Débug réseaux

Compilation de commandes réseaux pour configurer , débuger et scanner son réseau

1) Interface réseau

1.1) ifconfig Command

Utilisé pour connaitre son adresse IP, Hardware / MAC address, ainsi que le MTU (Maximum Transmission Unit) de l'interfaces active.

1.1.1) Montrer toutes les interfaces

```
sudo ifconfig
```

```
$ sudo ifconfig
[sudo] Mot de passe de ordinatous :
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.160.119  netmask 255.255.255.0  broadcast 192.168.160.255
        inet6 fe80::be76:60cd:b69d:a75e  prefixlen 64  scopeid 0x20<link>
        ether 00:16:96:e7:04:59  txqueuelen 1000  (Ethernet)
        RX packets 2355  bytes 1260562 (1.2 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 2229  bytes 253965 (248.0 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1  (Boucle locale)
        RX packets 171  bytes 15470 (15.1 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 171  bytes 15470 (15.1 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlan0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether 42:a7:fd:c0:a4:fc  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

1.1.2) Lister les interfaces avec le statut up ou down , option -a

```
$ ifconfig -a
```

1.1.3) Assigner une IP

Adapter en fonction de votre réseau .

```
$ sudo ifconfig eth0 192.168.56.5 netmask 255.255.255.0
```

1.1.4) Activer une interface

```
$ sudo ifconfig up eth0
```

1.1.5) Desactiver une interface

```
$ sudo ifconfig down eth0
```

1.2) ip Command

ip et le successeur de *ifconfig* , il permet également de fournir les informations des interfaces , et de les configurer.

1.2.1) Montrer les infpormations de configuration réseau

```
ip a
```

```
$ ip a

1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1
link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
inet 127.0.0.1/8 scope host lo
valid_lft forever preferred_lft forever
inet6 ::1/128 scope host
valid_lft forever preferred_lft forever
2: enp1s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
link/ether 28:d2:44:eb:bd:98 brd ff:ff:ff:ff:ff:ff
inet 192.168.0.103/24 brd 192.168.0.255 scope global dynamic enp1s0
valid_lft 5772sec preferred_lft 5772sec
inet6 fe80::8f0c:7825:8057:5eec/64 scope link
valid_lft forever preferred_lft forever
3: wlp2s0: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default qlen 1000
link/ether 38:b1:db:7c:78:c7 brd ff:ff:ff:ff:ff:ff
...
```

1.2.2) Assigner une adresse temporaire à l'interface eth0

```
$ sudo ip addr add 192.168.56.1 dev eth0
```

1.2.3) Enlever une adresse d'une interface.

```
$ sudo ip addr del 192.168.56.15/24 dev eth0
```

1.2.4) Montrer le voisinage réseau

Généralement la passerelle

```
ip neigh
```

```
$ ip neigh
192.168.160.254 dev eth0 lladdr 00:12:3f:f2:47:b6 REACHABLE
```

Indicateur	Description
INCOMPLETE	La résolution d'adresse de l'hôte voisin est en cours.
REACHABLE	La correspondance entre les adresses IP et MAC a bien été établie et l'hôte voisin est apparemment joignable.
STALE	La correspondance entre les adresses IP et MAC a bien été établie, mais l'hôte voisin n'est probablement plus joignable et une vérification sera lancée dès la première émission.
DELAY	Un paquet a été émis à destination d'un voisin dans l'état STALE et une confirmation de correspondance d'adresses est en attente.
PROBE	La temporisation de l'état DELAY est expirée et la correspondance d'adresses n'a pas été confirmée ; une nouvelle résolution d'adresse a été initiée.
FAILED	La résolution d'adresse a échoué.
NOARP	Le voisin est validé ; aucune vérification ne doit être faite.
PERMANENT	Identique à NOARP ; seul le super utilisateur a la possibilité de supprimer l'entrée de la

table

1.2.5) Erreur de transmission

```
ip -s link ls dev eth0
```

```
$ ip -s link ls dev eth0
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP mode DEFAULT group default qlen 1000
    link/ether 00:16:96:e7:04:59 brd ff:ff:ff:ff:ff:ff
    RX: bytes  packets  errors  dropped overrun mcast
    5526698    7736     0       0       0       612
    TX: bytes  packets  errors  dropped carrier collsns
    890229     8022     0       0       0       0
```

1.3) ifup, ifdown, and ifquery command

1.3.1) ifup active une interface man ifquery

```
$ sudo ifup eth0
```

1.3.2) ifdown desactive une interface

```
$ sudo ifdown eth0
```

1.3.3) ifquery recherche dans la configuration

```
$ sudo ifquery eth0
```

```
sudo  ifquery -l --allow=hotplug
```

1.4) ethtool Command

ethtool est un utilitaire qui permet de requêter et modifier les paramètres du controller de carte réseau, ainsi que les drivers.

➥ Sous CentOS

```
$ sudo ethtool enp0s3
```

```
$ sudo ethtool enp0s3
Settings for enp0s3:
Supported ports: [ TP ]
Supported link modes:   10baseT/Half 10baseT/Full
100baseT/Half 100baseT/Full
1000baseT/Full
Supported pause frame use: No
Supports auto-negotiation: Yes
Advertised link modes:  10baseT/Half 10baseT/Full
100baseT/Half 100baseT/Full
1000baseT/Full
Advertised pause frame use: No
Advertised auto-negotiation: Yes
Speed: 1000Mb/s
Duplex: Full
Port: Twisted Pair
PHYAD: 0
Transceiver: internal
Auto-negotiation: on
MDI-X: off (auto)
Supports Wake-on: umbg
Wake-on: d
Current message level: 0x00000007 (7)
drv probe link
Link detected: yes
```

➥ Sous debian:

```
sudo ethtool -S eth0
```

```
$ sudo ethtool -S eth0
NIC statistics:
     tx_packets: 25132
     rx_packets: 27080
     tx_errors: 0
     rx_errors: 0
     rx_missed: 0
     align_errors: 75
     tx_single_collisions: 0
     tx_multi_collisions: 0
     unicast: 26175
     broadcast: 9
     multicast: 896
     tx_aborted: 0
     tx_underrun: 0
```

1.5) ping Command

ping (Packet INternet Groper) est un utilitaire qui permet de vérifier la connectivité entre 2 systèmes sur un réseau local (Local Area Network (LAN) distant Wide Area Network (WAN)). Il utilisae ICMP (Internet Control Message Protocol).

1.5.1) Tester la connectivité

