# NAT

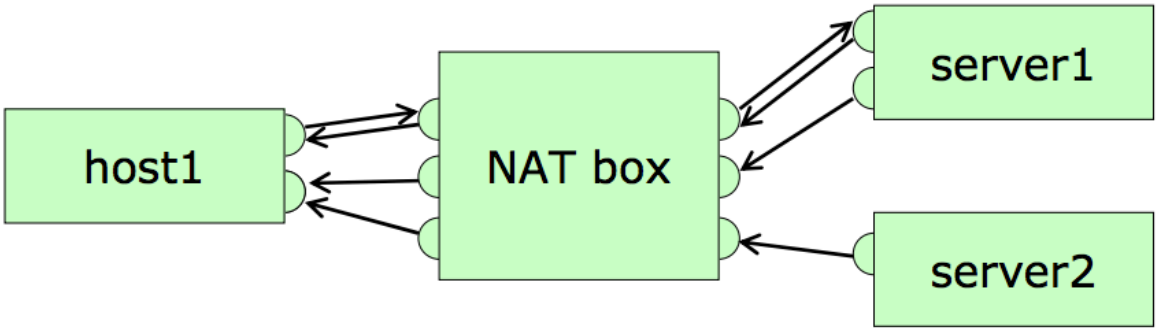
http://wiki.netkit.org/netkit-labs/netkit-labs_application-level/netkit-lab_nat/netkit-lab_nat.pdf

Le **RFC 3489** définit plusieurs typologies de NAT, notamment 4 :

1. full cone
2. (address) restricted cone
3. port restricted cone
4. symmetric

**Full cone nat** (défini dans le RFC 3489)

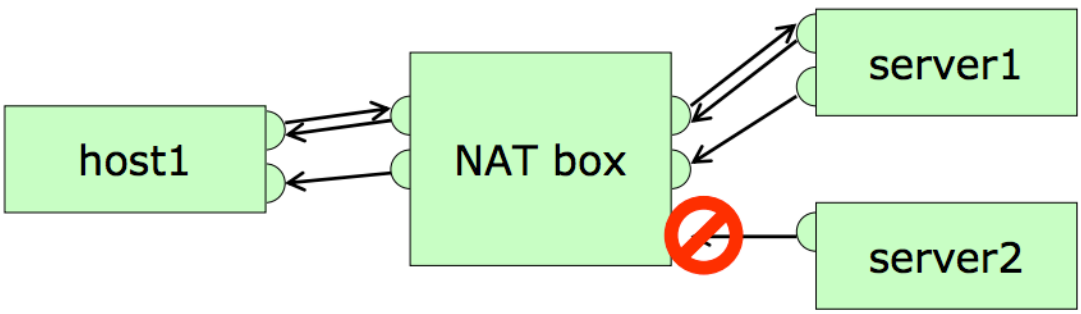
Les requêtes prévenantes du même (I_IP,I_port) interne sont "mappées" sur le même couple (E_IP,E_port) externe.




- N'importe quel Host externe peut envoyer un paquet à un host interne, en utilisant le mapping d'adresse interne/externe (privées/publique)

**(Address) restricted cone nat** (défini dans le RFC 3489)

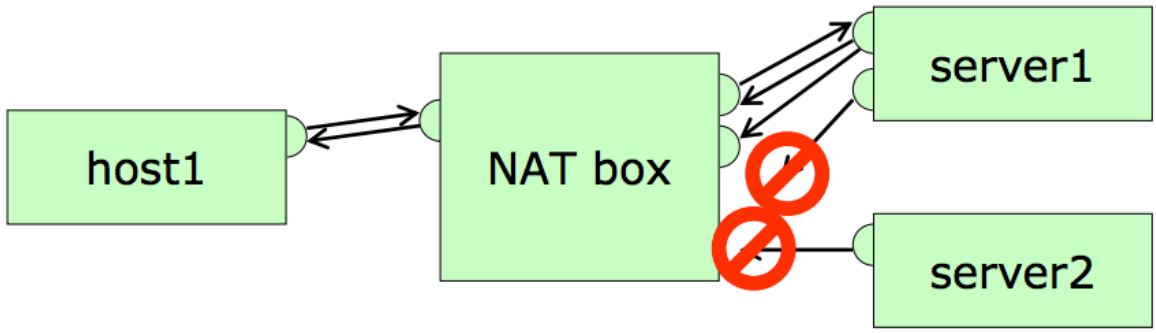
Les requêtes prévenantes du même (I_IP,I_port) interne sont "mappées" sur le même couple (E_IP,E_port) externe.




