Version : **2022.01**

Dernière mise-à-jour : 2022/11/01 07:27

Topic 205: Networking Configuration

        - 2.3 - traceroute
    - LAB #3 - Connexions à Distance
        - 3.1 - Telnet
        - 3.2 - wget
        - 3.3 - ftp
        - 3.4 - SSH
            - Présentation
                - SSH-1
                - SSH-2
            - Authentification par mot de passe
            - Authentification par clef asymétrique
            - Configuration du Serveur
            - Configuration du Client
            - Tunnels SSH
        - 3.5 - SCP
            - Présentation
            - Utilisation
        - 3.6 - Mise en Place des Clefs Asymétriques
        - 3.7 - Services réseaux
            - inetd
            - TCP Wrapper
    - Configuration du Réseau sous RHEL/CentOS 6
        - Configuration de TCP/IP
            - DHCP
                - /etc/sysconfig/network
                - /etc/sysconfig/network-scripts/ifcfg-ethX (où X=0,1 ...)
            - IP Fixe
                - /etc/sysconfig/network
                - /etc/sysconfig/network-scripts/ifcfg-ethX (où X=0,1 ...)
            - La Commande hostname
            - La Commande ifconfig
            - Activer/Désactiver une Interface Manuellement
            - /etc/networks
            - Résolution d'adresses IP

        - /etc/resolv.conf
        - /etc/nsswitch.conf
        - /etc/hosts
    - Services réseaux
        - xinetd
        - TCP Wrapper
    - Routage Statique
        - La Commande route
        - Activer/désactiver le routage sur le serveur
- Configuration du Réseau sous RHEL/CentOS 7
    - La Commande nmcli
    - Connections et Profils
    - Ajouter une Deuxième Adresse IP à un Profil
    - La Commande hostname
    - La Commande ip
    - Activer/Désactiver une Interface Manuellement
    - Routage Statique
        - La commande ip
        - Activer/désactiver le routage sur le serveur
    - LAB #4 - Utilisation de nmap et de netcat
        - 4.1 - nmap
            - Installation
            - Utilisation
            - Fichiers de Configuration
            - Scripts
        - 4.2 - netcat
            - Utilisation
    - LAB #5 - Utilisation de tcpdump
        - 5.1 - Utilisation
            - L'option -i
            - L'option -x
            - L'option -X
            - L'option -w
            - L'option -v

- 5.2 - Filtrage à l'écoute
- LAB #6 - Mise en place d'un VPN avec OpenVPN
  - Présentation
  - Configuration commune au client et au serveur
  - Configuration du client
  - Configuration du serveur
  - Tests
    - Du client vers le serveur
    - Du serveur vers le client
- Annexe #1 - Comprendre les Réseaux
  - Présentation des Réseaux
  - Classification des Réseaux
    - Classification par Mode de Transmission
    - Classification par Topologie
      - La Topologie Physique
      - La Topologie en Ligne
      - La Topologie en Bus
      - La Topologie en Étoile
      - La Topologie en Anneau
      - La Topologie en Arbre
      - La Topologie Maillée
    - Classification par Étendue
    - Les Types de LAN
      - Réseau à Serveur Dédié
      - Réseau Poste-à-Poste
  - Le Modèle Client/Serveur
  - Modèles de Communication
    - Le modèle OSI
      - Les Couches
      - Les Protocoles
      - Les Interfaces
      - Protocol Data Units
      - Encapsulation et Désencapsulation
    - Spécification NDIS et le Modèle ODI

        - Le modèle TCP/IP
    - Les Raccordements
        - Les Modes de Transmission
        - Les Câbles
            - Le Câble Coaxial
            - Le Câble Paire Torsadée
            - Catégories de Blindage
            - La Prise RJ45
            - Channel Link et Basic Link
            - La Fibre Optique
        - Les Réseaux sans Fils
        - Le Courant Porteur en Ligne
    - Technologies
        - Ethernet
        - Token-Ring
    - Périphériques Réseaux Spéciaux
        - Les Concentrateurs
        - Les Répéteurs
        - Les Ponts
            - Le Pont de Base
            - Le Pont en Cascade
            - Le Pont en Dorsale
        - Les Commutateurs
        - Les Routeurs
        - Les Passerelles
- Annexe #2 - Comprendre TCP Version 4
    - En-tête TCP
    - En-tête UDP
    - Fragmentation et Ré-encapsulation
    - Adressage
    - Masques de sous-réseaux
    - VLSM
    - Ports et sockets
    - /etc/services

Configuration de TCP/IP sous Debian 6

La configuration TCP/IP se trouve dans le fichier **/etc/network/interfaces** :

