



Gestion et Surveillance de Réseau

Analyse du réseau local



These materials are licensed under the Creative Commons *Attribution-Noncommercial 3.0 Unported* license
(<http://creativecommons.org/licenses/by-nc/3.0/>).

Analyse du réseau

Comme nous le savons...

Avant d'accuser le réseau, nous devons vérifier si le problème ne vient pas de nous :

- **Qu'est-ce qui peut être fautif au niveau local**
 - Problèmes matériels
 - Charge excessive (processeur, mémoire, E/S)
- **Qu'est-ce qui est considéré “normal” ?**
 - Utilisez régulièrement des outils d'analyse
 - Familiarisez-vous avec les valeurs et états normaux de votre machine (“établissement des références”).
 - **Il est important de maintenir l'historique**
 - Agents SNMP et bases de données

Analyse locale

Trois grandes catégories :

- Processus
 - Processus en cours d'exécution (actif)
 - Processus en attente (inactif)
 - attendant leur tour
 - bloqués
- Mémoire
 - Réelle
 - Virtuelle
- E/S (Input/Output)
 - Stockage
 - Réseau

Indicateurs clés

Processeur insuffisant

- Nombre de processus en attente d'exécution toujours élevé
- Utilisation élevée du processeur (charge moyenne)

Mémoire insuffisante

- Très peu de mémoire disponible
- Beaucoup d'activités de permutation (swapping)

E/S lentes

- Beaucoup de processus bloqués
- Nombre élevé de transferts de blocs

Analyse locale

Heureusement, sous Unix, il y a une douzaine d'outils utiles fournissant beaucoup d'informations sur notre machine

Parmi les plus connus :

- vmstat
- top
- lsof
- netstat
- tcpdump
- wireshark (ethereal)
- iptraf
- iperf

vmstat

- Affiche des rapports périodiques sur les processus, la mémoire, la pagination, les E/S, l'état du processeur, etc.
- `vmstat <-options> <delay> <count>`

```
# vmstat 2
procs -----memory----- ---swap-- -----io----- --system-- ----cpu----
 r  b   swpd   free   buff   cache    si    so    bi    bo    in     cs us sy id wa
 2  0  209648  25552  571332  2804876     0     0     3     4     3      3 15 11 73  0
 2  0  209648  24680  571332  2804900     0     0     0    444   273  79356 16 16 68  0
 1  0  209648  25216  571336  2804904     0     0     6   1234   439  46735 16 10 74  0
 1  0  209648  25212  571336  2804904     0     0     0     22   159 100282 17 21 62  0
 2  0  209648  25196  571348  2804912     0     0     0    500   270  82455 14 18 68  0
 1  0  209648  25192  571348  2804912     0     0     0    272   243  77480 16 15 69  0
 2  0  209648  25880  571360  2804916     0     0     0    444   255  83619 16 14 69  0
 2  0  209648  25872  571360  2804920     0     0     0    178   220  90521 16 18 66  0
```

top

- Outil d'analyse de performances standard pour les environnements Unix/Linux
- Affiche périodiquement une liste de statistiques de performances système :
 - Utilisation du processeur
 - Utilisation de la mémoire RAM et SWAP
 - Charges moyennes (Utilisation du processeur)
 - Information par processus

top (suite)

- **Information par processus (colonnes les plus intéressantes)**

- PID : ID du processus
- USER : utilisateur (propriétaire) du processus
- %CPU : pourcentage d'utilisation du processeur par le processus depuis le dernier rapport
- %MEM : pourcentage de mémoire physique (RAM) utilisée par le processus
- TIME : temps processeur total utilisé par le processus depuis son démarrage

Charge moyenne

Nombre moyen de processus actifs au cours des 1, 5 et 15 dernières minutes

- Une mesure simple mais très utile
- En fonction de la machine, les valeurs considérées normales peuvent varier :
 - Les machines multiprocesseurs peuvent gérer plus de processus actifs par unité de temps (qu'une machine monoprocesseur)

top

Quelques commandes clavier *interactive* utiles pour *top*

- **f** : Ajoute ou supprime des colonnes
- **F** : Spécifie sur quelle colonne s'effectue le tri
- **<** , **>** : Déplace la colonne sur laquelle le tri s'effectue
- **u** : Spécifie un utilisateur
- **k** : Spécifie le processus à tuer (arrêter)
- **d** , **s** : Change l'intervalle de mise à jour de l'affichage

netstat

Affiche des informations sur :