- Un Host externe (H_IP,*) peut envoyer un paquet à un host interne seulement si ce dernier a envoyé auparavant un paquet à (H_IP,*)

**Port restricted cone nat** (défini dans le RFC 3489)

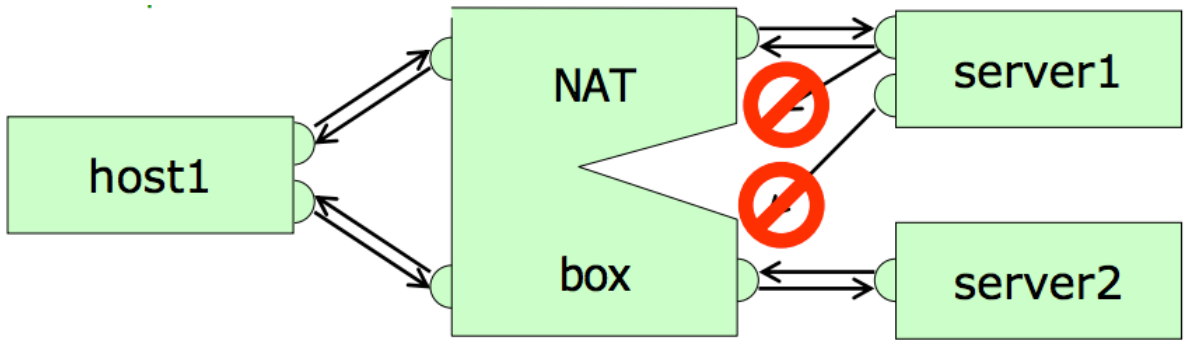
Les requêtes prévenantes du même (I_IP,I_port) interne sont “mappées” sur le même couple (E_IP,E_port) externe.




Un Host externe (H_IP,H_port) peut envoyer un paquet à un host interne seulement si ce dernier a envoyé auparavant un paquet à (H_IP,**H_port**)

**Symmetric NAT** (défini dans le RFC 3489)

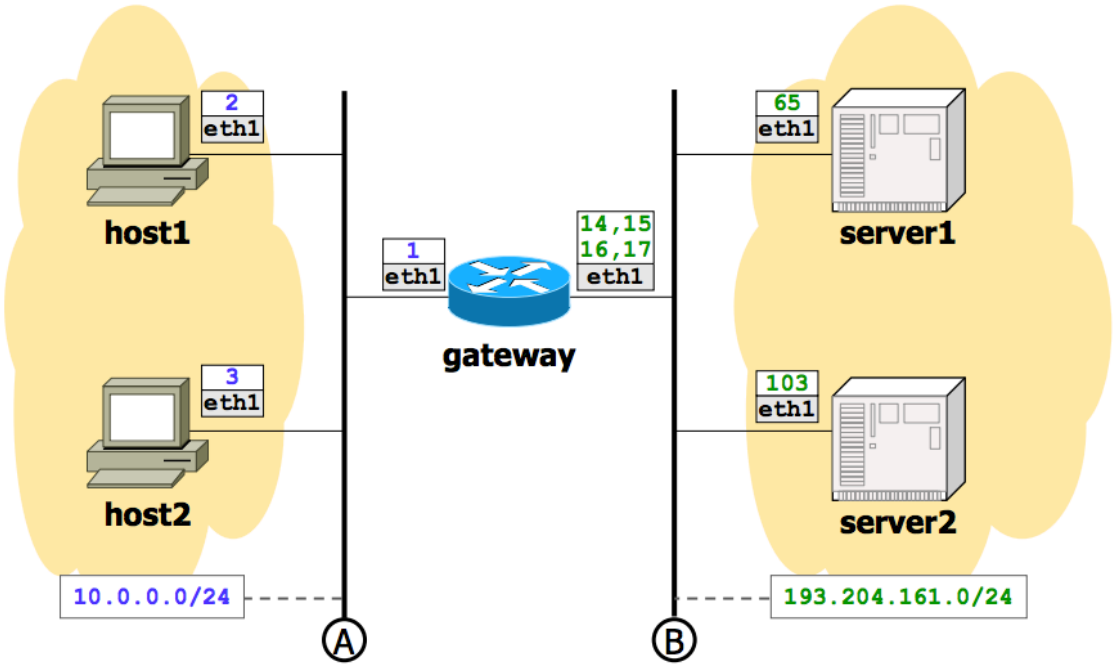
Des requêtes provenant du même host interne (I_IP,I_port) et pour une même destination (D_IP,D_port) sont mappées sur le même couple externe (E_IP,E_port)