DHCP

/etc/network/interfaces

```
root@debian6:~# cat /etc/network/interfaces
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
allow-hotplug eth0
#NetworkManager#iface eth0 inet dhcp
```

Dans ce fichier chaque déclaration est de la forme suivante :

```
interface   nom    type    mode
```

On peut constater donc dans notre exemple ci-dessus :

- une déclaration pour l'interface **lo** de loopback
- une déclaration pour l'interface **eth0** en dhcp

IP Fixe

Dans le cas où l'interface eth0 était configuré en IP statique, la déclaration concernant eth0 prendrait la forme suivante :

```
auto eth0
iface eth0 inet static
        address 10.0.2.15
        netmask 255.255.255.0
        broadcast 10.0.2.255
```

```
        network 10.0.2.0
        gateway 10.0.2.2
```

Dans ce fichier vous pouvez constater les directives suivantes :

Directive	Description
address	Indique l'adresse IPv4 de l'interface
netmask	Indique le masque de sous-réseau IPv4
broadcast	Indique l'adresse de diffusion IPv4
network	Indique l'adresse réseau IPv4
gateway	Indique l'adresse IPv4 de la passerelle par défaut

Notez que VirtualBox fournit une passerelle par défaut (10.0.2.2).

Après avoir modifier le fichier **/etc/network/interfaces** vous devez arrêter le service **network-manager** utilisé pour la connexion DHCP et activer le service **networking** :

```
root@debian6:~# service network-manager stop
Stopping network connection manager: NetworkManager.
root@debian6:~# update-rc.d -f network-manager remove
update-rc.d: using dependency based boot sequencing
root@debian6:~# chkconfig --level 2345 networking on
root@debian6:~# service networking start
Configuring network interfaces...done.
```

Si le service networking réfuse de démarrer en produisant une erreur, le problème vient certainement du fait que votre interface réseau a été configurée par **udev** en **eth1**. La solution la plus simple est d'éditer le fichier **/etc/udev/rules.d/70-persistent-net.rules** en supprimant toutes les lignes qui ne commencent pas par le caractère **#** et de re-démarrer votre machine virtuelle.

La Commande hostname

Le nom de la machine se trouve dans le fichier **/etc/hostname** :

```
root@debian6:~# cat /etc/hostname
debian6
```

Ce nom doit être un FQDN (*Fully Qualified Domain Name*). Modifiez donc ce fichier ainsi :

```
root@debian6:~# cat /etc/hostname
debian6.fenestros.loc
```

Afin d'informer le système immédiatement de la modification du FQDN, utilisez la commande **hostname** :

```
root@debian6:~# hostname
debian6
root@debian6:~# hostname debian6.fenestros.loc
root@debian6:~# hostname
debian6.fenestros.loc
```

Pour afficher le FQDN du système vous pouvez également utiliser la commande suivante :

```
root@debian6:~# uname -n
debian6.fenestros.loc
```

Options de la commande hostname

Les options de cette commande sont :

```
root@debian6:~# hostname --help
Usage: hostname [-v] [-b] {hostname|-F file}         set host name (from file)
```

```
       hostname [-v] [-d|-f|-s|-a|-i|-y|-A|-I]           display formatted name
       hostname [-v]                                  display host name

       {yp,nis,}domainname [-v] {nisdomain|-F file}  set NIS domain name (from file)
       {yp,nis,}domainname [-v]                       display NIS domain name

       dnsdomainname [-v]                             display dns domain name

       hostname -V|--version|-h|--help                print info and exit

Program name:
       {yp,nis,}domainname=hostname -y
       dnsdomainname=hostname -d

Program options:
    -s, --short            short host name
    -a, --alias            alias names
    -i, --ip-address       addresses for the host name
    -I, --all-ip-addresses all addresses for the host
    -f, --fqdn, --long     long host name (FQDN)
    -A, --all-fqdns        all long host names (FQDNs)
    -d, --domain           DNS domain name
    -y, --yp, --nis        NIS/YP domain name
    -b, --boot             set default hostname if none available
    -F, --file             read host name or NIS domain name from given file

Description:
   This command can get or set the host name or the NIS domain name. You can
   also get the DNS domain or the FQDN (fully qualified domain name).
   Unless you are using bind or NIS for host lookups you can change the
   FQDN (Fully Qualified Domain Name) and the DNS domain name (which is
   part of the FQDN) in the /etc/hosts file.
```