```
$ ping 192.168.0.103
```

```
$ ping 192.168.0.103
PING 192.168.0.103 (192.168.0.103) 56(84) bytes of data.
64 bytes from 192.168.0.103: icmp_seq=1 ttl=64 time=0.191 ms
64 bytes from 192.168.0.103: icmp_seq=2 ttl=64 time=0.156 ms
64 bytes from 192.168.0.103: icmp_seq=3 ttl=64 time=0.179 ms
64 bytes from 192.168.0.103: icmp_seq=4 ttl=64 time=0.182 ms
64 bytes from 192.168.0.103: icmp_seq=5 ttl=64 time=0.207 ms
64 bytes from 192.168.0.103: icmp_seq=6 ttl=64 time=0.157 ms
^C
--- 192.168.0.103 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5099ms
rtt min/avg/max/mdev = 0.156/0.178/0.207/0.023 ms
```

1.5.2) Spécifier le nombre de ECHO_REQUEST packets option `-c`

```
$ ping -c 4 192.168.0.103
```

```
$ ping -c 4 192.168.0.103
PING 192.168.0.103 (192.168.0.103) 56(84) bytes of data.
64 bytes from 192.168.0.103: icmp_seq=1 ttl=64 time=1.09 ms
64 bytes from 192.168.0.103: icmp_seq=2 ttl=64 time=0.157 ms
64 bytes from 192.168.0.103: icmp_seq=3 ttl=64 time=0.163 ms
64 bytes from 192.168.0.103: icmp_seq=4 ttl=64 time=0.190 ms
--- 192.168.0.103 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3029ms
rtt min/avg/max/mdev = 0.157/0.402/1.098/0.402 ms
```

1.6) traceroute Command

Traceroute est un utilitaire qui affiche le chemin complet entre 2 systèmes. Il indique le nombre de *hops* (router IP's) sur le chemin pour se connecter au système final. Trés pratique pour compléter le debug après le *ping* .
Par exemple, traçons la route que les packets prennent depuis le système local system pour un des serveurs de google répondant à l'adresse IP 216.58.204.46. `traceroute 216.58.204.46`

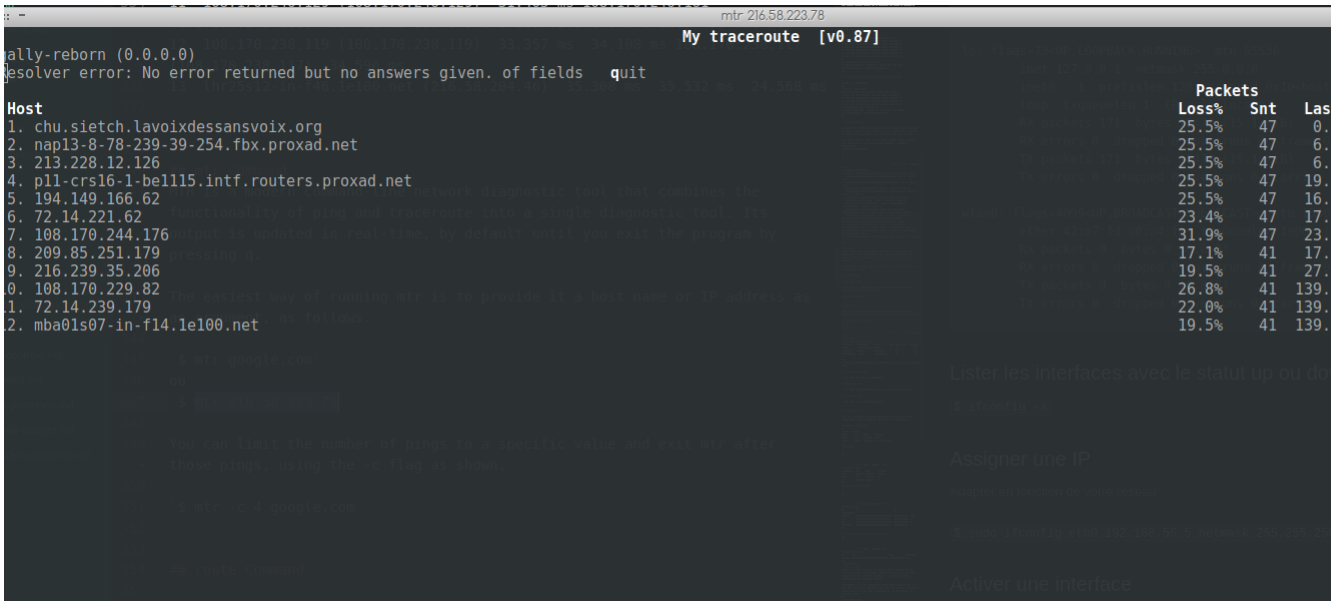
```
$ traceroute 216.58.204.46
traceroute to 216.58.204.46 (216.58.204.46), 30 hops max, 60 byte packets
 1  chu.sietch.lavoixdessansvoix.org (192.168.160.254)  0.165 ms  0.203 ms  0.202 ms
 2  nap13-8-78-239-39-254.fbx.proxad.net (78.239.39.254)  7.031 ms  7.714 ms  8.688 ms
 3  213.228.12.126 (213.228.12.126)  11.721 ms  11.957 ms  11.952 ms
 4  p11-crs16-1-be1115.intf.routers.proxad.net (194.149.162.153)  21.716 ms  22.916 ms  23.155 ms
 5  194.149.166.62 (194.149.166.62)  24.137 ms  25.126 ms  26.339 ms
 6  72.14.221.62 (72.14.221.62)  26.051 ms  25.918 ms  26.896 ms
 7  108.170.244.177 (108.170.244.177)  28.922 ms 108.170.244.240 (108.170.244.240)  21.758 ms  22.712 ms
 8  108.170.230.211 (108.170.230.211)  23.765 ms 209.85.248.117 (209.85.248.117)  17.522 ms 209.85.255.106 (209.85.255.106)
18.290 ms
 9  108.170.236.36 (108.170.236.36)  27.736 ms  28.191 ms  28.934 ms
10  216.239.58.132 (216.239.58.132)  29.172 ms 64.233.175.112 (64.233.175.112)  30.182 ms 216.239.58.2 (216.239.58.2)  30.437
ms
11  108.170.246.129 (108.170.246.129)  31.403 ms 108.170.246.161 (108.170.246.161)  33.701 ms 108.170.246.129
(108.170.246.129)  33.121 ms
12  108.170.238.119 (108.170.238.119)  33.357 ms  34.108 ms 108.170.238.117 (108.170.238.117)  34.596 ms
13  lhr25s12-in-f46.1e100.net (216.58.204.46)  35.308 ms  35.532 ms  24.568 ms
```

1.7) mtr Command

MTR pour My TraceRoute est un outil plus moderne qui combine traceroute et ping . Sa sortie est en temps réelle.

1.7.1) Usage

Il suffit de lui fournir une *ip* ou un domaine.
`$ mtr google.com` **ou** `$ mtr 216.58.223.78`



Il est possible de limiter le ping avec l'option `-c`.