- Paquets provenant du même (I_IP,I_port) mais avec destinations différentes sont mappés sur un couple (E_IP,E_port) différent.
- Seule des réponses peuvent être envoyées vers (I_IP,I_port)

## Laboratoire

Topologie du réseau (2 hosts, 1 gateway, 2 servers):




**Fichier de configuration du lab à télécharger ici :**
http://wiki.netkit.org/netkit-labs/netkit-labs_application-level/netkit-lab_nat/netkit-lab_nat.tar.gz

Au startup, la configuration (très simple) des 2 hosts est la suivante :

**host1.startup**

```
ifconfig eth0 10.0.0.2 netmask 255.255.255.0 up
route add default gw 10.0.0.1
```

**host2.startup**

```
ifconfig eth0 10.0.0.3 netmask 255.255.255.0 up
route add default gw 10.0.0.1
```

Pour les 2 serveurs, un packet sniffer est lancé automatiquement :

**server1.startup**

```
ifconfig eth0 193.204.161.65 up
touch /hostlab/$HOSTNAME.ready
cd /root
screen -c /root/screenrc
```

**server2.startup**

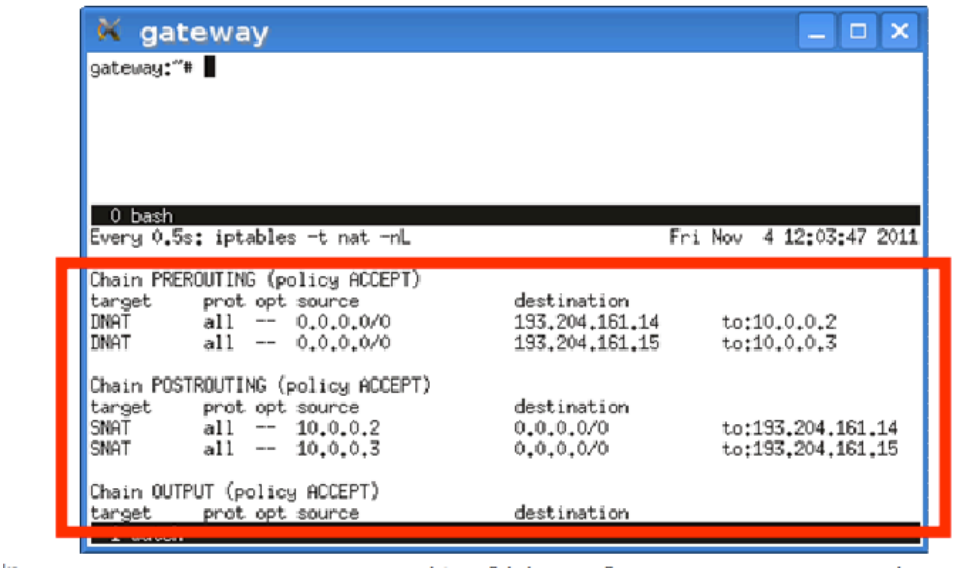
```
ifconfig eth0 193.204.161.103 up
touch /hostlab/$HOSTNAME.ready
cd /root
screen -c /root/screenrc
```

Le Gateway utilise la commande « iptables » pour implémenter les différents types de NAT. Par default, il implémente le « full cone NAT ». Le script « set_nat_type.sh » peut être utilisé pour choisir entre différents types de NAT :

```
gateway
root@gateway:/root# ./set_nat_type.sh
Usage: ./set_nat_type.sh nat_type

where nat_type is one of the following,
self-explaining, NAT types:
  f)ullcone
  r)estricted
  p)ortrestricted
  s)ymmetric
root@gateway:/root#
```

Les règles d'iptables qui implémentent le NAT sont visualisées en temps réel sur le Gateway :




