Surveillance du réseau et de gestion

Définition et analyse des performances du réseau

Métrologie des performances réseau

- Planification de la gestion des performances
- Métrologie
  - réseau
  - *système*
  - *service*
- Définitions

Mesures courantes de performances réseau

- En termes de trafic :
  - Bits par seconde
  - Paquets par seconde
  - Paquets *unicast* ou *non-unicast*
  - Erreurs
  - Paquets perdus
  - Flux par seconde
  - Temps “aller-retour” (RTT, *Round trip time*)
  - Gigue (variation des temps RTT)

Capacité nominale des canaux

- Nombre maximal de bits pouvant être transmis pendant une unité de temps (ex : bits par seconde)
- Dépend de :
  - la largeur de bande du support physique
    - câble
    - ondes électromagnétiques
  - la capacité de traitement de chaque élément de transmission
  - l'efficacité des algorithmes utilisés pour accéder au support
  - le codage et la compression des canaux

Capacité effective des canaux

- Toujours inférieure à la capacité nominale
- Dépend :
  - du temps système consommé par les protocoles de chaque couche
  - de la capacité des périphériques aux deux extrémités
    - efficacité des algorithmes de contrôle de flux, etc.
      - Exemple : TCP

Utilisation des canaux

- Quelle est la proportion de capacité nominale des canaux réellement utilisée ?
- Importante !
  - Planification ultérieure
    - quelle est la croissance anticipée ?
    - quand faudra-il prévoir l'achat de capacité supplémentaire?
    - où investir en termes de mises à jour ?
  - Résolution des problèmes
    - quels sont les goulots d'étranglement, etc.

95e centile

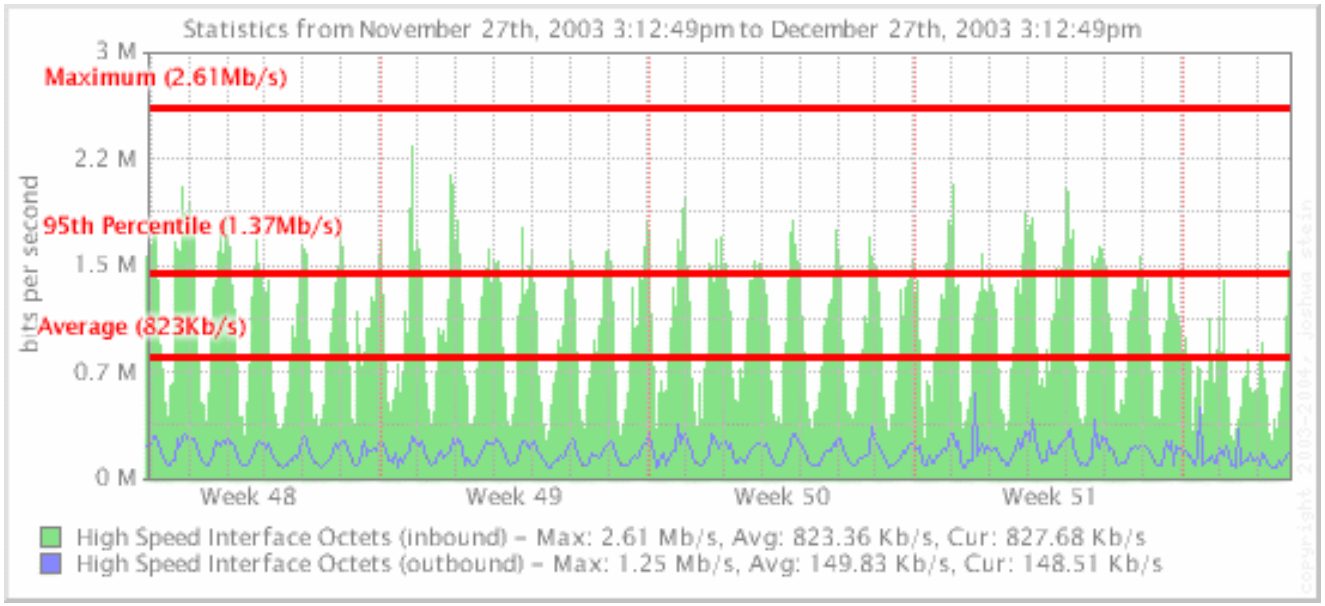
- Plus petite valeur supérieure à 95 % des valeurs d'un échantillon donné
- Cela signifie qu'à 95% du temps, l'utilisation du canal est égale ou *inférieure* à cette valeur

  - ou que les pics ne sont pas pris en compte
- Qu'est-ce qui est important dans les réseaux ?