- Les connexions réseau
- Les tables de routage
- Les statistiques des interfaces (NIC)
- Les membres de groupes multicast

netstat

Quelques options utiles

- n**: Affiche les adresses, ports et ID utilisateurs en format numérique
- r**: Table de routage
- s**: Statistiques par protocole
- i**: Statut des interfaces
- l**: Sockets à l'écoute
- tcp, --udp**: Spécifie le protocole
- A**: Famille d'adresse [inet | inet6 | unix | etc.]
- p**: Affiche le nom de chaque processus pour chaque port
- c**: Affiche les sorties/résultats en continu

netstat

Exemples (ci-après)

```
# netstat -anr
```

```
Kernel IP routing table
```

Destination	Gateway	Genmask	Flags	MSS	Window	irtt	Iface
192.168.5.128	0.0.0.0	255.255.255.128	U	0	0	0	eth0
0.0.0.0	192.168.5.129	0.0.0.0	UG	0	0	0	eth0

```
# netstat -o -t
```

```
Active Internet connections (w/o servers)
```

Proto	Recv-Q	Send-Q	Local Address	Foreign Address	State	Timer
tcp	0	0	192.168.5.135:ssh	192.168.3.124:34155	ESTABLISHED	
keepalive (6754.95/0/0)						

```
# netstat -atv
```

```
Active Internet connections (servers and established)
```

Proto	Recv-Q	Send-Q	Local Address	Foreign Address	State
tcp	0	0	*:ssh	*:*	LISTEN
tcp	0	0	192.168.5.135:ssh	192.168.3.124:34155	ESTABLISHED
tcp6	0	0	:::ssh	:::*	LISTEN

netstat

Examples :

```
# netstat -n --tcp -c
```

```
Active Internet connections (w/o servers) ^
```

Proto	Recv-Q	Send-Q	Local Address	Foreign Address	State
tcp	0	272	::ffff:192.188.51.40:22	::ffff:128.223.60.27:60968	ESTABLISHED
tcp	0	0	::ffff:192.188.51.40:22	::ffff:128.223.60.27:53219	ESTABLISHED

```
# netstat -lnp --tcp
```

```
Active Internet connections (only servers) ^
```

Proto	Recv-Q	Send-Q	Local Address	Foreign Address	State	PID/Program name
tcp	0	0	0.0.0.0:199	0.0.0.0:*	LISTEN	11645/snmpd
tcp	0	0	0.0.0.0:3306	0.0.0.0:*	LISTEN	1997/mysqld

```
# netstat -ic
```

```
Kernel Interface table
```

Iface	MTU	Met	RX-OK	RX-ERR	RX-DRP	RX-OVR	TX-OK	TX-ERR	TX-DRP	TX-OVR	Flg
eth0	1500	0	2155901	0	0	0	339116	0	0	0	BMRU
lo	16436	0	18200	0	0	0	18200	0	0	0	LRU
eth0	1500	0	2155905	0	0	0	339117	0	0	0	BMRU
lo	16436	0	18200	0	0	0	18200	0	0	0	LRU
eth0	1500	0	2155907	0	0	0	339120	0	0	0	BMRU
lo	16436	0	18200	0	0	0	18200	0	0	0	LRU
eth0	1500	0	2155910	0	0	0	339122	0	0	0	BMRU
lo	16436	0	18200	0	0	0	18200	0	0	0	LRU
eth0	1500	0	2155913	0	0	0	339124	0	0	0	BMRU

netstat (suite)

Exemples :

```
# netstat -tcp -listening --program
```

```
Active Internet connections (only servers)
```