L'interface "externe" du Gateway (eth1) a plusieurs adresses IP grace à l'utilisation d'alias. Ces adresses forment le "pool" d'adresses publiques disponibles pour faire le mapping entre adresse privées et publiques.

```
gateway
gateway:~# ifconfig  | grep -A 1 eth1
eth1      Link encap:Ethernet  HWaddr de:c3:bf:f8:7a:aa
          inet addr:193.204.161.14  Bcast:193.204.161.255  Mask:255.255.255.0
--
eth1:1    Link encap:Ethernet  HWaddr de:c3:bf:f8:7a:aa
          inet addr:193.204.161.15  Bcast:193.204.161.255  Mask:255.255.255.0
--
eth1:2    Link encap:Ethernet  HWaddr de:c3:bf:f8:7a:aa
          inet addr:193.204.161.16  Bcast:193.204.161.255  Mask:255.255.255.0
--
eth1:3    Link encap:Ethernet  HWaddr de:c3:bf:f8:7a:aa
          inet addr:193.204.161.17  Bcast:193.204.161.255  Mask:255.255.255.0
```

# Experiment 1 – Address Mapping

Nous commençons par expérimenter le mapping d'une adresse privée à une publique (pour l'instant nous ne considérons pas les port numbers)
- Allez sur **host1** et lancez un "ping" vers 193.204.161.65 (**server1**)
- Faites ensuite la même chose à partir de **host2**

En regardant le sniffer sur server1, nous pouvons voir que :
- host1 est "mappé" sur l'adresse IP externe 193.204.161.14
- host2 sur 193.204.161.15

Le gateway ne va pas, dans cette configuration, changer le port number. Donc, le trafic de (host1,Iport ) est toujours mappé en (193.204.161.14, Iport), et celui de (host2, Iport) en (193.204.161.15, Iport )
**- Vérifiez cela** en utilisant la commande suivante (netcat) :

nc 193.204.161.65 999 –p 100

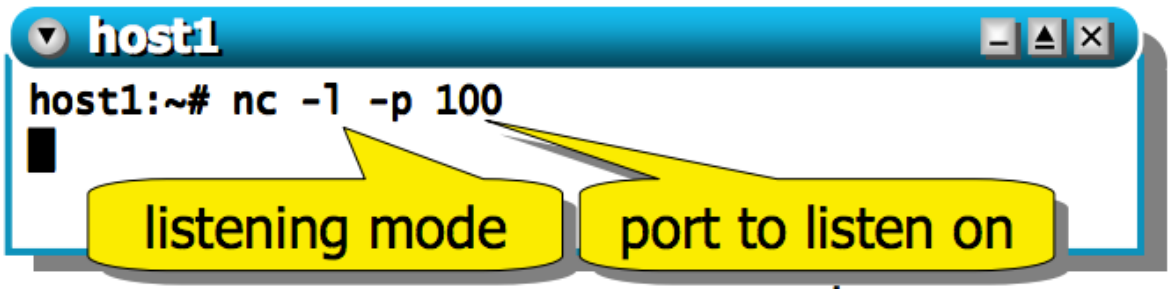
à partir de host1 et host2 (la commande ouvre une connexion TCP vers 193.204.161.65, port 999, avec source port 100). Vérifiez avec des numéros de ports différents.

En conclusion
- nous avons mis en place un mapping pour host1 et host2 vers des addresses IP publiques
- host1 et host2 sont toujours mappés vers la meme adresse
- les "port numbers" ne sont jamais changés

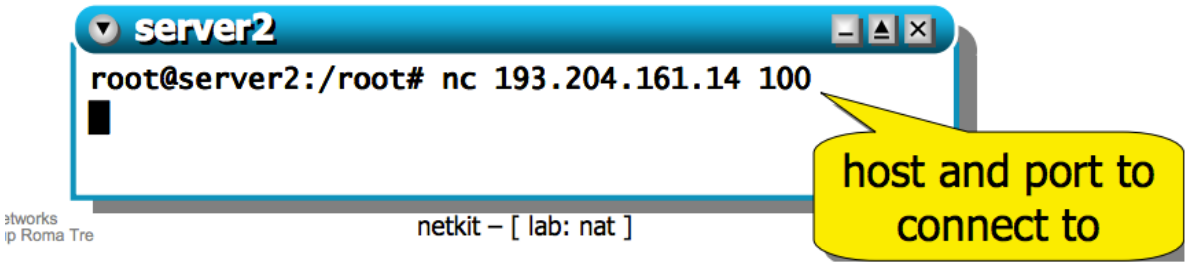
**Exercice** : faites les mêmes opérations pour les autres types de NAT, en utilisant le script set_nat_type.sh (à exception du "symmetric NAT"), et observez le résultat.

