Sommaire

INTRODUCTION

Objectifs

Être capable d'afficher et d'exploiter les statistiques réseaux d'une machine.

Être capable de mettre en oeuvre une surveillance de l'activité réseau de sa machine.

netstat

La commande netstat (*network statistics*) est une commande en ligne affichant des informations sur les connexions réseau, les tables de routage et un certain nombre de statistiques. La commande est disponible sous Unix/Linux et sous Windows.

En savoir plus :

```
$ netstat --help
$ man netstat   # Sous Unix/Linux

C:\>netstat /?  # Sous Windows
```

[Source : http://fr.wikipedia.org/wiki/Netstat]

Séquence 1 : affichage

Statistiques sur les interfaces

```
$ netstat -i
Table d'interfaces noyau
Iface MTU  Met RX-OK RX-ERR RX-DRP RX-OVR TX-OK TX-ERR TX-DRP TX-OVR Flg
eth1  1500   0 14759173   0      0      0 17579724  0      0      0 BMRU
lo    16436  0   827750   0      0      0   827750  0      0      0 LRU
```

Remarque : cette option n'est pas disponible sous Windows

1) Décoder les flags (colonne Flg) de la commande précédente.

Statistiques par protocole

Par défaut, les statistiques sont affichées pour IP, IPv6, ICMP, ICMPv6, TCP, TCPv6, UDP et UDPv6.

```
$ netstat -s | less
$ netstat -s
Ip:
    2302343 total packets received
    0 forwarded
    0 incoming packets discarded
    2302277 incoming packets delivered
    2151721 requests sent out
    47 outgoing packets dropped
    42 reassemblies required
    21 packets reassembled ok
    1599 fragments received ok
    47 fragments failed
    3198 fragments created
Icmp:
    5534 ICMP messages received
...
IcmpMsg:
        InType0: 6
        InType3: 5500
        InType8: 7
        InType11: 21
        OutType0: 7
        OutType3: 12678
        OutType8: 23
Tcp:
    34098 active connections openings
...
```

```
Udp:
    124518 packets received
    8626 packets to unknown port received.
    5 packet receive errors
    127800 packets sent
...
UdpLite:
TcpExt:
    1622 invalid SYN cookies received
...
IpExt:
    InMcastPkts: 6724
...
```

On peut choisir son protocole : [--tcp|-t] [--udp|-u] [--raw|-w] [--inet|--ip] etc ...

```
$ netstat -t -s | less
$ netstat -u -s | less
$ netstat -w -s | less
$ netstat --ip -s
```

2) À partir des statistiques de la couche IP, déterminer si votre machine est un routeur ?

3) Déterminer combien de « ping » ont été émis à partir de votre machine ?

4) Votre machine utilise-t-elle beaucoup la fragmentation IP ?

5) Exécuter la commande suivante et déterminer quelle statistique est affectée par son exécution ?

```
$ ping www.ibm.fr -s 1500 -M do -c 1
```

6) Qu'est-ce qu'un SYN cookies affiché dans la couche TCP ?

Statistiques sur les sockets actives et passives

Le paramètre -a affiche toutes les sockets, y compris les sockets d'écoute des serveurs. On peut spécifier un protocole : [--tcp|-t] [--udp|-u] etc ...

```
$ netstat -ta
Connexions Internet actives (serveurs et établies)
Proto Recv-Q Send-Q Local Address    Foreign Address         State
tcp        0      0 *:sunrpc         *:*                     LISTEN
...
tcp        0      0 127.0.0.1:57246  127.0.0.1:5000          ESTABLISHED
```

On peut désactiver la résolution de noms avec l'option -n :

```
$ netstat -nta
Connexions Internet actives (serveurs et établies)
Proto Recv-Q Send-Q Local Address    Foreign Address         State
tcp        0      0 0.0.0.0:111      0.0.0.0:*               LISTEN
...
```

Pour UDP :

```
$ netstat -ua
Connexions Internet actives (serveurs et établies)
Proto Recv-Q Send-Q Local Address   Foreign Address         State
udp        0      0 *:sunrpc        *:*
...
udp        0      0 0.0.0.0:5000    0.0.0.0:*
```

La colonne State donne l'état de la socket. Puisqu'il n'y a pas d'état dans le mode RAW et généralement pas d'état utilisé en UDP, cette colonne peut rester vierge. Cette colonne est particulièrement intéressante en TCP.