Proto	Recv-Q	Send-Q	Local Address	Foreign Address	State	PID/Program name
tcp	0	0	*:5001	*:*	LISTEN	13598/iperf
tcp	0	0	localhost:mysql	*:*	LISTEN	5586/mysqld
tcp	0	0	*:www	*:*	LISTEN	7246/apache2
tcp	0	0	t60-2.local:domain	*:*	LISTEN	5378/named
tcp	0	0	t60-2.local:domain	*:*	LISTEN	5378/named
tcp	0	0	t60-2.local:domain	*:*	LISTEN	5378/named
tcp	0	0	localhost:domain	*:*	LISTEN	5378/named
tcp	0	0	localhost:ipp	*:*	LISTEN	5522/cupsd
tcp	0	0	localhost:smtp	*:*	LISTEN	6772/exim4
tcp	0	0	localhost:953	*:*	LISTEN	5378/named
tcp	0	0	*:https	*:*	LISTEN	7246/apache2
tcp6	0	0	:::ftp	:::*	LISTEN	7185/proftpd
tcp6	0	0	:::domain	:::*	LISTEN	5378/named
tcp6	0	0	:::ssh	:::*	LISTEN	5427/sshd
tcp6	0	0	:::3000	:::*	LISTEN	17644/ntop
tcp6	0	0	ip6-localhost:953	:::*	LISTEN	5378/named
tcp6	0	0	:::3005	:::*	LISTEN	17644/ntop

netstat (suite)

```
$ sudo netstat -atup
```

Active Internet connections (servers and established) (if run as root PID/Program name is included)

Proto	Recv-Q	Send-Q	Local Address	Foreign Address	State	PID/Program name
tcp	0	0	*:35586	*:*	LISTEN	2540/ekpd
tcp	0	0	localhost:mysql	*:*	LISTEN	2776/mysqld
tcp	0	0	*:www	*:*	LISTEN	14743/apache2
tcp	0	0	d229-231.uoregon:domain	*:*	LISTEN	2616/named
tcp	0	0	*:ftp	*:*	LISTEN	3408/vsftpd
tcp	0	0	localhost:domain	*:*	LISTEN	2616/named
tcp	0	0	*:ssh	*:*	LISTEN	2675/sshd
tcp	0	0	localhost:ipp	*:*	LISTEN	3853/cupsd
tcp	0	0	localhost:smtp	*:*	LISTEN	3225/exim4
tcp	0	0	localhost:953	*:*	LISTEN	2616/named
tcp	0	0	*:https	*:*	LISTEN	14743/apache2
tcp6	0	0	[::]:domain	[::]:*	LISTEN	2616/named
tcp6	0	0	[::]:ssh	[::]:*	LISTEN	2675/sshd
tcp6	0	0	ip6-localhost:953	[::]:*	LISTEN	2616/named
udp	0	0	*:50842	*:*		3828/avahi-daemon:
udp	0	0	localhost:snmp	*:*		3368/snmpd
udp	0	0	d229-231.uoregon:domain	*:*		2616/named
udp	0	0	localhost:domain	*:*		2616/named
udp	0	0	*:bootpc	*:*		13237/dhclient
udp	0	0	*:mdns	*:*		3828/avahi-daemon:
udp	0	0	d229-231.uoregon.ed:ntp	*:*		3555/ntpd
udp	0	0	localhost:ntp	*:*		3555/ntpd
udp	0	0	*:ntp	*:*		3555/ntpd
udp6	0	0	[::]:domain	[::]:*		2616/named
udp6	0	0	fe80::213:2ff:felf::ntp	[::]:*		3555/ntpd
udp6	0	0	ip6-localhost:ntp	[::]:*		3555/ntpd
udp6	0	0	[::]:ntp	[::]:*		3555/ntpd

lsuf (Liste des fichiers ouverts)

`lsuf` est particulièrement utile parce que sous Unix, tout est fichier : sockets unix, sockets ip, répertoires, etc.

Vous permet d'associer les fichiers ouverts par :

-p : PID (ID Processus)

-i : Adresse réseau (protocole:port)

-u : Utilisateur

Isof

Exemple :

- Tout d'abord *netstat -ln --tcp* détermine que le port 6010 est ouvert et en attente de connexion (LISTEN)

```
# netstat -ln --tcp
```

```
Active Internet connections (only servers)
```

Proto	Recv-Q	Send-Q	Local Address	Foreign Address	State
tcp	0	0	127.0.0.1:6010	0.0.0.0:*	LISTEN
tcp	0	0	127.0.0.1:6011	0.0.0.0:*	LISTEN

Isof (suite)

