv:~#

```

❖ SERVEUR MAIL /WEB (mail_web_serv)

/etc/init.d/exim4 start : Commande de mise en place du service MAIL

/etc/init.d/apache2 start : Commande pour mettre en place le service WEB.

```

mail_web_serv
mail_web_serv:~# /etc/init.d/apache2 start
Starting web server: apache2
apache2: Could not reliably determine the server's
fully qualified domain name, using 127.0.0.1 for ServerName
httpd (pid 799) already running
mail_web_serv:~# /etc/init.d/exim4 start
Starting MTA: exim4.
mail_web_serv:~#

```

❖ SERVEUR DNS, TELNET, SSH,FTP (dns_serv)

- Mise en place du service TELNET :

/etc/init.d/inetd start

- Mise en place du service SSH :

/etc/init.d/ssh start

- Mise en place du service FTP :

/etc/init.d/proftpd start

- Mise en place du service DNS :

/etc/init.d/bind9 star



```
dns_serv:~# /etc/init.d/bind9 start
Starting domain name service...: bind9.
dns_serv:~# /etc/init.d/inetd start
Starting internet superserver: inetd.
dns_serv:~# /etc/init.d/ssh start
Starting OpenBSD Secure Shell server: sshd.
dns_serv:~# /etc/init.d/proftpd start
Starting ftp server: proftpd.
dns_serv:~#
```

❖ CONFIGUARTION DES FIREWALL

1. Autorisation du trafic web (port 80) en entrée du serveur sur la DMZ.

- Règle à appliquer sur le firewall1

```
iptables -A FORWARD -p tcp --dport 80 -m state --state NEW -j ACCEPT
iptables -A FORWARD -m state --state ESTABLISHED,RELATED -j ACCEPT
iptables -t nat -A PREROUTING -i eth1 -p tcp --dport 80 -j DNAT --to-
destination 10.0.0.1:80 2.
```

2. Autorisation du trafic web (port 80) en sortie du réseau local -Règles à appliquer sur le firewall2

```
iptables -A FORWARD -m state --state ESTABLISHED,RELATED -j
ACCEPT
```

```
iptables -A FORWARD -m state --state NEW -p tcp --dport 80 -j ACCEPT
```

- Règles à appliquer sur le firewall1