Normalement, on trouvera une des valeur suivante :

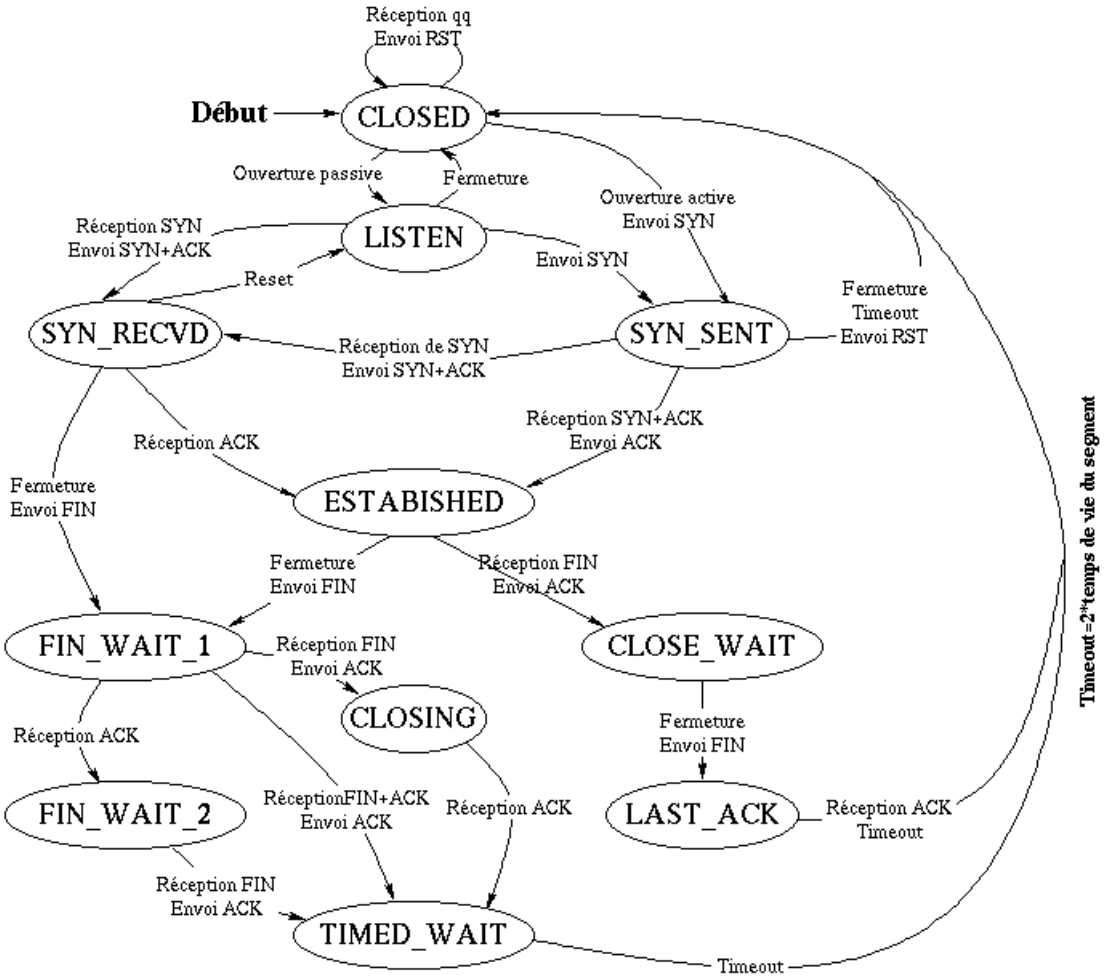
- ESTABLISHED : La socket a une connexion établie.
- SYN_SENT : La socket attend activement d'établir une connexion.
- SYN_RECV : Une requête de connexion a été reçue du réseau.
- FIN_WAIT1 : La socket est fermée, et la connexion est en cours de terminaison.
- FIN_WAIT2 : La connexion est fermée, et la socket attend une terminaison du distant.
- TIME_WAIT : La socket attend le traitement de tous les paquets encore sur le réseau avant d'entreprendre la fermeture.
- CLOSED : La socket n'est pas utilisée.
- CLOSE_WAIT : Le distant a arrêté, attendant la fermeture de la socket.
- LAST_ACK : Le distant termine, et la socket est fermée. Attente d'acquittement.
- LISTEN : La socket est à l'écoute de connexions entrantes. Ces sockets ne sont affichées que si le paramètre -a,--listening est fourni.
- CLOSING : Les deux prises sont arrêtées mais toutes les données locales n'ont pas encore été envoyées.
- UNKNOWN : L'état de la prise est inconnu.