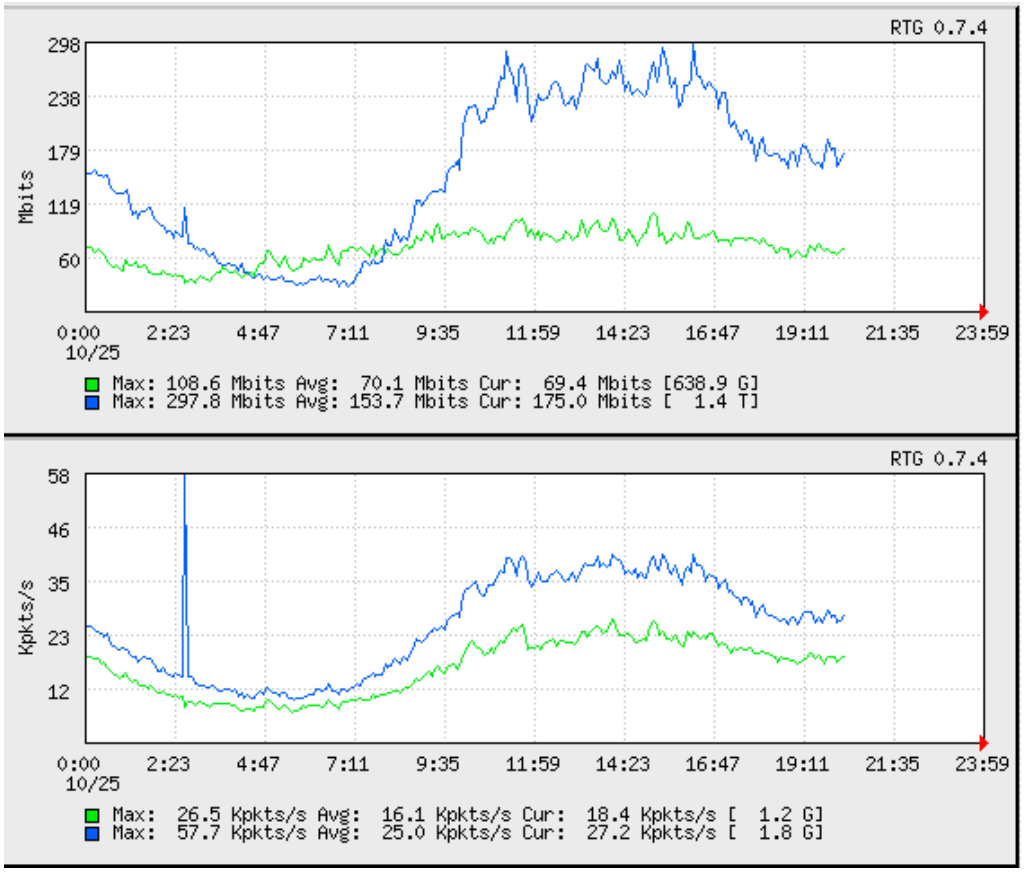
  - le centile donne une idée de l'utilisation type et soutenue du channel

  - les FAI se basent sur cette mesure pour facturer aux clients des connexions plus "puissantes".

95e centile (suite)



Bits / paquets par seconde



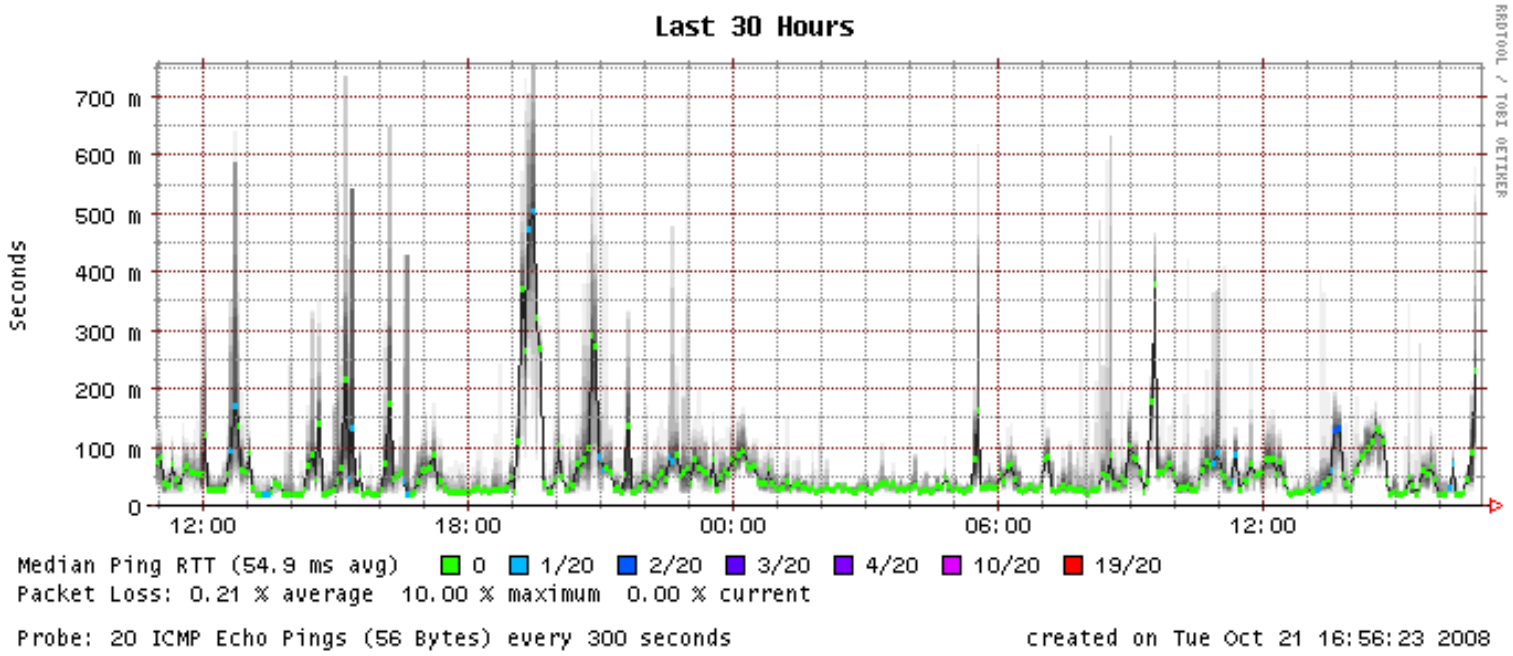
Retards de bout en bout

Temps de transmission d'un paquet sur <u>tout son chemin réseau</u>

- *Le paquet est créé par une application, confié au système d'exploitation, transmis à une carte d'terface réseau (NIC), codé, acheminé sur un support physique (cuivre, fibre, air), reçu par un dispositif intermédiaire (commutateur, routeur), analysé, retransmis sur un autre support, etc.*

- Les mesures les plus courantes sont effectuées avec *ping* qui mesure le temp de transmission aller-retour (RTT*, round-trip-time*) complet des paquets.