```
$ mtr -c 4 google.com
```

1.8) route Command

route est un utilitaire qui permet d'afficher et de manipuler la table de routage d'un système *linux*. Souvent utiliser pour indiquer une route static.

1.8.1) Afficher la table de routage

```
$ sudo route
```

```
$ sudo route
Table de routage IP du noyau
Destination     Passerelle      Genmask         Indic Metric Ref    Use Iface
default         chu.sietch.lavo 0.0.0.0         UG    100    0        0 eth0
link-local      0.0.0.0         255.255.0.0     U     1000   0        0 eth0
192.168.160.0   0.0.0.0         255.255.255.0   U     100    0        0 eth0
```

Il existe de nombreuses commande pour configurer une route. En voici quelques unes:

1.8.2) Ajouter une passerelle

```
$ sudo route add default gw <gateway-ip>
```

1.8.3) Ajouter une route

```
$ sudo route add -net <network ip/cidr> gw <gateway ip> <interface>
```

1.8.4) Effacer une route

```
$ sudo route del -net <network ip/cidr>
```

1.9) nmcli Command

Nmcli est l'interface en ligne de commande du Network-Manager.

1.9.1) Afficher périphériques

```
$ nmcli dev status
```

```
$ nmcli dev status
DEVICE      TYPE      STATE      CONNECTION
virbr0      bridge    connected  virbr0
enp0s3      ethernet  connected  Wired connection 1
```

➥ Sous debian

```
$ nmcli dev status
```

```
$ nmcli dev status
PÉRIPHÉRIQUE  TYPE      ÉTAT        CONNEXION
eth0          ethernet  connecté    H3G3_USB
wlan0         wifi      déconnecté  --
lo            loopback  non-géré    --
```

1.9.2) Verifier les connections

```
$ nmcli con show
```

```
$ nmcli con show
NOM           UUID                                  TYPE             PÉRIPHÉRIQUE
H3G3_USB      08dcc4ed-1754-3b67-8a51-028f971884b9  802-3-ethernet   eth0
H3G3          1bdb80a2-c80a-4003-a25a-178ec7ec304c  802-11-wireless  --
H4G4          04345c54-9d51-41fa-aab6-8d9d3dd5ba3c  802-11-wireless  --
HUAWEI-0601   51b991a9-03b3-4116-bf6c-5830a877e0af  802-11-wireless  --
```

1.9.3) Afficher seulement les connexions actives

```
$ nmcli con show -a
```

```
$ nmcli con show -a
NOM       UUID                                  TYPE            PÉRIPHÉRIQUE
H3G3_USB  08dcc4ed-1754-3b67-8a51-028f971884b9  802-3-ethernet  eth0
```

2) Statistiques

2.1) netstat Command

netstat permet d'afficher les informations du sous-système réseau de Linux; les connexions réseau, les tables de routage, les statistiques des interfaces, les connexions masquées, les messages netlink, et les membres multicast.
Il permet de savoir quelle ressource réseau est utilisée, quelles applications sur quels ports.

```
$ sudo netstat tnlp
```

```
$ sudo netstat -tnlp
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:587             0.0.0.0:*               LISTEN      1257/master
tcp        0      0 127.0.0.1:5003          0.0.0.0:*               LISTEN      1/systemd
tcp        0      0 0.0.0.0:110             0.0.0.0:*               LISTEN      1015/dovecot
tcp        0      0 0.0.0.0:143             0.0.0.0:*               LISTEN      1015/dovecot
tcp        0      0 0.0.0.0:111             0.0.0.0:*               LISTEN      1/systemd
tcp        0      0 0.0.0.0:465             0.0.0.0:*               LISTEN      1257/master
tcp        0      0 0.0.0.0:53              0.0.0.0:*               LISTEN      1404/pdns_server
tcp        0      0 0.0.0.0:21              0.0.0.0:*               LISTEN      1064/pure-ftpd (SER
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      972/sshd
tcp        0      0 127.0.0.1:631           0.0.0.0:*               LISTEN      975/cupsd
tcp        0      0 0.0.0.0:25              0.0.0.0:*               LISTEN      1257/master
tcp        0      0 0.0.0.0:8090            0.0.0.0:*               LISTEN      636/lscpd (lscpd -
tcp        0      0 0.0.0.0:993             0.0.0.0:*               LISTEN      1015/dovecot
tcp        0      0 0.0.0.0:995             0.0.0.0:*               LISTEN      1015/dovecot
tcp6       0      0 :::3306                 :::*                    LISTEN      1053/mysqld
tcp6       0      0 :::3307                 :::*                    LISTEN      1211/mysqld
tcp6       0      0 :::587                  :::*                    LISTEN      1257/master
tcp6       0      0 :::110                  :::*                    LISTEN      1015/dovecot
tcp6       0      0 :::143                  :::*                    LISTEN      1015/dovecot
tcp6       0      0 :::111                  :::*                    LISTEN      1/systemd
tcp6       0      0 :::80                   :::*                    LISTEN      990/httpd
tcp6       0      0 :::465                  :::*                    LISTEN      1257/master
tcp6       0      0 :::53                   :::*                    LISTEN      1404/pdns_server
tcp6       0      0 :::21                   :::*                    LISTEN      1064/pure-ftpd (SER
tcp6       0      0 :::22                   :::*                    LISTEN      972/sshd
tcp6       0      0 ::1:631                 :::*                    LISTEN      975/cupsd
tcp6       0      0 :::25                   :::*                    LISTEN      1257/master
tcp6       0      0 :::993                  :::*                    LISTEN      1015/dovecot
tcp6       0      0 :::995                  :::*                    LISTEN      1015/dovecot
```

2.1.1) Afficher la table de routage, option `-r`

```
$ netstat -r
Table de routage IP du noyau
Destination     Passerelle      Genmask         Indic   MSS Fenêtre irtt Iface
default         chu.sietch.lavo 0.0.0.0         UG        0 0          0 eth0
link-local      0.0.0.0         255.255.0.0     U         0 0          0 eth0
192.168.160.0   0.0.0.0         255.255.255.0   U         0 0          0 eth0
```

2.1.1.1) State

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 se trouver vierge. Normalement, on trouvera une des valeur suivante:

Etat	Signification
ESTABLISHED	La socket a une connexion établie.
SYN_SENT	La socket attend activement d'établir une connexion.
SYN_REC	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.
CLOSE	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.

2.1.1.2) Liste des Options

```
-v, --verbose
       active le mode verbeux. Affiche quelques informations utiles concernant les familles d'adresses non configurées.