En TCP, les états d'une socket proviennent des 6 bits du champ Flag de l'entête TCP :

- URG (Urgent Pointer) : contient des données urgentes
- ACK (Acknowledge Field) : acquittement
- PSH (Push Flag) : passer immédiatement les données à la couche application
- RST (Reset Flag) : forcer la clôture d'une connexion après une erreur irrécupérable.
- SYN (Synchronize Flag) : synchroniser le démarrage d'une connexion
- FIN : pour terminer une connexion

Une socket est toujours créée dans un état fermé (CLOSED). Puis elle passera soit en **« ouverture passive » pour un serveur** (état LISTEN ou en écoute de demande connexion) soit en **« ouverture active » pour un client** qui initie une demande de synchronisation de numéro de séquence (envoi SYN) que l'on nomme aussi « demande de connexion ».

Ce qui donne le schéma suivant pour les différents états d'une socket :



Remarque :

Il est normal d'avoir des sockets dans l'état TIME_ WAIT pour une longue période de temps. Le temps est spécifiée dans la RFC793 comme deux fois le maximum segment life (MSL) time. Le MSL de Microsoft est défini comme étant de 2 minutes. Par conséquent, une socket peut être dans un état TIME_ WAIT aussi longtemps que 4 minutes. Certains systèmes mettent en oeuvre des valeurs différentes. Sous Unix, la valeur recommandée est aussi de 2 minutes (120 s).

7) Démarrer un serveur TCP sur le port 5000 (avec netcat) et observer dans quel état se trouve la socket ?

8) Connecter un client TCP (avec telnet ou netcat) vers le port du serveur et observer les différents états avec la commande suivante. Commenter.

```
$ netstat -nta | grep -E ":5000|Address"
```

9) Mettre fin au client et observer à nouveau les états. Commenter.

10) Démarrer un serveur TCP multiclients sur le port 5000 (avec netcat) et un serveur UDP multiclients sur le port 5000. Donner les commandes exactes.

11) Observer les états des sockets TCP et UDP sur le port 5000 ? Puis, connecter plusieurs clients (TCP et UDP) et observer à nouveau.

12) Mettre fin au client UDP et observer les états des sockets pour UDP. Pourquoi la socket UDP ne passe-t-elle pas à l'état TIME_WAIT et reste dans l'état ESTABLISHED ? Que se passe-t-il si on met fin au serveur UDP ? Commenter.

Séquence 2 : exploitation

Introduction

Il est peut-être utile ici de rappeler la « philosophie UNIX ». Résumer la philosophie d'UNIX n'est pas chose évidente. Il s'agit d'un ensemble de principes. Nombreux sont ceux qui ont essayé de les résumer ou les lister (taper « philisophie UNIX » ou « less is more » dans un moteur de recherche). En voici quatre qui forment la base de cette séquence :

Des programmes qui effectuent une seule chose et qui le font bien

Voilà la base de toutes choses dans le monde UNIX (« dans le monde » tout court peut-être également).

Le silence est d'or

En d'autres termes, lorsqu'un programme n'a rien à dire, il doit garder le silence. Ce n'est que lorsqu'il y a un problème qu'un outil doit devenir bavard et signaler explicitement une erreur. Un programme qui fait ce qu'on lui demande n'affiche rien, ne signale rien. C'est le cas de la plupart des outils de base en ligne de commande.

Des programmes qui collaborent

Si tous les programmes ne font, chacun, qu'une chose et qu'ils la font bien, ceci implique qu'ils doivent alors fonctionner de concert pour pouvoir achever des tâches plus importantes. Ces « briques » doivent alors collaborer les unes avec les autres du mieux possible. Le but est de former un système complet où la somme des parties est supérieure à l'ensemble. Lorsqu'on dispose d'un ensemble de briques fiables, il est possible de construire un mur solide.

Des programmes pour gérer des flux de texte

Les flux de texte représentent une interface universelle (la seule ?). La notion de flux de texte est véritablement caractéristique des UNIX.

Surveiller sa machine

Voici quelques commandes qui permettent d'exploiter les statistiques fournies par netstat afin de surveiller l'activité réseau de sa machine. Certaines de ces commandes, après analyse, permettront de répondre à cette fameuse question « Suis-je attaqué ? » ...