Historique de mesure des délais



Types de délais

Causes de retards dans les transmissions de bout en bout :

- délais de traitement (processeur)
- délais liés à la mise en tampon (files d'attente)
- délais de transmission
- délais de propagation

Délais de traitement

Temps requis pour analyser un en-tête de paquets et décider où l'envoyer (décision d'acheminement, par exemple)

- à l'intérieur d'un routeur, ce délai dépend du nombre d'entrées de la table d'acheminement, de la mise en oeuvre des structures de données, du matériel utilisé, etc.

Ceci peut inclure la vérification d'erreurs (calculs de somme d'en-têtes IPv4, IPv6, par exemple).

Délais de file d'attente

- Temps écoulé entre la mise en file d'attente d'un paquet et sa transmission
- Le nombre de paquets en attente dans la file d'attente est fonction de l'intensité et de la nature du traffic
- Les agorithmes de file d'attente des routeurs s'efforcent d'adapter les délais en fonction des préférences spécifiées ou imposent des délais équivalents à l'ensemble du trafic.

Délais de transmission

Temps requis pour faire passer tous les bits d'un paquet sur le support de transmission.

Pour N=nombre de bits, T=taille du paquet, d=délai

d = T/N

Ainsi, pour transmettre 1024 bits avec une connexion Ethernet rapide (100 Mbps) :

d = 1024/1x10e8 = 10,24 microsecondes

Délais de propagation

- Une fois un bit “poussé” sur le support de transmission, temps requis pour qu’il se propage jusqu’à la fin de sa trajectoire physique.
- La vitesse de propagation du circuit dépend principalement de la longueur du circuit physique.
- Dans la majorité des cas, cette vitesse est proche de celle de la lumière.

Pour d = distance, v = vitesse de propagation :

DP = d/v

Transmission ou propagation ?

La distinction peut sembler ténue

Prenez cet exemple :

Deux circuits à 100 Mbps

- 1 km de fibre optique
- 30 km de liaison satellite entre la base et le satellite

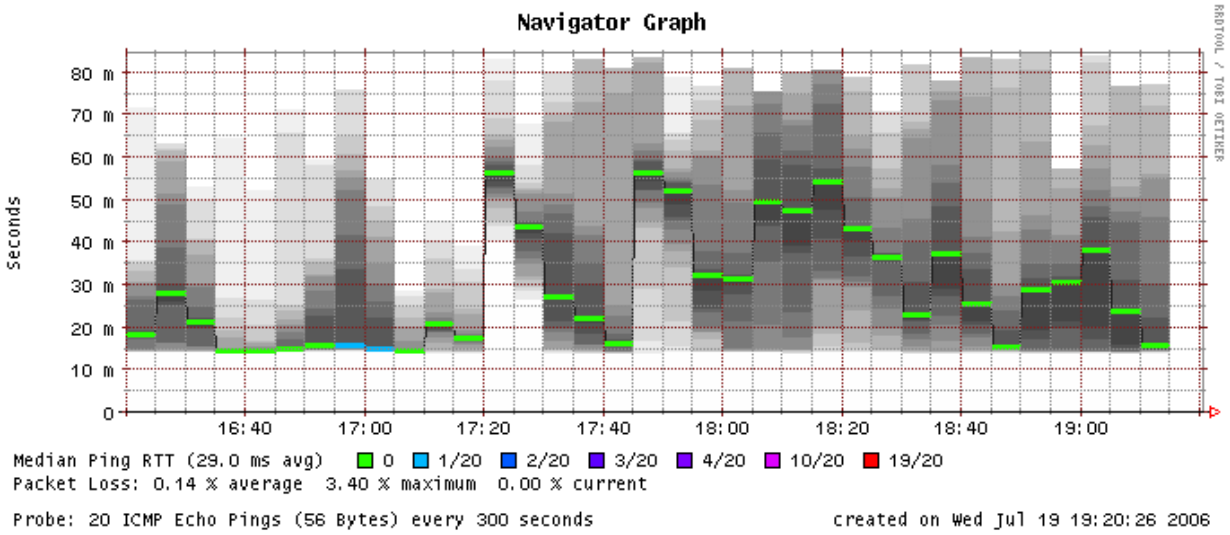
Pour deux paquets de même taille, quel sera le circuit le plus performant en termes de transmission ? De propagation ?