-n, --numeric
       affiche les adresses en format numérique au lieu d'essayer de déterminer le nom symbolique d'hôte, de port ou
d'utilisateur.

-p, --programs
       affiche le nom et le PID des processus propriétaires de chaque socket décrite. Vous devez être le propriétaire d'un
processus pour visualiser les sockets qui  lui
       appartiennent ou être l'utilisateur root pour disposer de toutes les informations.

-A, --af famille
       utilise  une  méthode différente pour affecter les familles d'adresses. famille est une liste de familles d'adresses
séparées par des (',') telles que inet, unix,
       ipx, ax25, netrom et ddp. L'utilisation des options longues suivantes a le même effet --inet, --unix, --ipx, --ax25, --
netrom et --ddp.

-c, --continous
       Demandera à netstat d'afficher la table sélectionnée chaque seconde jusqu'à ce que vous l'interrompiez.
```

Note: netstat est encore beaucoup utilisé, mais il est *deprecated* (déprécié), son successeurs est *ss*

2.2) ss Command

ss (socket statistics) est un outil puissant permetant de faire de l'investigation réseau. Il dump les statistiques socket et semble afficher plus de connexions *TCP*

2.2.1) Afficher les port TCP ouvert

```
$ ss -ta
State      Recv-Q Send-Q                  LocalAddress:Port                                  Peer Address:Port
LISTEN     0      128                        127.0.0.1:5939                                       *:*
LISTEN     0      50                         127.0.0.1:10391                                      *:*
LISTEN     0      5                          127.0.0.1:ipp                                        *:*
LISTEN     0      20                         *:smtp                                               *:*
LISTEN     0      128                        127.0.0.1:9050                                       *:*
LISTEN     0      128                        *:1022                                               *:*
LISTEN     0      80                        127.0.0.1:mysql                                      *:*
CLOSE-WAIT 47     0                          192.168.160.119:41064                          52.85.58.242:https
CLOSE-WAIT 32     0                            192.168.160.119:60828                        151.101.120.201:https
ESTAB      0      0                          127.0.0.1:34520                                127.0.0.1:10391
ESTAB      0      0                          127.0.0.1:10391                                127.0.0.1:34430
ESTAB      0      0                           127.0.0.1:34430
127.0.0.1:10391
ESTAB      0      0                           192.168.160.119:59718
34.210.41.110:https
ESTAB      0      0                         127.0.0.1:10391                                  127.0.0.1:34520
LISTEN     0      5                           ::1:ipp                                              :::*
LISTEN     0      20                          :::smtp                                              :::*
LISTEN     0      128                         :::1022                                              :::*
```

2.2.2) Afficher les connexions TCP active + timer

```
$ ss -to
State      Recv-Q Send-Q                                    Local Address:Port
Peer Address:Port
CLOSE-WAIT 47     0                                      192.168.160.119:41064
52.85.58.242:https
CLOSE-WAIT 32     0                                      192.168.160.119:60828
151.101.120.201:https
ESTAB      0      0                                           127.0.0.1:34520
127.0.0.1:10391
ESTAB      0      0                                           127.0.0.1:10391
127.0.0.1:34430
ESTAB      0      0                                           127.0.0.1:34430
127.0.0.1:10391
ESTAB      0      0                                      192.168.160.119:59718
34.210.41.110:https                timer:(keepalive,,0)
ESTAB      0      0                                            127.0.0.1:10391
127.0.0.1:34520
```

3) Scann réseaux

3.1) nc Command

NC (NetCat) appelé également "Network Swiss Army knife", est un utilitaire puissant qui sert à toutes les taches liées à TCP, UDP, or UNIX-domain sockets. Il permet d'ouvrir des connexions, d'écouter et de scanner les ports.
Note: il peut servir de proxy, pour trasférer des fichiers sur un autre port d'une machine distante.

3.1.1) Scanner les ports

```
$ nc -zv server.domaine.lan 21 22 80 443 3000
```

3.1.2) Scanner une plage de ports

```
$ nc -zv server.domaine.lan 20-90
```

3.1.3) Timeout

Cette commande permet de se connecter sur le port 5000 du server.domain.lan depuis son port 3000 avec un timeout de 10 secondes.

```
$ nc -p 3000 -w 10 server.domaine.lan 5000
```

3.2) nmap Command

Nmap (Network Mapper) est un outils puissant très utile pour l'administrateurs Linux system/network . Il permet d'obtenir

des informations sur un hote ou un réseau entier.
Il existe une interface graphique : ZenMap, très pratique , elle permet de jouer des scripts *sne* facilement, et de se créer des profils.

3.2.1) Usage:

```
# nmap [Scan Type(s)] [Options] {target specification}
```

On peut simplement lui fournir une *ip* ou un domaine.

```
$ nmap google.com
Starting Nmap 6.40 ( http://nmap.org ) at 2018-07-12 09:23 BST
Nmap scan report for google.com (172.217.166.78)
Host is up (0.0036s latency).
rDNS record for 172.217.166.78: bom05s15-in-f14.1e100.net
Not shown: 998 filtered ports
PORT    STATE SERVICE
80/tcp  open  http
443/tcp open  https
Nmap done: 1 IP address (1 host up) scanned in 4.92 seconds
```

3.2.2) Scanner une plage

```
┌$ nmap -sn 192.168.160.0/24

Starting Nmap 7.40 ( https://nmap.org ) at 2018-07-14 11:02 CEST
Nmap scan report for brussel.amghar.me (192.168.160.111)
Host is up (0.0053s latency).
Nmap scan report for kali.sietch.lavoixdessansvoix.org (192.168.160.117)
Host is up (0.00093s latency).
Nmap scan report for gally-reborn.sietch.lavoixdessansvoix.org (192.168.160.119)
Host is up (0.00020s latency).
Nmap scan report for chu.sietch.lavoixdessansvoix.org (192.168.160.254)
Host is up (0.0013s latency).
Nmap done: 256 IP addresses (4 hosts up) scanned in 6.87 seconds
```

3.2.3) Scanner plusieurs hotes:

```
# nmap 192.168.0.101 192.168.0.102 192.168.0.103
```

4) DNS information

4.1) host Command

host est une commande très simple, elle permet d'obtenir des informations sur les serveurs de noms, et de faire de la translation d'adresses.