Compter les états des sockets passives et actives

```
$ netstat -nat | awk '{print $6}' | sort | uniq -c | sort -n
      1 CLOSE_WAIT
      1 établies)
      1 FIN_WAIT1
      1 Foreign
      1 SYN_SENT
     11 LISTEN
     12 TIME_WAIT
     56 ESTABLISHED
```

Compter les états des sockets pour une adresse IP en particulier

```
$ netstat -nat |grep {IP-address} | awk '{print $6}' | sort | uniq -c |
sort -n
```

Afficher la liste des adresses IP connectés

```
$ netstat -nat | sed 1d | sed 1d | awk '{ print $5}' | cut -d: -f1 | sed
-e '/^$/d' | uniq
```

Afficher le nombre d'adresses IP connectés

```
$ netstat -nat | sed 1d | sed 1d | awk '{ print $5}' | cut -d: -f1 | sed
-e '/^$/d' | uniq | wc -l
```

Compter le nombre de sockets connectés par adresse IP

```
$ netstat -atun | sed 1d | sed 1d | awk '{print $5}' | cut -d: -f1 | sed
-e '/^$/d' |sort | uniq -c | sort -n

$ netstat -atun | awk '{print $5}' | sed -n -e '/[0-9]\{1,3\}\.[0-9]\
{1,3\}\.[0-9]\{1,3\}\.[0-9]\{1,3\}/p' | sed 's/::ffff://' | cut -d: -f1
| sort | uniq -c | sort -n
```

Compter le nombre de sockets dans l'état ESTABLISHED par adresse IP

```
$ netstat -atun | grep ESTABLISHED | awk '{print $5}' | cut -d: -f1 |
sed -e '/^$/d' |sort | uniq -c | sort -n
```

Et pour finir avec netstat, des options intéressantes :

- -l, --listening display listening server sockets
- -p, --programs display PID/Program name for sockets
- -e, --extend display other/more information

```
$ netstat -tulpn
$ netstat -npl
$ netstat -antpuew
```

Il existe deux autres commandes qui sont généralement utilisés dans la surveillance système :

- **lsof** : lister les fichiers ouverts (ne pas oublier que sous Unix, TOUT est FICHIER !)
- **fuser** : identifier les processus qui utilisent des fichiers ou des sockets

Remarque : les commandes finger, w et who peuvent être aussi de bons compléments.

Afficher les processus ayant une activité réseau

```
# lsof -Pnl +M -i4
COMMAND     PID     USER   FD   TYPE   DEVICE SIZE/OFF NODE NAME
cupsd      1270        0    6u  IPv4     8867      0t0  TCP *:631
(LISTEN)
cupsd      1270        0    9u  IPv4     8871      0t0  UDP *:631
...
nmbd       2383        0   10u  IPv4    11175      0t0  UDP *:137
nmbd       2383        0   11u  IPv4    11176      0t0  UDP *:138
nmbd       2383        0   12u  IPv4    11178      0t0  UDP
192.168.52.2:137
nmbd       2383        0   13u  IPv4    11179      0t0  UDP
192.168.52.255:137
nmbd       2383        0   14u  IPv4    11180      0t0  UDP
192.168.52.2:138
nmbd       2383        0   15u  IPv4    11181      0t0  UDP
192.168.52.255:138
firefox   17822      500   25u  IPv4 14346042      0t0  TCP
192.168.52.2:60559->75.126.153.206:80 (ESTABLISHED)
```

Afficher les fichiers ouverts par un processus

```
# lsof -p PID
```

Afficher les fichiers dont l'adresse Internet correspond à l'adresse spécifiée

Adresse sous la forme : [46][protocole][@nom_hôte|adresse_hôte][:service|port]

```
# lsof -i@192.168.52.2
```

Afficher la liste des PID utilisant un service|port/protocole

```
# fuser http/tcp
# fuser 80/tcp
```

Chercher des sockets IPv4 seulement (-4,--ipv4)

Syntaxe : noms udp/tcp: [port_local][,[hôte_distant][,[port_distant]]]

```
# fuser -4 -v -n tcp ,
```

13) Démarrer un serveur apache sur votre machine

Vérifier d'abord si celui-ci n'est pas déjà démarré

```
# /etc/init.d/httpd status
httpd (pid  2870) est en cours d'exécution...
ou
# apachectl status
```

Sinon, on le démarre :

```
# /etc/init.d/httpd start
ou
# apachectl start
```

14) Déterminer le(s) PID(s) du serveur apache. Vous utiliserez la commande fuser (voir aussi pidof)

15) Afficher puis compter le(s) fichier(s) ouvert(s) par un des processus du serveur apache. Vous utiliserez la commande lsof (voir aussi wc pour le comptage)

16) Afficher seulement les sockets connectés sur votre serveur apache (mettre en place une situation favorable : il vous faut des clients !)

Remarque : osez utiliser grep pour filtrer les affichages qui vous intéressent

17) Compter le nombre de sockets connectés par adresse IP sur votre serveur apache

18) Compter les états des sockets ouvertes vers votre serveur apache (essayer plusieurs fois pour observer les différents états)