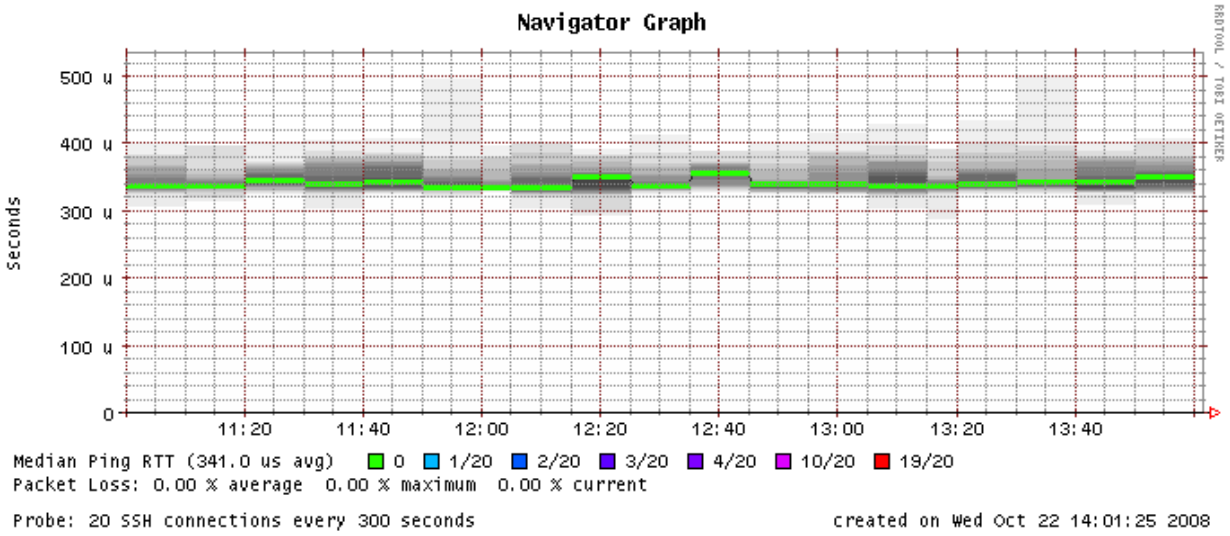
Perte de paquets

Les pertes de paquets sont dues au fait que les tampons ne sont pas extensibles

- Lorsqu'un paquet est acheminé vers un tampon déjà plein, il est éliminé
- Ce problème de perte, lorsqu'il doit être résolu, l'est à des niveaux supérieurs de la structure du réseau (couche transport ou application)
- La correction de cette perte par une retransmission des paquets peut aggraver d'encombrement si elle ne s'accompagne pas d'une forme de contrôle (des flux) (informant la source qu'il est momentanément inutile de continuer d'envoyer des paquets).

Gigue


Navigator Graph
Seconds
80 m
70 m
60 m
50 m
40 m
30 m
20 m
10 m
0
16:40
17:00
17:20
17:40
18:00
18:20
18:40
19:00
Median Ping RTT (29.0 ms avg)
0
1/20
2/20
3/20
4/20
10/20
19/20
Packet Loss: 0.14 % average 3.40 % maximum 0.00 % current
Probe: 20 ICMP Echo Pings (56 Bytes) every 300 seconds
created on Wed Jul 19 19:20:26 2006


Navigator Graph
Seconds
500 u
400 u
300 u
200 u
100 u
0
11:20
11:40
12:00
12:20
12:40
13:00
13:20
13:40
Median Ping RTT (341.0 us avg)
0
1/20
2/20
3/20
4/20
10/20
19/20
Packet Loss: 0.00 % average 0.00 % maximum 0.00 % current
Probe: 20 SSH connections every 300 seconds
created on Wed Oct 22 14:01:25 2008

Contrôle des flux et encombrements

- Limite les volumes transmis (débit) lorsque le débit de l'installation réceptrice est inférieur au débit d'arrivée des paquets.
- Limite les volumes envoyés (débit de transmission) du fait de pertes ou délais du circuit.

Contrôles dans TCP

IP (protocole internet) met en oeuvre un service non axé sur les connexions

- ce protocole ne prévoit pas de mécanisme de gestion de la perte de paquets.

TCP (Transmission *Control* Protocol) met en oeuvre un contrôle des flux et des encombrements

- uniquement aux extrémités car les noeuds intermédiaires au niveau du réseau ne parlent pas TCP.