# Experiment 2 – Full Cone NAT

On revient maintenant à Full Cone NAT (toujours en utilisant set_nat_type.sh )
On utilise à nouveau netcat (nc) : cette commande permet des transferts stdin/stdout sur une connexion TCP.
Faitez demarrer « nc » en mode server sur host1




Sur server2, utilisez donc "nc" pour vous connecter au TCP server on host1




Tapez le texte que vous souhaitez sur le terminal de **server2**, et vous allez le voir apparaître sur le terminal de **host1.**

**Conclusion et exercises :**

- server2 peut envoyer du trafic à host1 même si host1 n'a jamais envoyé du trafic à server2 auparavant.
- **Faites les mêmes opérations** avec « nc » de server2 à host2, de server1 à host1, et de server1 à host2, en changeant aussi les port numbers

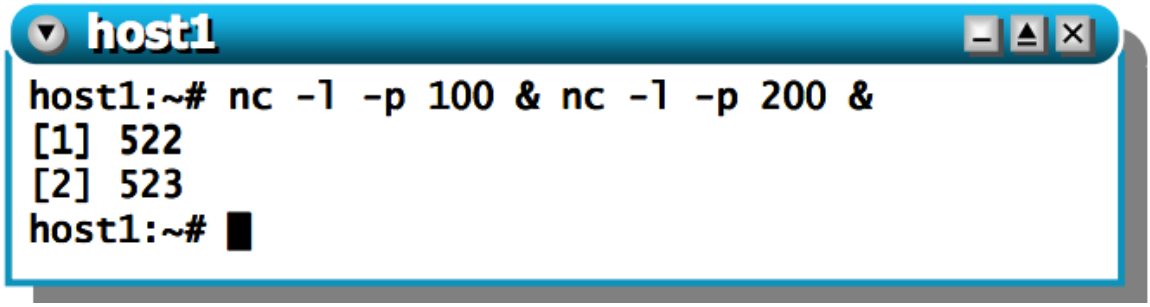
## Experiment 3 – restricted

Tout d'abord, changez sur le Gateway le type de NAT en "restricted" :
./set_nat_type.sh restricted

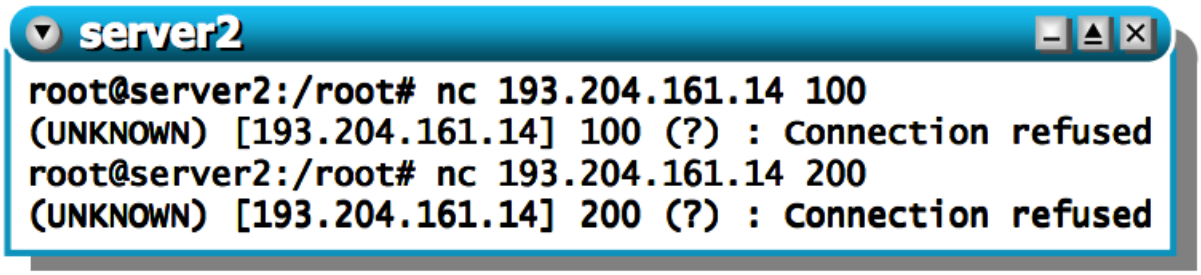
Vous devrez pouvoir observer les règles d'iptables sur le Gateway changer.

Il faut noter que le *(port) restricted nat* est difficile à implémenter en Linux. Dans ce Lab, le (port) restricted nat est implémenté de la façon suivante : les règles iptables sont ajoutées automatiquement en fonction du trafic qui passe à travers le Gateway, et ensuite elles "expirent" (sont effacées) après 30 secondes.

- Faites donc démarrer deux instances de nc sur host1, chacune en écoute sur une porte différente :




- A partir du server2, essayez de vous connecter à une des instances de nc sur host1

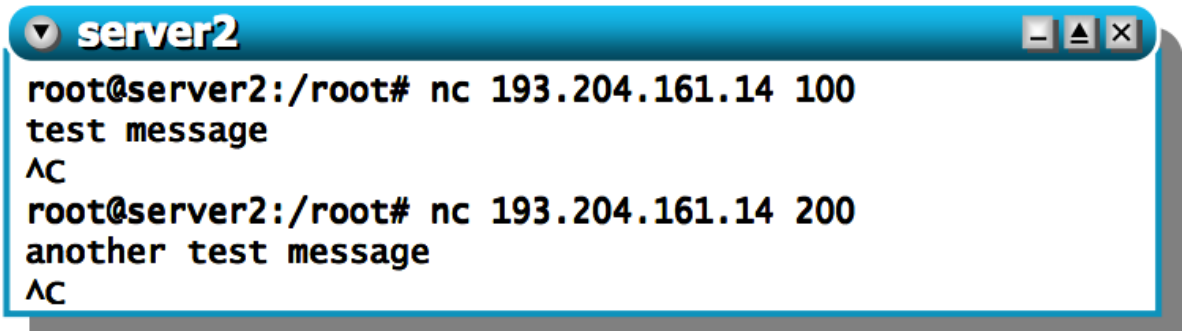



La connexion est refusée, correctement, jusqu'à ce que host1 envoie du trafic TCP vers server2.
Essayons d'envoyer du trafic de host1 vers **server1** :

host1:~# nc 193.204.161.65 999
**Résultat :** (UNKNOWN) [193.204.161.65] 999 (?) : Connection refused