Quels services réseau sont en cours d'exécution pour moi ?

```
# lsof -i
COMMAND      PID        USER      FD  TYPE  DEVICE  SIZE  NODE  NAME
firefox      4429      hervey    50u  IPv4  1875852      TCP 192.168.179.139:56890->128.223.60.21:www (ESTABLISHED)
named        5378      bind      20u  IPv6  13264      TCP *:domain (LISTEN)
named        5378      bind      21u  IPv4  13267      TCP localhost:domain (LISTEN)
sshd         5427      root       3u  IPv6  13302      TCP *:ssh (LISTEN)
cupsd        5522      root       3u  IPv4  1983466     TCP localhost:ipp (LISTEN)
mysqld       5586      mysql     10u  IPv4  13548      TCP localhost:mysql (LISTEN)
snmpd        6477      snmp       8u  IPv4  14633      UDP localhost:snmp
exim4        6772      Debian-exim 3u  IPv4  14675      TCP localhost:smtp (LISTEN)
ntpd         6859      ntp        16u  IPv4  14743      UDP *:ntp
ntpd         6859      ntp        17u  IPv6  14744      UDP *:ntp
ntpd         6859      ntp        18u  IPv6  14746      UDP [fe80::250:56ff:fec0:8]:ntp
ntpd         6859      ntp        19u  IPv6  14747      UDP ip6-localhost:ntp
proftpd      7185      proftpd    1u  IPv6  15718      TCP *:ftp (LISTEN)
apache2      7246      www-data   3u  IPv4  15915      TCP *:www (LISTEN)
apache2      7246      www-data   4u  IPv4  15917      TCP *:https (LISTEN)
...
iperf        13598     root       3u  IPv4  1996053     TCP *:5001 (LISTEN)
apache2      27088     www-data   3u  IPv4  15915      TCP *:www (LISTEN)
apache2      27088     www-data   4u  IPv4  15917      TCP *:https (LISTEN)
```

tcpdump

- Affiche les en-têtes de paquets reçus par une interface donnée. En option, permet de filtrer à l'aide d'expressions booléennes.
- Vous permet d'enregistrer des informations dans un fichier pour analyse ultérieure.
- Nécessite des droits d'administrateur (root) étant donné que vous devez configurer les cartes d'interfaces réseau (NIC) en mode "espion".

tcpdump

Quelques options utiles :

- i** : Spécifie l'interface (ex : -i eth0)
- l** : Place la ligne de sortie standard (stdout) en tampon (visualisation au fil de la capture)
- v, -vv, -vvv** : Affiche des informations complémentaires
- n** : Pas de conversion des adresses en noms (pas de DNS)
- nn** : Pas de traduction des numéros de ports
- w** : Enregistre les paquets bruts dans un fichier
- r** : Lit les paquets d'un fichier créé avec '-w'

tcpdump

Expressions booléennes :

- Utilisant les opérateurs 'AND', 'OR', 'NOT'
- Les expressions se composent d'une ou plusieurs primitives, constituées d'un qualificatif et d'un ID (nom ou numéro) :

Expression ::= [NOT] <primitive> [AND | OR | NOT <primitive> ...]

<primitive> ::= <qualifier> <name|number>

<qualifier> ::= <type> | <address> | <protocol>

<type> ::= host | net | port | port range

<address> ::= src | dst

<protocol> ::= ether | fddi | tr | wlan | ip | ip6 | arp | rarp | decnet | tcp | udp

tcpdump

Exemples :

- Affiche tout le trafic HTTP en provenance de 192.168.1.1

```
# tcpdump -lnXvvv port 80 and src host 192.168.1.1
```

- Affiche tout le trafic en provenance de 192.168.1.1 *excepté le trafic SSH*

```
# tcpdump -lnXvvv src host 192.168.1.1 and not port 22
```

Wireshark

- Wireshark est un analyseur de paquets graphique basé sur *libpcap*, la bibliothèque que *tcpdump* utilise pour capturer et stocker des paquets
- L'interface graphique présente entre autres les avantages suivants :
 - Visualisation hiérarchique par protocole (drill-down)
 - Suivi de “conversation” TCP (Follow TCP Stream)
 - Couleurs pour distinguer les types de trafic
 - Quantité de statistiques, graphes, etc.