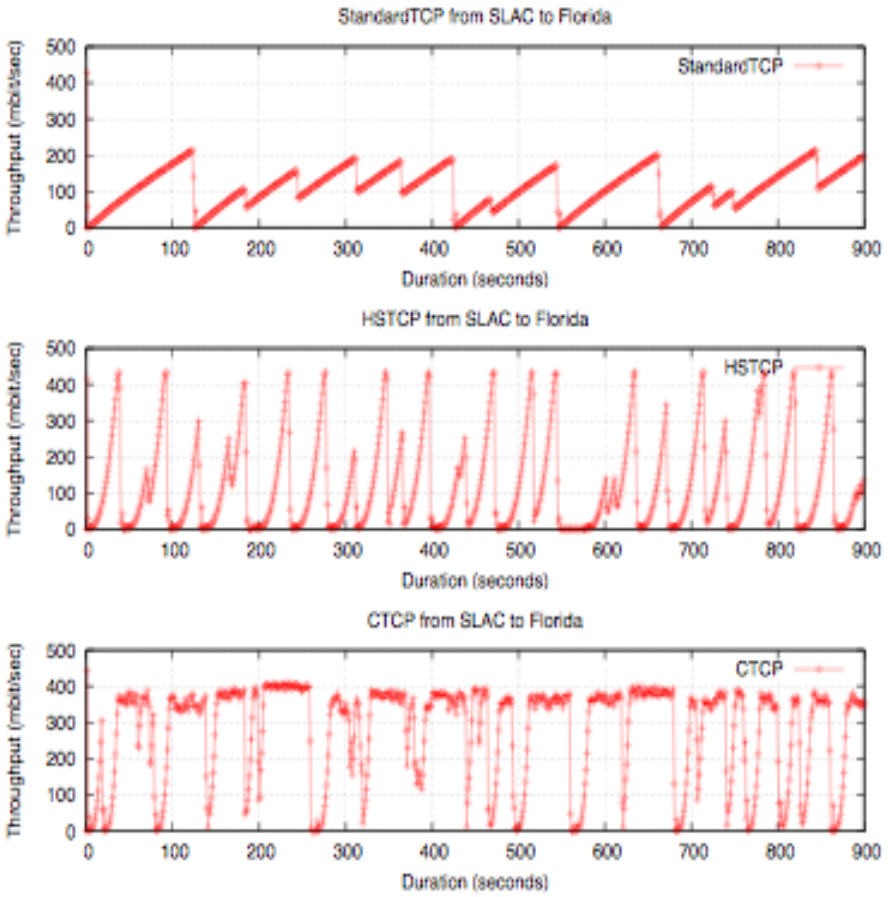
Encombrement / flux dans TCP

Flux : tributaire de la taille de fenêtre de réception (RcvWindow) envoyée par le côté récepteur.

Encombrement : contrôlé par la valeur de la fenêtre d'encombrement (Congwin)

- géré de manière autonome par l'émetteur
- variable selon la détection des paquets perdus
  - temporisation ou réception de trois ACK successifs
- **comportements :**
  - augmentation additive/retrait multiplicatif (algorithme AIMD)
  - démarrage lent
  - réaction aux événements de *temporisation.*

Différents algorithmes TCP de contrôle d'encombrements



Des questions ?

?

Section II

Analyse du réseau

Nous le savons déjà...

avant d'incriminer le réseau, vérifions si le problème vient de nous.

- **Problèmes à l'échelle locale ?**
  - problèmes de matériel
  - charge excessive (UC, mémoire, E/S)
- **Qu'est-ce qui est considéré "normal" ?**
  - recourez fréquemment aux outils d'analyse
    - familiarisez-vous avec l'état et les valeurs normales de votre machine.
  - **Les historiques sont indispensables**
    - agents et bases de données SNMP

Analyse locale

Trois catégories principales :

- Les processus
  - processus en cours d'exécution (en fonctionnement).
  - processus en attente (veille)
    - en attendant leur tour
    - bloqués
- La mémoire
  - réelle
  - virtuelle
- Les E/S (entrées/sorties)
  - stockage
  - réseau

Principaux indicateurs

UC insuffisante

- le nombre de processus en attente d'exécution est toujours élevé
- forte utilisation de l'UC (charge moyenne)

Mémoire insuffisante

- très peu de mémoire libre
- beaucoup d'opérations de permutations (en entrée/en sortie)

E/S lentes

- beaucoup de processus bloqués
- nombre élevé de transferts de blocs

Analyse sur le plan local

Unix s'accompagne heureusement de dizaines d'outils qui nous fournissent nombre d'informations utiles sur notre machine.

Parmi les plus connus :

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

vmstat

- Affichage périodiques d'informations de synthèse sur les processus, les mémoires, !!!pagin, les E/S/, l'état de l'UC, etc.

```
# 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 de performances de base pour environnements Unix/Linux
- Affichage périodique d'une liste de statistiques de performances système :
  - utilisation de l'UC
  - utilisation de la RAM et de la mémoire SWAP
  - charge moyenne (utilisation de l'UC)
  - informations par processus

top (suite)

- **Informations par processus (les colonnes les plus pertinentes sont affichées) :**
  - PID : ID de processus
  - USER : utilisateur (propriétaire) du processus
  - % UC : pourcentage d'utilisation de l'UC par le processus depuis le dernier échantillonnage
  - % MEM : pourcentage d'utilisation de la mémoire physique (RAM) par le processus
  - TIME : temps d'UC 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 à la fois simple et utile
- selon la machine, la plage acceptable considérée normale peut varier selon la machine :
  - les machines multi-processeur peuvent gérer des processus plus actifs par unité de temps (que les machines monoprocesseur).

top

Commandes clavier *interactives* pour *top*

- **f** : ajout ou suppression de colonnes
- **F** : spécifie la colonne de tri
- **<** , **>** : déplace la colonne de tri
- **u** : spécifie un utilisateur
- **k** : spécifie un processus à “tuer” (arrêter)
- **d**, **s** : modifie l’intervalle de mise à jour de l’affichage

Netstat (suite)

Affichage d'informations sur :

- les connexions réseau
- les tables d'acheminement
- les statistiques d'interface (NIC)
- les membres de groupes de multidiffusion (*multicast*)

Netstat (suite)

Options utiles

-n : affiche les adresses, les ports et les ID d'utilisateur en format *numérique*

-r : table d'acheminement (*routing*)

-s : *statistiques* par protocole

-I : état des *interfaces*

-l : écoute (*listening*) des interfaces de connexion (*sockets*)

--tcp, **--udp** : spécification du protocole

-A : famille d'*adresses* [inet | inet6 | unix | etc.]

-p : nom de chaque *processus* pour chaque port

-c : affichage en *continu* de la sortie et des résultats

netstat (suite)

Exemples :

```
# 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 *3333https                 *:*                     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:fe1f::ntp [::]:*                              3555/ntpd
udp6       0      0 ip6-localhost:ntp       [::]:*                              3555/ntpd
udp6       0      0 [::]:ntp                [::]:*                              3555/ntpd
```

lsof (LiSt of Open Files)

`lsof` est particulièrement utile car tout est fichier dans Unix : les interfaces de connexion unix, les interfaces de connexion ip, les répertoires, etc.

Permet d'associer les fichiers ouverts par :

-p : ID de processus (PID)

-i : adresse réseau (protocole:port)

-u : utilisateur

lsof (suite)

Exemple:

- Utilisez tout d'abord *netstat -ln –tcp* pour déterminer quel port 6010 est ouvert et en attente d'une 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
```