```
$ host google.com
google.com has address 216.58.204.142
google.com has IPv6 address 2a00:1450:4007:811::200e
google.com mail is handled by 40 alt3.aspmx.l.google.com.
google.com mail is handled by 20 alt1.aspmx.l.google.com.
google.com mail is handled by 30 alt2.aspmx.l.google.com.
google.com mail is handled by 10 aspmx.l.google.com.
google.com mail is handled by 50 alt4.aspmx.l.google.com.
```

4.2) dig Command

dig (domain information groper) est un outil flexible pour interroger les serveurs de noms, souvent utiliser pour débuger , de par sa simplicité et la clareté de ses réponses.

```
$ dig google.com

; <<>> DiG 9.10.3-P4-Debian <<>> google.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 38099
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;google.com.            IN  A

;; ANSWER SECTION:
google.com.     152 IN  A   216.58.204.142

;; Query time: 0 msec
;; SERVER: 192.168.160.254#53(192.168.160.254)
;; WHEN: Sat Jul 14 11:08:04 CEST 2018
;; MSG SIZE  rcvd: 55
```

4.3) nslookup Command

```
$ nslookup google.com
Server:     192.168.160.254
Address:    192.168.160.254#53

Non-authoritative answer:
Name:   google.com
Address: 216.58.213.174
```

5) Capture

5.1) tcpdump Command

➥ Install tcpdump:

```
➥ sudo apt install tcpdump
```

Tcpdump est un *sniffer* réseau, il est utiliser pour capturer les packets qui circulent.

5.1.1) Capturer des packets

Il faut sécifier l'interface de capture avec l'option `-i`.

```
sudo tcpdump -i eth0
```

```
$ sudo tcpdump -i eth0
[sudo] Mot de passe de ordinatous :
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
11:23:35.040997 STP 802.1w, Rapid STP, Flags [Learn, Forward, Agreement], bridge-id 8000.24:b6:57:0f:53:da.8007, length 47
11:23:37.040980 STP 802.1w, Rapid STP, Flags [Learn, Forward, Agreement], bridge-id 8000.24:b6:57:0f:53:da.8007, length 47
11:23:37.301125 IP francisco.sietch.lavoixdessansvoix.org.mdns > 224.0.0.251.mdns: 0- [0q] 1/0/0 TXT "type=0" "version=1"
"refresh-age-timeout=0" "priority=0" "refresh-flag=0" "root-mac-address=24:b6:57:0f:53:da" "cost=0" "transm-
address=192.168.160.253" "transm-interface=300000" "voice-vlan-id=2" "voice-vlan-vpt=5" "voice-vlan-dscp=46" "md5-
auth=01724e4cab261d9828f1aef17afe9b66e8" (323)
11:23:37.307372 IP gally-reborn.sietch.lavoixdessansvoix.org.35140 > chu.sietch.lavoixdessansvoix.org.domain: 59845+ PTR?
251.0.0.224.in-addr.arpa. (42)
11:23:37.498527 IP chu.sietch.lavoixdessansvoix.org.domain > gally-reborn.sietch.lavoixdessansvoix.org.35140: 59845 NXDomain
0/1/0 (99)
11:23:37.498667 IP gally-reborn.sietch.lavoixdessansvoix.org.54881 > chu.sietch.lavoixdessansvoix.org.domain: 50566+ PTR?
253.160.168.192.in-addr.arpa. (46)
11:23:37.498949 IP chu.sietch.lavoixdessansvoix.org.domain > gally-reborn.sietch.lavoixdessansvoix.org.54881: 50566* 2/0/0 PTR
francisco.sietch.lavoixdessansvoix.org., PTR nexus.sietch.lavoixdessansvoix.org. (118)
11:23:37.499095 IP gally-reborn.sietch.lavoixdessansvoix.org.36483 > chu.sietch.lavoixdessansvoix.org.domain: 5920+ PTR?
254.160.168.192.in-addr.arpa. (46)
11:23:37.499332 IP chu.sietch.lavoixdessansvoix.org.domain > gally-reborn.sietch.lavoixdessansvoix.org.36483: 5920* 1/0/0 PTR
chu.sietch.lavoixdessansvoix.org. (92)
11:23:37.499422 IP gally-reborn.sietch.lavoixdessansvoix.org.45486 > chu.sietch.lavoixdessansvoix.org.domain: 37846+ PTR?
119.160.168.192.in-addr.arpa. (46)
11:23:39.040944 STP 802.1w, Rapid STP, Flags [Learn, Forward, Agreement], bridge-id 8000.24:b6:57:0f:53:da.8007, length 47
11:23:41.040903 STP 802.1w, Rapid STP, Flags [Learn, Forward, Agreement], bridge-id 8000.24:b6:57:0f:53:da.8007, length 47
11:23:42.301064 IP francisco.sietch.lavoixdessansvoix.org.mdns > 224.0.0.251.mdns: 0- [0q] 1/0/0 TXT "type=0" "version=1"
"refresh-age-timeout=0" "priority=0" "refresh-flag=0" "root-mac-address=24:b6:57:0f:53:da" "cost=0" "transm-
address=192.168.160.253" "transm-interface=300000" "voice-vlan-id=2" "voice-vlan-vpt=5" "voice-vlan-dscp=46" "md5-
auth=01724e4cab261d9828f1aef17afe9b66e8" (323)
11:23:43.040898 STP 802.1w, Rapid STP, Flags [Learn, Forward, Agreement], bridge-id 8000.24:b6:57:0f:53:da.8007, length 47
^C
14 packets captured
15 packets received by filter
1 packet dropped by kernel
```

5.1.2) Capturer 5 packets