La Commande ifconfig

Pour afficher la configuration IP de la machine il faut saisir la commande suivante :

```
root@debian6:~# ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:2a:02:5c
          inet adr:10.0.2.15  Bcast:10.0.2.255  Masque:255.255.255.0
          adr inet6: fe80::a00:27ff:fe2a:25c/64 Scope:Lien
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:990 errors:0 dropped:0 overruns:0 frame:0
          TX packets:580 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 lg file transmission:1000
          RX bytes:684107 (668.0 KiB)  TX bytes:97392 (95.1 KiB)

lo        Link encap:Boucle locale
          inet adr:127.0.0.1  Masque:255.0.0.0
          adr inet6: ::1/128 Scope:Hôte
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:8 errors:0 dropped:0 overruns:0 frame:0
          TX packets:8 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 lg file transmission:0
          RX bytes:560 (560.0 B)  TX bytes:560 (560.0 B)
```

La commande ifconfig est également utilisée pour configurer une interface.

Créez maintenant une interface fictive ainsi :

```
root@debian6:~# ifconfig eth0:1 192.168.1.2 netmask 255.255.255.0 broadcast 192.168.1.255
```

Constatez maintenant le résultat :

```
root@debian:~# ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:2a:02:5c
```

```
          inet adr:10.0.2.15  Bcast:10.0.2.255  Masque:255.255.255.0
          adr inet6: fe80::a00:27ff:fe2a:25c/64 Scope:Lien
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1013 errors:0 dropped:0 overruns:0 frame:0
          TX packets:611 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 lg file transmission:1000
          RX bytes:686171 (670.0 KiB)  TX bytes:100060 (97.7 KiB)

eth0:1    Link encap:Ethernet  HWaddr 08:00:27:2a:02:5c
          inet adr:192.168.1.2  Bcast:192.168.1.255  Masque:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1

lo        Link encap:Boucle locale
          inet adr:127.0.0.1  Masque:255.0.0.0
          adr inet6: ::1/128 Scope:Hôte
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:8 errors:0 dropped:0 overruns:0 frame:0
          TX packets:8 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 lg file transmission:0
          RX bytes:560 (560.0 B)  TX bytes:560 (560.0 B)
```

Options de la commande ifconfig

Les options de cette commande sont :

```
root@debian:~# ifconfig --help
Usage:
  ifconfig [-a] [-v] [-s] <interface> [[<AF>] <address>]
  [add <adresse>[/<lg_prefixe>]]
  [del <adresse>[/<lg_prefixe>]]
  [[-]broadcast [<adresse>]]  [[-]pointopoint [<adresse>]]
  [netmask <address>]  [dstaddr <address>]  [tunnel <address>]
  [outfill <NN>] [keepalive <NN>]
```

```
  [hw <HW> <adresse>]  [metric <NN>]  [mtu <NN>]
  [[-]trailers]  [[-]arp]  [[-]allmulti]
  [multicast]  [[-]promisc]
  [mem_start <NN>]  [io_addr <NN>]  [irq <NN>]  [media <type>]
  [txqueuelen <NN>]
  [[-]dynamic]
  [up|down] ...

  <HW>=Type de matériel.
  Liste des types de matériels possibles:
    loop (Boucle locale) slip (IP ligne série) cslip (IP ligne série - VJ )
    slip6 (IP ligne série - 6 bits) cslip6 (IP ligne série - 6 bits VJ) adaptive (IP ligne série adaptative)
    strip (Metricom Starmode IP) ash (Ash) ether (Ethernet)
    tr (16/4 Mbps Token Ring) tr (16/4 Mbps Token Ring (New)) ax25 (AMPR AX.25)
    netrom (AMPR NET/ROM) rose (AMPR ROSE) tunnel (IPIP Tunnel)
    ppp (Protocole Point-à-Point) hdlc ((Cisco)-HDLC) lapb (LAPB)
    arcnet (ARCnet) dlci (Frame Relay DLCI) frad (Périphériue d'accès Frame Relay)
    sit (IPv6-dans-IPv4) fddi (Fiber Distributed Data Interface) hippi (HIPPI)
    irda (IrLAP) ec (Econet) x25 (generic X.25)
    eui64 (Generic EUI-64)
  <AF>=famille d'Adresses. Défaut: inet
  Liste des familles d'adresses possibles:
    unix (Domaine UNIX) inet (DARPA Internet) inet6 (IPv6)
    ax25 (AMPR AX.25) netrom (AMPR NET/ROM) rose (AMPR ROSE)
    ipx (Novell IPX) ddp (Appletalk DDP) ec (Econet)
    ash (Ash) x25 (CCITT X.25)
```