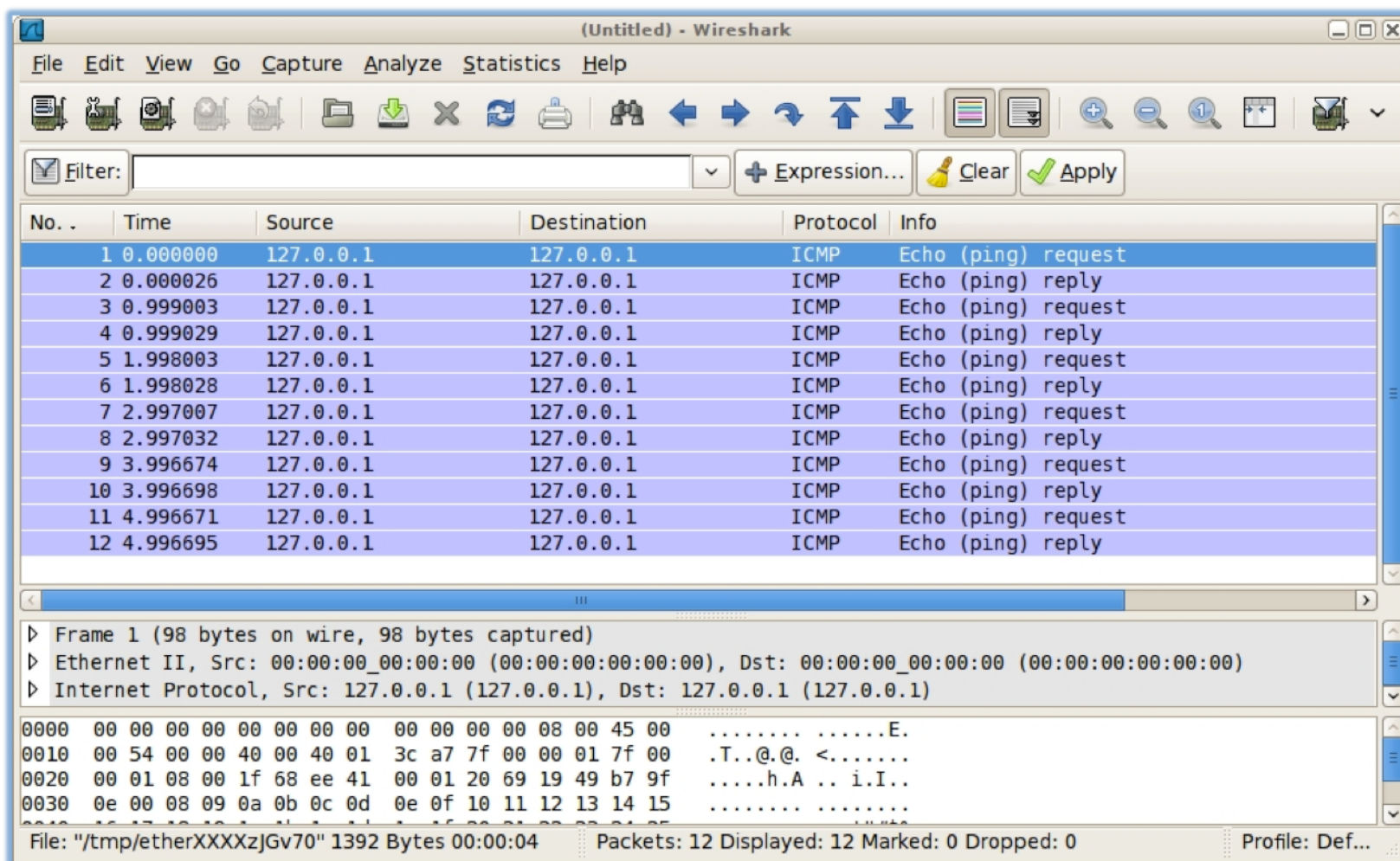
Wireshark

- Wireshark est le successeur d'*Ethereal*.
- L'association de *tcpdump* et *wireshark* peut être très puissante. Par exemple :

```
# tcpdump -i eth1 -A -s1500 -2 dump.log port 21  
$ sudo wireshark -r dump.log
```



Wireshark



iptraf

- **Nombreuses statistiques et fonctions mesurables**
 - Par protocole/port
 - Par taille de paquet
 - Génère des journaux
 - Utilise le DNS pour traduire les adresses
- **Avantages**
 - Simplicité
 - Par Menu (utilise “curses”)
 - Configuration flexible

iptraf

Peut être exécuté périodiquement en tâche de fond (-B)

- Peut, par exemple, être lancé sous forme de tâche cron pour analyser périodiquement les journaux.
 - Génère des alarmes
 - Enregistre dans une base de données
 - A un grand nom... “Interactive Colorful IP LAN Monitor”
 - etc.