```
$ sudo tcpdump -c 5 -i eth0
```

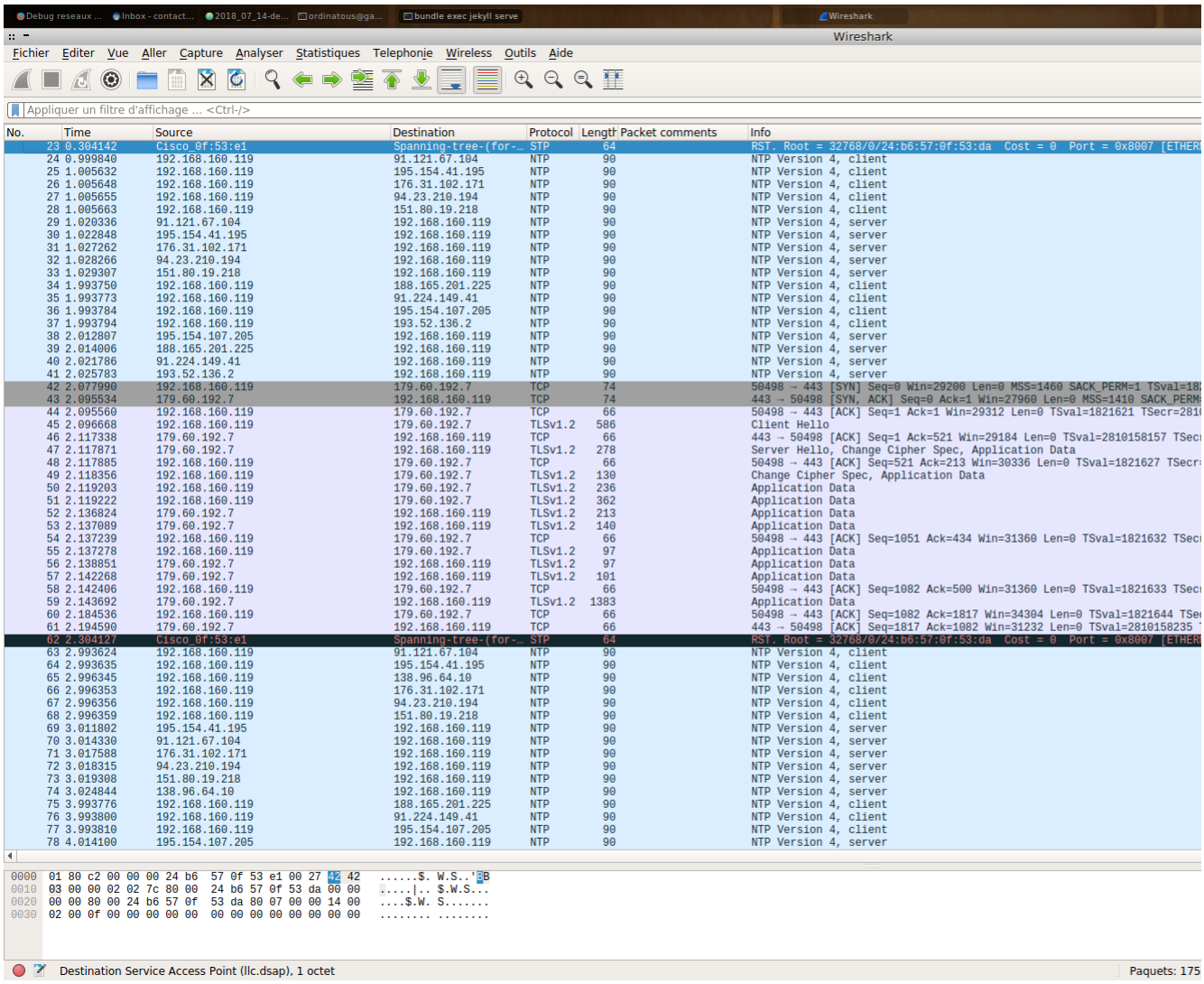
```
$ sudo tcpdump -c 5 -i eth0
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
11:25:45.039441 STP 802.1w, Rapid STP, Flags [Learn, Forward, Agreement], bridge-id 8000.24:b6:57:0f:53:da.8007, length 47
11:25:45.132827 IP gally-reborn.sietch.lavoixdessansvoix.org.38278 > tor-relay-005.parckwart.de.imaps: Flags [S], seq
1396906183, win 29200, options [mss 1460,sackOK,TS val 1774192 ecr 0,nop,wscale 7], length 0
11:25:45.133641 IP gally-reborn.sietch.lavoixdessansvoix.org.42740 > chu.sietch.lavoixdessansvoix.org.domain: 30710+ PTR?
80.21.249.173.in-addr.arpa. (44)
11:25:45.167027 IP tor-relay-005.parckwart.de.imaps > gally-reborn.sietch.lavoixdessansvoix.org.38278: Flags [S.], seq
838166629, ack 1396906184, win 28960, options [mss 1460,sackOK,TS val 88680858 ecr 1774192,nop,wscale 7], length 0
11:25:45.167077 IP gally-reborn.sietch.lavoixdessansvoix.org.38278 > tor-relay-005.parckwart.de.imaps: Flags [.], ack 1, win
229, options [nop,nop,TS val 1774201 ecr 88680858], length 0
5 packets captured
414 packets received by filter
405 packets dropped by kernel
```

5.1.3) Enregistrer une capture