Lsof (suite)

Identifiez le processus pour lequel le port (6010) est ouvert ainsi que les autres ressources utilisées :

```
# lsof -i tcp:6010
COMMAND   PID   USER   FD   TYPE DEVICE SIZE NODE NAME
sshd      10301 root    6u  IPv4  53603       TCP localhost.localdomain:x11-ssh-offset (LISTEN)
sshd      10301 root    7u  IPv6  53604       TCP [::1]:x11-ssh-offset (LISTEN)
```

```
# lsof -p 10301
COMMAND   PID USER   FD   TYPE     DEVICE    SIZE    NODE NAME
sshd    10301 root  cwd    DIR        8,2    4096       2 /
sshd    10301 root  rtd    DIR        8,2    4096       2 /
sshd    10301 root  txt    REG        8,2  379720 1422643 /usr/sbin/sshd
sshd    10301 root  mem    REG        8,2   32724 1437533 /usr/lib/libwrap.so.0.7.6
sshd    10301 root  mem    REG        8,2   15088 3080329 /lib/libutil-2.4.so
sshd    10301 root  mem    REG        8,2   75632 1414093 /usr/lib/libz.so.1.2.3
sshd    10301 root  mem    REG        8,2   96040 3080209 /lib/libnsl-2.4.so
sshd    10301 root  mem    REG        8,2  100208 1414578 /usr/lib/libgssapi_krb5.so.2.2
sshd    10301 root  mem    REG        8,2   11684 1414405 /usr/lib/libkrb5support.so.0.0
sshd    10301 root  mem    REG        8,2   10368 3080358 /lib/libsetrans.so.0
sshd    10301 root  mem    REG        8,2    7972 3080231 /lib/libcom_err.so.2.1
sshd    10301 root  mem    REG        8,2   30140 1420233 /usr/lib/libcrack.so.2.8.0
sshd    10301 root  mem    REG        8,2   11168 3080399 /lib/security/pam_succeed_if.so
...
```

lsof (suite)