```
iptables -A OUTPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
```

```
iptables -A OUTPUT -m state --state NEW -p udp --dport 53 -j ACCEPT
```

iptables -t nat -A POSTROUTING -o eth1 -j MASQUERADE

Après avoir autorisé le DNS, nous faisons un SNAT. Après l'application de ces règles, le PC1 arrive à faire du telnet sur notre serveur web et sur le PC Internet (Port 80). 3. Autorisation du trafic ICMP, pop, smtp et ftp en entrée et en sortie sur la DMZ Signalons que après l'application des règles précédentes sur les firewall, les ping ne passent plus, tout simplement parce que le trafic icmp n'a pas encore été autorisé.

Nous allons à présent autoriser le trafic icmp,smtp,pop,ftp.

iptables -A FORWARD -p icmp -j ACCEPT

iptables -A FORWARD -p tcp --dport 110 -j ACCEPT

iptables -A FORWARD -p tcp --dport 25 -j ACCEPT

iptables -A FORWARD -p tcp --dport 21 -j ACCEPT

. - Redirection du trafic vers les serveurs appropriés :

Configurons le firewall1 Avant d'effectuer la redirection, **autorisons le trafic sur les ports 25,110,21 et 22.**

iptables -A FORWARD -p tcp --dport 25 -j ACCEPT

iptables -A FORWARD -p tcp --dport 110 -j ACCEPT

iptables -A FORWARD -p tcp --dport 21 -j ACCEPT

iptables -A FORWARD -p tcp --dport 22 -j ACCEPT

Maintenant on procède à la redirection.

iptables -t nat -A PREROUTING -i eth1 -p tcp --dport 25 -j DNAT --to-destination 10.0.0.1:25 *iptables -t nat -A PREROUTING -i eth1 -p tcp --dport 110 -j DNAT --to-destination 10.0.0.1:110* *iptables -t nat -A PREROUTING -i eth1 -p tcp --dport 21 -j DNAT --to-destination 10.0.0.4:21* *iptables -t nat -A PREROUTING -i eth1 -p tcp --dport 22 -j DNAT --to-destination 10.0.0.4:22*

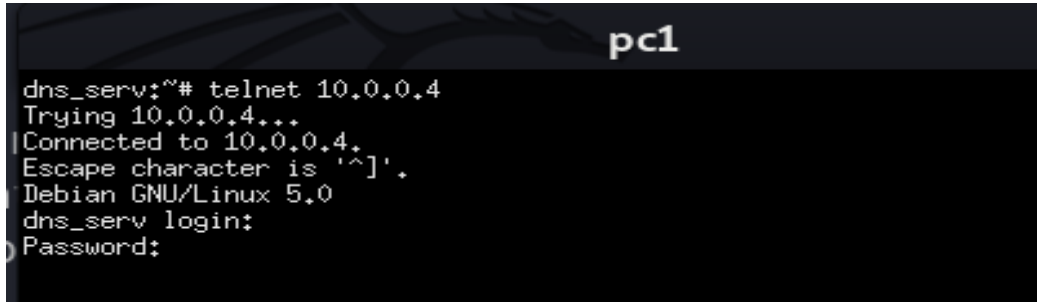
Avec les commandes ci-dessus, nous faisons du DNAT afin que le trafic adressé à l'interface du firewall2 (129.124.44.223) soit redirigé en fonction du port vers le service approprié..

Autorisation des connexions SSH en entrée et en sortie avec le réseau

localiptables -A FORWARD -p tcp --dport 22 -j ACCEPT

❖ ETAPE DE TEST DES SERVICES HEHERGES SUR LES SERVEURS

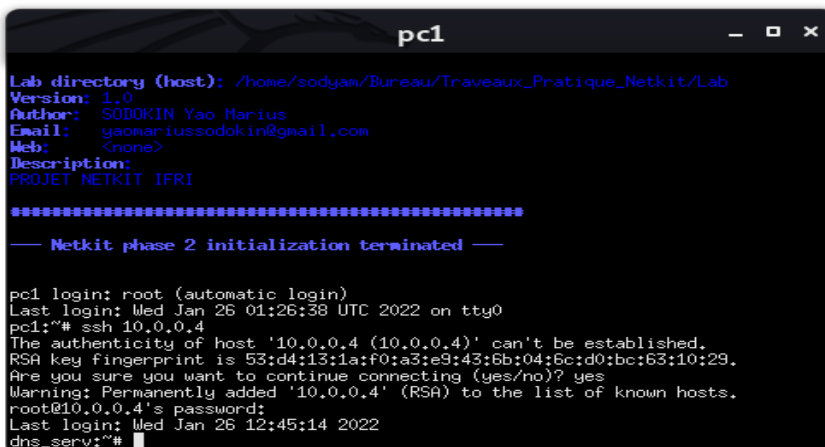
Après application de cette règle sur le firewall1, le PC1 du réseau local peut maintenant faire un telnet sur le serveur ssh. Montrons une capture d'écran



```
pc1
dns_serv:~# telnet 10.0.0.4
Trying 10.0.0.4...
Connected to 10.0.0.4.
Escape character is '^]'.
Debian GNU/Linux 5.0
dns_serv login:
Password:
```

de la connexion ssh.

- TEST des services (connexion du pc1 au serveur ssh du réseau DMZ)
Vérifiez si le trafic ssh a bien été envoyé au serveur ssh avec la commande



```
pc1
Lab directory (host): /home/sodgarn/Bureau/Travaux_Pratique_Netkit/Lab
Version: 1.0
Author: SODOKIN Yao Marius
Email: yaomariussodokin@gmail.com
Web: <none>
Description:
PROJET NETKIT IFRI

*****
— Netkit phase 2 initialization terminated —

pc1 login: root (automatic login)
Last login: Wed Jan 26 01:26:38 UTC 2022 on tty0
pc1:~# ssh 10.0.0.4
The authenticity of host '10.0.0.4 (10.0.0.4)' can't be established.
RSA key fingerprint is 53:d4:13:1a:f0:a3:e9:43:6b:04:6c:d0:bc:63:10:29.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.0.0.4' (RSA) to the list of known hosts.
root@10.0.0.4's password:
Last login: Wed Jan 26 12:45:14 2022
dns_serv:~#
```