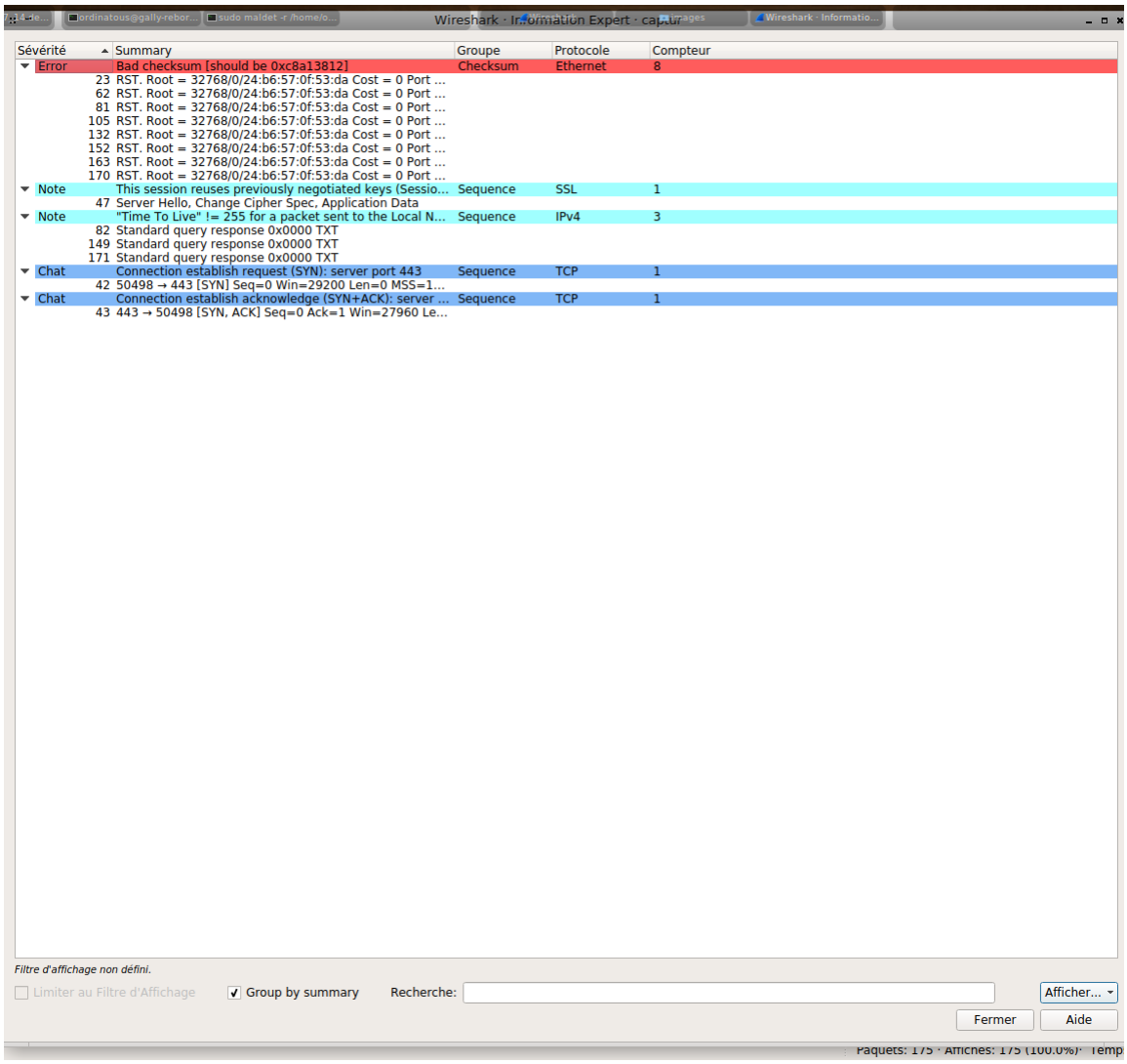
```
$ tcpdump -w captur.pacs -i eth0
```

5.2) Wireshark Utility

Wireshark... Outil indispensable. C'est l'environnement graphique de T-shark, qui permet de faire de la capture en live, d'enregistrer les captures, de lire plusieurs type de format de fichier de capture.
Sa grande force c'est son moteur de disécation, Wireshark capture l'intégralité des trames.
On va lire la capture précédente.



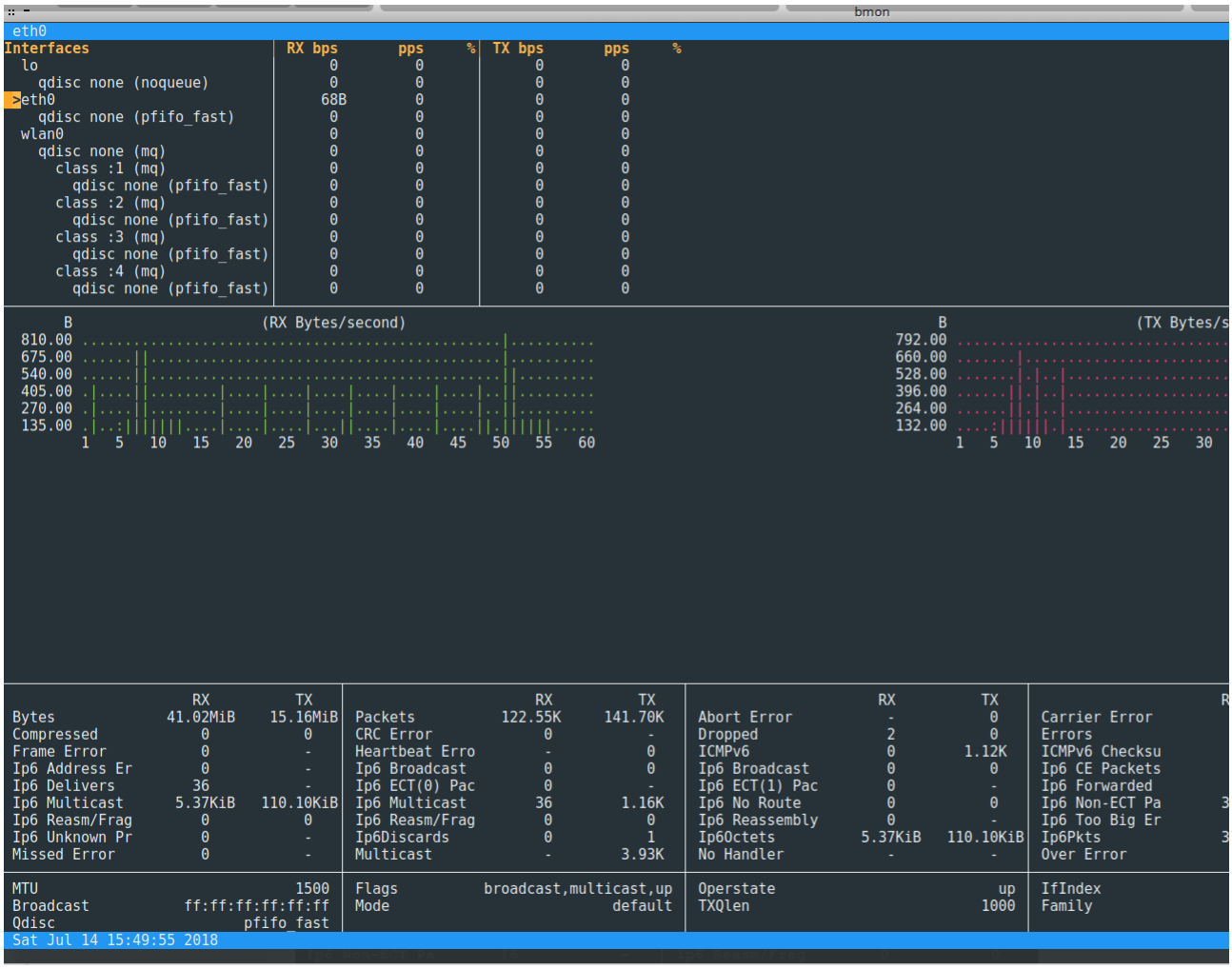
Rapidement on voit que *wireshark* colore les *packets* en fonction du *protocole*, c'est assez lisible.
Quand on sait ce que l'on cherche ça va, on va directement dans la trame qui nous intéresse.
Wireshark à une fonction *expert*, c'est le petit icone rouge en bas à gauche, celui-ci va nous ouvrir une console nous indiquant les *erreur, note* qu'il a trouvé. La couleur dépendra du niveau d'alerte.



C'est très pratique, néanmoins *wireshark* vous poussera vers les *RFC* et l'étude des trames et des protocoles.

5.3) bmon Tool

bmon est un outil de monitoring qui permet de surveiller la bande passante.



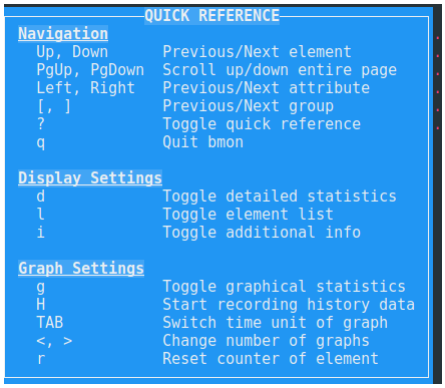
bmon

5.3.1) Installation

Il faut cloner le dépôt git du projet.