Quels sont les services de réseau en cours d'exécution ?

```
# 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 l'en-tête des paquets reçus par une interface donnée. Filtrage optionnel au moyen d'expressions booléennes.
- Permet d'écrire des informations dans un fichier en vue d'une analyse ultérieure.
- Nécessite les privilèges administrateur (*root*) car vous devez configurer les interfaces réseau (NIC) en mode "promiscuité".

Tcpdump (suite)

Options utiles :

- **-i** : Spécifier l'interface (ex : -i eth0)
- **-l** : Mettre la ligne stdout en tampon (visualisation pendant capture)
- **-v**, **-vv**, **-vvv** : Afficher d'autres informations
- **-n** : Ne pas convertir les adresses en noms (éviter DNS)
- **-nn** : Ne pas convertir les numéros de port
- **-w** : Ecrire les paquets bruts dans un fichier
- **-r** : Lire les paquets dans un fichier créé par '-w'

Tcpdump (suite)

Expressions booléennes :

- Recourent aux opérateurs 'AND', 'OR', 'NOT'
- Se composent d'un ou plusieurs primitives constituées d'un qualificateur et d'un ID (nom ou numéro) :

Expression ::= [NOT] <primitive> [AND | OR | NOT <primitive> ...]
<primitive> ::= <qualificateur> <nom|numéro>
<qualificateur> ::= <type> | <adresse> | <protocole>
<type> ::= hôte | net | port | plage ports
<adresse> ::= src | dst
<protocole> ::= ether | fddi | tr | wlan | ip | ip6 | arp | rarp | decnet | tcp | udp

Tcpdump (suite)

Exemples:

- Affiche tout le trafic HTTP issu de 192.168.1.1

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

- Affiche tout le trafic issu de 192.168.1.1 *à l'exception* de SSH

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

Wireshark

- Wireshark est un analyseur représentant les paquets sous une forme graphique à partir de la bibliothèque *libpcap*, également utilisée par *tcpdump* pour capturer et enregistrer les paquets

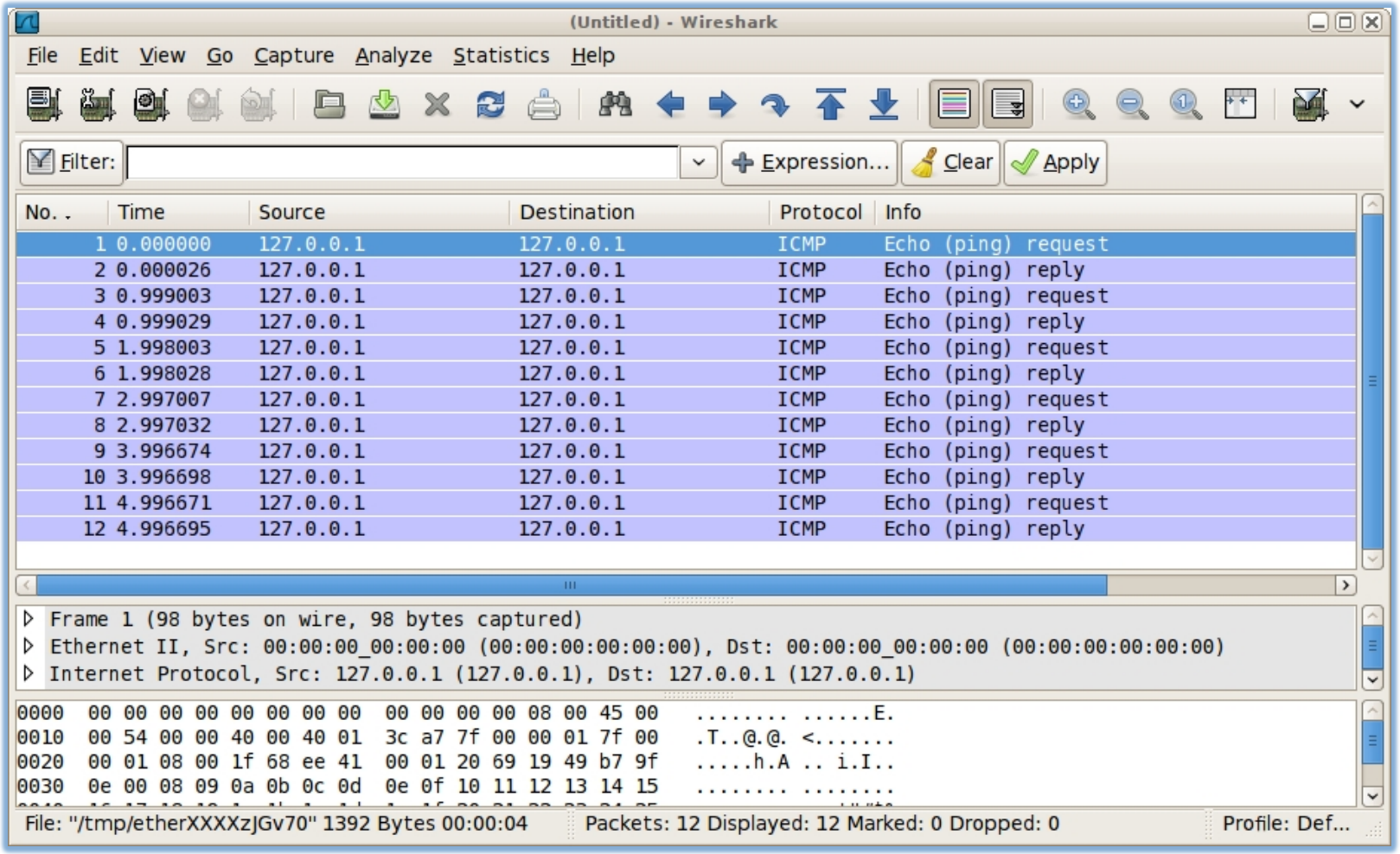
- L'interface graphique présente différents avantages parmi lesquels :
  - la visualisation par protocole (drill-down!!!)
  - le suivi d'une "conversation" TCP (suivi de flux TCP)
  - des couleurs distinguant les différents types de trafic
  - beaucoup de statistiques, graphiques, etc.

Wireshark (suite)

- Wireshark est issu d'*Ethereal*.
- La combination *tcpdump* et *wireshark* peut être très puissante. Exemple :

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

Wireshark (suite)



iptraf

- **De nombreuses fonctions et statistiques mesurables**
  - par protocole/port
  - par taille de paquets
  - génération de journaux
  - traduction d'adresses avec DNS
- **Avantages**
  - simplicité
  - basé sur des menus (utilise des sous-programmes "curses")
  - configuration souple

Iptraf (suite)

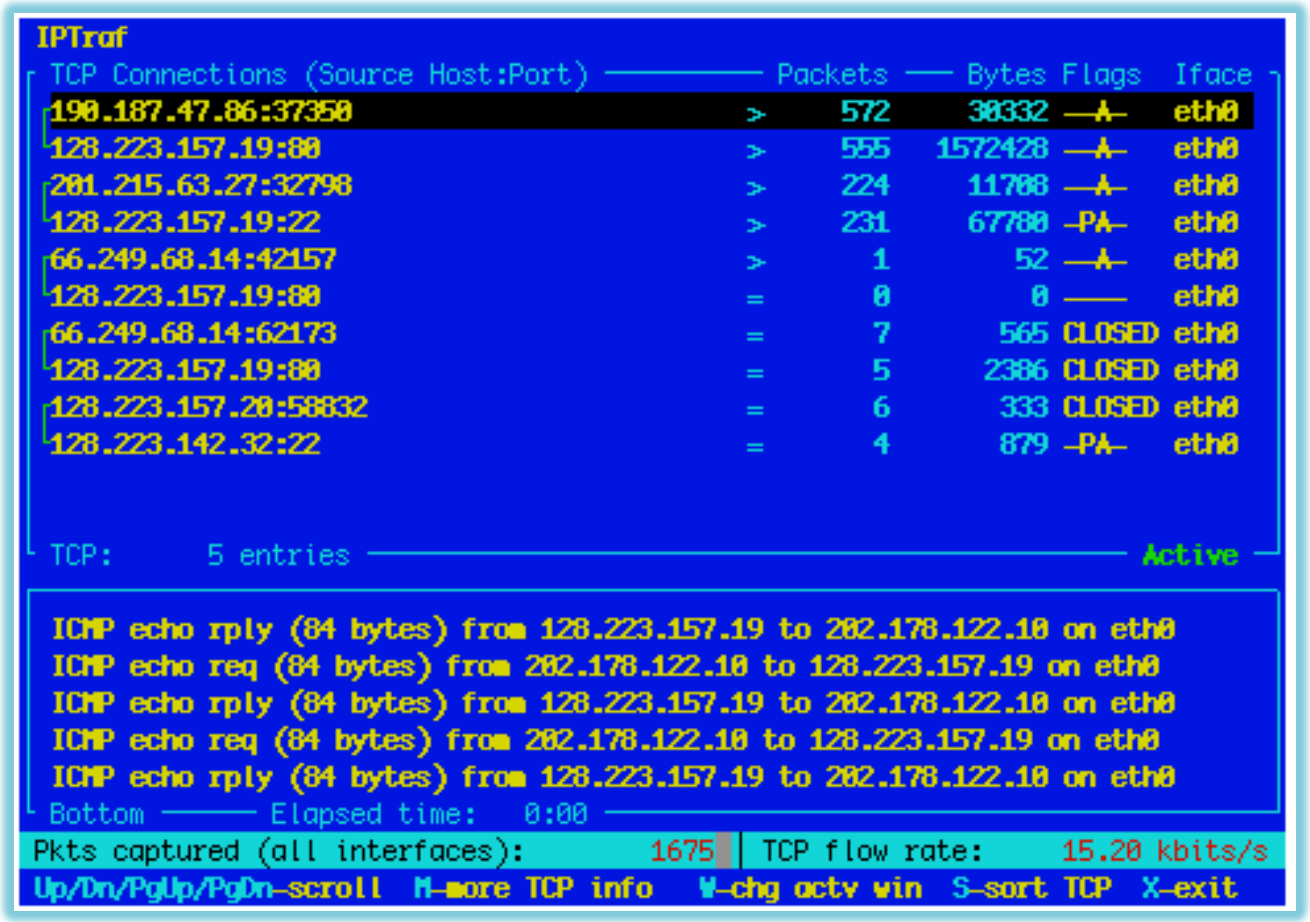
Peut être exécuté périodiquement en arrière-plan (-B)

- permet par exemple de programmer une analyse périodique des journaux au moyen d'une tâche *cron*.
  - émission d'alertes
  - enregistrement dans une base de données
  - et un nom qui a de l'allure... "Interactive Colorful IP LAN Monitor"
  - etc...

Exemple : `iptraf -i eth1`

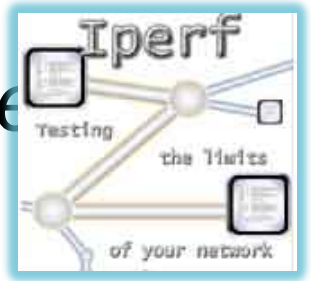
iptraf –i eth0