Exemple : `iptraf -i eth1`

iptraf -i eth0

Exemple de sortie *iptraf* résultant de la commande ci-dessus :

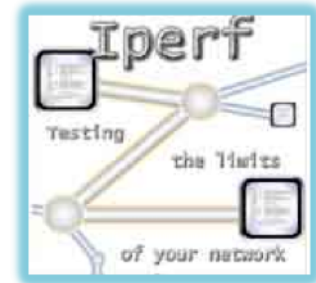
```
IPtraf
TCP Connections (Source Host:Port) ———— Packets — Bytes Flags Iface
190.187.47.86:37350 > 572 30332 —A— eth0
128.223.157.19:80 > 555 1572428 —A— eth0
201.215.63.27:32798 > 224 11708 —A— eth0
128.223.157.19:22 > 231 67780 —PA— eth0
66.249.68.14:42157 > 1 52 —A— eth0
128.223.157.19:80 = 0 0 — eth0
66.249.68.14:62173 = 7 565 CLOSED eth0
128.223.157.19:80 = 5 2386 CLOSED eth0
128.223.157.20:58832 = 6 333 CLOSED eth0
128.223.142.32:22 = 4 879 —PA— eth0

TCP: 5 entries ———— Active

ICMP echo rply (84 bytes) from 128.223.157.19 to 202.178.122.10 on eth0
ICMP echo req (84 bytes) from 202.178.122.10 to 128.223.157.19 on eth0
ICMP echo rply (84 bytes) from 128.223.157.19 to 202.178.122.10 on eth0
ICMP echo req (84 bytes) from 202.178.122.10 to 128.223.157.19 on eth0
ICMP echo rply (84 bytes) from 128.223.157.19 to 202.178.122.10 on eth0
Bottom ——— Elapsed time: 0:00 ———
Pkts captured (all interfaces): 1675 | TCP flow rate: 15.20 kbits/s
Up/Dn/PgUp/PgDn-scroll H-more TCP info W-chg actv win S-sort TCP X-exit
```

iperf

- Pour mesurer le débit réseau entre deux points
- *iperf* comporte deux modes, *serveur* et *client*
- Facile à utiliser
- Très utile pour optimiser les paramètres TCP
 - Taille de fenêtre TCP (socket buffer)
 - Taille maximale de segment MTU
 - Voir `man iperf` pour plus d'informations



iperf

- En utilisant UDP, vous pouvez générer des rapports sur les paquets perdus et les instabilités (*jitter*)
- Vous pouvez lancer plusieurs sessions parallèles en utilisant les *threads*
- Prend en charge IPv6

Paramètres iperf

Usage: iperf [-s|-c host] [options]
iperf [-h|--help] [-v|--version]

Client/Serveur :

- f, --format [kmKM]** format pour le rapport : Kbits, Mbits, Ko, Mo
- i, --interval #** nombre de secondes entre les rapports périodiques de bande passante
- l, --len #[KM]** longueur du tampon de lecture ou écriture (par défaut 8 Ko)
- m, --print_mss** affiche la taille maximale de segment TCP (en-tête MTU - TCP/IP)
- p, --port #** port du serveur à écouter ou auquel se connecter
- u, --udp** utiliser UDP plutôt que TCP
- w, --window #[KM]** taille de la fenêtre TCP (taille du tampon de la socket)
- B, --bind <host>** attacher à l'hôte, une interface ou une adresse multicast
- C, --compatibility** pour utilisation avec des versions anciennes ; n'envoie pas de messages supplémentaires
- M, --mss #** définit la taille de segment TCP maximale (MTU - 40 octets)
- N, --nodelay** configure le paramètre "TCP no delay", désactivation de l'algorithme de Nagle.
- V, --IPv6Version** Configure le domaine en IPv6

Spécifiques au serveur :

- s, --server** lancement en mode serveur
- U, --single_udp** lancement en mode UDP single thread
- D, --daemon** lancement du serveur en démon

Spécifiques au client :