**La connexion de** server2 à host1 ne fonctionne pas encore
- Mais envoyons maintenant du trafic de **host1** vers **server2** :
**host1**:~# nc 193.204.161.103 999

Maintenant, envoyer du trafic du server2 (n'importe quelle porte) vers n'importe quelle porte de host1 devient possible, et les messages sont délivrés directement par nc :




Mais après 30 secondes, il est à nouveau impossible d'initier une connexion de server2 à host1.

## Conclusion & exercises

- Le trafic peut être envoyé d'un host externe vers un interne seulement si l'host interne a envoyé au moins un paquet à l'host externe auparavant.
- **Essayez de faire des tests** avec différents couples source-destination, par rapport à celles considérées ci-dessus, et en changeant les port numbers.

# Experiment 4 – port restricted

Sur le Gateway, tout d'abord, il faut lancer le script :
gateway:~# ./set_nat_type.sh portrestricted
Les règles d'iptable changent.

Faites démarrer nc sur host1 (en background) :
host1:~# nc - host1:~# nc -l -p 100

Essayez de le contacter à partir de server2 :
root@server2:/root# nc 193.204.161.14 100
**Qu'est-ce que vous observez ?**

Envoyez maintenant du trafic TCP avec nc de host1 à server 2 et essayez à nouveau :
host1:~# nc 193.204.161.103 999
**Qu'est-ce que vous observez ?**

Rappelez-vous que ce type de NAT est en train d'appliquer une restriction aussi sur le numéro de porte :
- si host1 envoie un paquet au server2 sur la porte 999, alors server2 est autorisé à établir une connexion avec host1 seulement en utilisant comme source port la 999.
Essayez à nouveau :

**Sur Host1** : host1:~# nc 193.204.161.103 999
**Sur server2 :** root@server2:/root# nc 193.204.161.14 100 - root@server2:/root# nc 193.204.161.14 100 -p 999
Il devrait fonctionner maintenant.

**Conclusion & exercises**
- un host externe peut établir et initier une connexion vers un host interne seulement s'il a été contacté auparavant par ce même host interne, et seulement en utilisant le même numéro de port (comme *source port*) sur lequel il a été contacté.
- Essayez à nouveau avec d'autres combinaisons de hosts source/destination et de numéro de ports
- Qu'est-ce qu'il se passe si le server2 essaie de se connecter à une autre porte sur host1 (utilisant toujours 999 comme source port) ?

# Experiment 5 – symmetric NAT

- Nous allons vérifier que les paquets du même source host vers des destinations différentes sont "mappés" à des adresses publiques différentes.
- Dans host1 envoyez un ping 193.204.161.65 (vers server1, donc), et ensuite vers 193.204.161.103 (server2)
- faites la même chose à partir de host2
- regardez le packet sniffer dans server1 et server2. On peut observer que :
--- host1 est mappé en 193.204.161.14 quand il contacte server1
--- host1 est mappé en 193.204.161.15 quand il contacte server2
--- host2 est mappé en 193.204.161.16 quand il contacte server1

--- host2 est mappé en 193.204.161.17 quand il contacte server2
- Cela implémente la “première moitié” du symmetric NAT

- La deuxième moitié du comportement du symmetric NAT est de refuser du trafic (“unsolicited traffic”) qui entre dans le réseau privé (par exemple, des réponses à des paquets envoyés par des host du réseau privé).
- Pour verifier cela :
--- utilisez nc pour essayer de créer une connexion de host1 à server1 (cela devrait fonctionner)
--- utilisez nc pour essayer de créer une connexion de server1 à host1

**Conclusion & exercises**
- il est impossible d’établir (initier) une connexion de l’extérieur vers un host interne, parce que le trafic externe “unsolicited” est toujours rejeté.
- Essayez de tester les différentes combinaisons de connexion entre hosts et servers.