Exemple de sortie *iptraf* à partir de la commande ci-dessus :



iperf

- Mesure du débit du réseau entre deux points
- *iperf* comporte deux modes, *serve* *client*
- Facile à utiliser
- Très utile pour identifier les paramètres TCP optimaux
  - taille de fenêtre TCP (tampon d'interface)
  - longueur maximale des segments (unités MTU)
  - voir `man iperf` pour plus d'information.



Iperf (suite)

- UDP permet de générer des rapports de perte de paquets et de *gigue*
- Les *fils (threads)* permettent d'exécuter plusieurs sessions parallèles
- Supporte IPv6.

Paramètres iperf

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

Client/Server:
  -f, --format    [kmKM]   format to report: Kbits, Mbits, KBytes, MBytes
  -i, --interval  #        seconds between periodic bandwidth reports
  -l, --len       #[KM]    length of buffer to read or write (default 8 KB)
  -m, --print_mss          print TCP maximum segment size (MTU - TCP/IP header)
  -p, --port      #        server port to listen on/connect to
  -u, --udp                use UDP rather than TCP
  -w, --window    #[KM]    TCP window size (socket buffer size)
  -B, --bind      <host>   bind to <host>, an interface or multicast address
  -C, --compatibility      for use with older versions does not sent extra msgs
  -M, --mss       #        set TCP maximum segment size (MTU - 40 bytes)
  -N, --nodelay            set TCP no delay, disabling Nagle's Algorithm
  -V, --IPv6Version        Set the domain to IPv6

Server specific:
  -s, --server             run in server mode
  -U, --single_udp         run in single threaded UDP mode
  -D, --daemon             run the server as a daemon

Client specific:
  -b, --bandwidth #[KM]    for UDP, bandwidth to send at in bits/sec
                           (default 1 Mbit/sec, implies -u)
  -c, --client    <host>   run in client mode, connecting to <host>
  -d, --dualtest           Do a bidirectional test simultaneously
  -n, --num       #[KM]    number of bytes to transmit (instead of -t)
  -r, --tradeoff           Do a bidirectional test individually
  -t, --time      #        time in seconds to transmit for (default 10 secs)
  -F, --fileinput <name>   input the data to be transmitted from a file
  -I, --stdin              input the data to be transmitted from stdin
  -L, --listenport #       port to recieve bidirectional tests back on
  -P, --parallel  #        number of parallel client threads to run
  -T, --ttl       #        time-to-live, for multicast (default 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%)¶
¶...
               [ 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%)¶
```

Références

- *Supervision 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
- *linux : exemple de commande tcpdump*
  http://smartproteam.com/linux-tutorials/linux-command-tcpdump/
- *Exemples simples d'utilisation de tcpdump*
  http://linux.byexamples.com/archives/283/simple-usage-of-tcpdump/
- *TCPDUMP Command man page et exemples*
  http://www.cyberciti.biz/howto/question/man/tcpdump-man-page-with-examples.php
- *Didacticiel TCPDump*
  http://inst.eecs.berkeley.edu/~ee122/fa06/projects/tcpdump-6up.pdf