- b, --bandwidth #[KM]** pour UDP, bande passante de transmission en bits/sec (par défaut 1 Mbit/sec, implique -u))
- c, --client <host>** lancement en mode client, connexion à l'hôte.
- d, --dualtest** Effectue un test bidirectionnel simultané
- n, --num #[KM]** nombre d'octets à transmettre (au lieu de -t)
- r, --tradeoff** Effectue un test bidirectionnel individuel
- t, --time #** temps en secondes pour transmettre (par défaut 10 sec)
- F, --fileinput <name>** entre les données à transmettre depuis un fichier
- I, --stdin** entre les données à transmettre depuis stdin
- L, --listenport #** port devant recevoir des tests bidirectionnels en retour.
- P, --parallel #** nombre de threads clients en parallèle à lancer
- T, --ttl #** time-to-live (temps de vie), pour multicast (par défaut 1))

iperf - TCP

```
$ iperf -s
```

```
-----  
Server listening on TCP port 5001  
TCP window size: 85.3 KByte (default)  
-----
```

```
[ 4] local 128.223.157.19 port 5001 connected with 201.249.107.39 port 39601  
[ 4] 0.0-11.9 sec  608 KBytes  419 Kbits/sec  
-----
```

```
# iperf -c nsrc.org
```

```
-----  
Client connecting to nsrc.org, TCP port 5001  
TCP window size: 16.0 KByte (default)  
-----
```

```
[ 3] local 192.168.1.170 port 39601 connected with 128.223.157.19 port 5001  
[ 3] 0.0-10.3 sec  608 KBytes  485 Kbits/sec
```

iperf - UDP

```
# iperf -c host1 -u -b100M
```

```
-----  
Client connecting to nsdb, UDP port 5001  
Sending 1470 byte datagrams  
UDP buffer size: 106 KByte (default)↵
```

```
-----  
[ 3] local 128.223.60.27 port 39606 connected with 128.223.250.135 port 5001  
[ 3] 0.0-10.0 sec 114 MBytes 95.7 Mbits/sec  
[ 3] Sent 81377 datagrams  
[ 3] Server Report:  
[ 3] 0.0-10.0 sec 114 MBytes 95.7 Mbits/sec 0.184 ms 1/81378 (0.0012%)↵
```

```
$ iperf -s -u -i 1
```

```
-----  
Server listening on UDP port 5001  
Receiving 1470 byte datagrams  
UDP buffer size: 108 KByte (default)↵
```

```
-----  
[ 3] local 128.223.250.135 port 5001 connected with 128.223.60.27 port 39606  
[ 3] 0.0- 1.0 sec 11.4 MBytes 95.4 Mbits/sec 0.184 ms 0/ 8112 (0%)↵  
[ 3] 1.0- 2.0 sec 11.4 MBytes 95.7 Mbits/sec 0.177 ms 0/ 8141 (0%)↵  
[ 3] 2.0- 3.0 sec 11.4 MBytes 95.6 Mbits/sec 0.182 ms 0/ 8133 (0%)↵  
↵...  
9.0 -8.0 ]3 [ sec 11.4 MBytes 95.7 Mbits/sec 0.177 ms 0/ 8139 (0%)↵  
[ 3] 9.0-10.0 sec 11.4 MBytes 95.7 Mbits/sec 0.180 ms 0/ 8137 (0%)↵  
[ 3] 0.0-10.0 sec 114 MBytes 95.7 Mbits/sec 0.184 ms 1/81378 (0.0012%)↵
```

Bibliographie

- *Monitoring de la mémoire virtuelle avec vmstat*
<http://www.linuxjournal.com/article/8178>
- *Comment utiliser TCPDump*
<http://www.erg.abdn.ac.uk/users/alastair/tcpdump.html>
- *Exemple de commande linux tcpdump*
<http://smartproteam.com/linux-tutorials/linux-command-tcpdump/>
- *Utilisation simple de tcpdump*
<http://linux.byexamples.com/archives/283/simple-usage-of-tcpdump/>
- *Page de manuel Commande TCPDUMP avec exemples*
<http://www.cyberciti.biz/howto/question/man/tcpdump-man-page-with-examples.php>
- *Tutoriel TCPDump*
<http://inst.eecs.berkeley.edu/~ee122/fa06/projects/tcpdump-6up.pdf>