```
$ git clone https://github.com/tgraf/bmon.git
$ cd bmon
$ sudo apt-get install build-essential make libconfuse-dev libnl-3-dev libnl-route-3-dev libncurses-dev pkg-config dh-autoreconf
$ sudo ./autogen.sh
$ sudo ./configure
$ sudo make
$ sudo make install
```

Il dispose d'un petit menu d'aide : ?



bmon menu

bmon à plusieurs format de sortie: * curses: interface graphique * ascii: sans l'interface curses * format: pour etre exploiter par un autre programme

6) Parefeu

6.1) iptables Firewall

6.1.1) Lister les règles

Article plus complet dans iptables

```
sudo iptables -L -n -v
```

```
 sudo iptables -L -n -v
[sudo] Mot de passe de ordinatous :
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
```

6.1.2) Bloquer le ping en entré

En réalité , `DROP` ignore simplement ce protocole et n'enverra aucune réponse.

```
sudo iptables -A input --proto icmp -j DROP
```

```
 sudo iptables -A INPUT --proto icmp -j DROP
```

```
$ sudo iptables -L -n -v
Chain INPUT (policy ACCEPT 2 packets, 152 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 DROP       icmp --  *      *       0.0.0.0/0            0.0.0.0/0

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain OUTPUT (policy ACCEPT 2 packets, 152 bytes)
 pkts bytes target     prot opt in     out     source               destination
```

Cependant en cas de reboot , la règle disparait.

```
$ sudo iptables -L -n -v
[sudo] Mot de passe de ordinatous :
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
```

Avec iptable , il faut lister les règles dans un fichier `rules` qui est restauré au démarrage du système.

6.1.3) Persistence

➥ Modifions le port d'écoute du serveur ssh:

```
sudo sed -i "s/#Port 22/Port 1022/g" /etc/ssh/sshd_config
systemctl restart sshd.service
```

Créons des règles iptables : * pour conserver les connexions établies * pour bloquer *ssh* le port *22* * pour accepter *ssh* sur *1022* * pour bloquer les requête *ICMP* * pour bloquer le port de *telnet*.
Note: c'est règles sont totalement arbitraire et ne reflète pas une configuration complète de *iptables*, se référer à l'article consacré iptables

```
$ sudo iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
$ sudo iptables -A INPUT -p tcp --dport 22 -j DROP
$ sudo iptables -A INPUT -p tcp --dport 1022 -j ACCEPT
$ sudo iptables -A input --proto icmp -j DROP
$ sudo iptables -A INPUT -p tcp --dport 23 -j DROP
```

On revérifie , c'est la table INPUT qui nous intéresse: `sudo iptables -L -n -v`

```
Chain INPUT (policy ACCEPT 4 packets, 168 bytes)
 pkts bytes target     prot opt in     out     source               destination
 3344  240K f2b-SSH    tcp  --  *      *       0.0.0.0/0            0.0.0.0/0           tcp dpt:1022
77054   11M ACCEPT     all  --  *      *       0.0.0.0/0            0.0.0.0/0           state RELATED,ESTABLISHED
 2405  143K DROP       tcp  --  *      *       0.0.0.0/0            0.0.0.0/0           tcp dpt:22
  148  8880 ACCEPT     tcp  --  *      *       0.0.0.0/0            0.0.0.0/0           tcp dpt:1022
   40  4011 DROP       icmp --  *      *       0.0.0.0/0            0.0.0.0/0
    1    40 DROP       tcp  --  *      *       0.0.0.0/0            0.0.0.0/0           tcp dpt:23
```

On comprend ici que iptables: * connait les protocoles tcp/udp/icmp ... il reconnait aussi les étapes de *tcp* . * reconnait l'état d'une connexion RELATED/ESTABLISHED et sait donc les maintenir * travail avec d'autres application comme fail2ban

6.1.4) Script de restauration

Précédement nous avons entré nos commandes *iptables* une à une, il nous faut les sauver , puis indiquer que l'on souhaite restaurer la configuration lors d'un redémarrage (nuance: en réalité les règles sont restauré aprés le boot, qui lui contient dèjà une configuration au niveau du noyaux, avec *iptables*, noous utilisons donc un "logiciel").
Les commandes indiquent de : * sauver les règles * créer notre script *iptables* * le rendre executable * y ecrire l'entête du script (interpréteur) * y ecrire la commande de restauration (/sbin/iptables-restore) en lui passant les règles.

```
# iptables-save > /etc/iptables.up.rules
# touch /etc/network/if-pre-up.d/iptables
# chmod +x /etc/network/if-pre-up.d/iptables
# echo "#! /bin/bash" >> /etc/network/if-pre-up.d/iptables
# echo "/sbin/iptables-restore < /etc/iptables.up.rules" >> /etc/network/if-pre-up.d/iptables
```

6.1.4.1) Autre methode

En modifiant une variable du noyaux linux, attention il s'agit des configurations de bases inscritent en dur, il s'agit véritablement de votre système.
Nous pouvons consulter la configuration avec : * `# sysctl -a` * `# sysctl net.ipv4.icmp_echo_ignore_all`

```
# sysctl net.ipv4.icmp_echo_ignore_all
net.ipv4.icmp_echo_ignore_all = 0
```

Pour rendre un réglage persistant , il faut modifier `/etc/sysctl.conf`, et passer par exemple la valeur de `net.ipv4.icmp_echo_ignore_all` de 0 à 1 pour ignorer les pings.

6.2) firewalld

Principalement utilisé sur les systèmes CentOS

6.3) UFW

UncomplicatedFireWall est généralement utilisé sous les système Débian et dérivés afin de facilité la gestion du *parefeu*.