Activer/Désactiver une Interface Manuellement

Deux commandes existent pour activer et désactiver manuellement une interface réseau :

```
root@debian6:~# ifdown eth0
root@debian6:~# ifconfig
```

```
lo        Link encap:Boucle locale
          inet adr:127.0.0.1  Masque:255.0.0.0
          adr inet6: ::1/128 Scope:Hôte
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:8 errors:0 dropped:0 overruns:0 frame:0
          TX packets:8 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 lg file transmission:0
          RX bytes:560 (560.0 B)  TX bytes:560 (560.0 B)

root@debian6:~# ifup eth0
root@debian6:~# ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:2a:02:5c
          inet adr:10.0.2.15  Bcast:10.0.2.255  Masque:255.255.255.0
          adr inet6: fe80::a00:27ff:fe2a:25c/64 Scope:Lien
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:2556 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1632 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 lg file transmission:1000
          RX bytes:2893516 (2.7 MiB)  TX bytes:176291 (172.1 KiB)

eth0:1    Link encap:Ethernet  HWaddr 08:00:27:2a:02:5c
          inet adr:192.168.1.2  Bcast:192.168.1.255  Masque:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1

lo        Link encap:Boucle locale
          inet adr:127.0.0.1  Masque:255.0.0.0
          adr inet6: ::1/128 Scope:Hôte
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:8 errors:0 dropped:0 overruns:0 frame:0
          TX packets:8 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 lg file transmission:0
          RX bytes:560 (560.0 B)  TX bytes:560 (560.0 B)
```

/etc/networks

Ce fichier contient la correspondance entre des noms de réseaux et l'adresse IP du réseau :

```
root@debian6:~# cat /etc/networks
default     0.0.0.0
loopback    127.0.0.0
link-local  169.254.0.0
```

Résolution d'adresses IP

La configuration DNS est stockée dans le fichier **/etc/resolv.conf**.

/etc/resolv.conf

La configuration DNS est stockée dans le fichier /etc/resolv.conf :

```
root@debian:~# cat /etc/resolv.conf
nameserver 8.8.8.8
nameserver 8.8.4.4
```

Notez que les DNS utilisés sont les serveurs DNS publics de Google.

/etc/nsswitch.conf

L'ordre de recherche des services de noms est stocké dans le fichier **/etc/nsswitch.conf**. Pour connaître l'ordre, saisissez la commande suivante :

```
root@debian6:~# grep '^hosts:' /etc/nsswitch.conf
hosts:          files mdns4_minimal [NOTFOUND=return] dns mdns4
```

/etc/hosts

Le mot **files** dans la sortie de la commande précédente fait référence au fichier **/etc/hosts** :

```
root@debian6:~# cat /etc/hosts
127.0.0.1   localhost
127.0.1.1   debian6.fenestros.loc    debian6

# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
```

Pour tester le serveur DNS, deux commandes sont possibles :

```
root@debian6:~# nslookup www.linuxelearning.com
Server:     8.8.8.8
Address:    8.8.8.8#53

Non-authoritative answer:
www.linuxelearning.com  canonical name = linuxelearning.com.
Name:   linuxelearning.com
Address: 212.198.31.61

root@debian6:~# dig www.linuxelearning.com

; <<>> DiG 9.7.3 <<>> www.linuxelearning.com
```

```
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 45521
;; flags: qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;www.linuxelearning.com.        IN    A

;; ANSWER SECTION:
www.linuxelearning.com. 42847    IN    CNAME    linuxelearning.com.
linuxelearning.com. 60    IN    A    212.198.31.61

;; Query time: 51 msec
;; SERVER: 8.8.8.8#53(8.8.8.8)
;; WHEN: Wed May  9 15:47:18 2012
;; MSG SIZE  rcvd: 70
```

LAB #1 - Configuration du Réseau sous Debian 11

Debian 11 utilise **Network Manager** pour gérer le réseau. Network Manager est composé de deux éléments :

- un service qui gère les connexions réseaux et rapporte leurs états,
- des front-ends qui passent par un API de configuration du service.

Important : Notez qu'avec cette version de NetworkManager, IPv6 est activée par défaut.

Le service NetworkManager doit toujours être lancé :

```
root@debian11:~# systemctl status NetworkManager.service
● NetworkManager.service - Network Manager
```

```
     Loaded: loaded (/lib/systemd/system/NetworkManager.service; enabled; vendor pres>
     Active: active (running) since Sun 2022-05-01 18:00:05 CEST; 20h ago
       Docs: man:NetworkManager(8)
   Main PID: 499 (NetworkManager)
      Tasks: 3 (limit: 4632)
     Memory: 13.3M
        CPU: 1.811s
     CGroup: /system.slice/NetworkManager.service
             └─499 /usr/sbin/NetworkManager --no-daemon

May 01 18:00:05 debian11 NetworkManager[499]: <info>  [1651420805.4957] device (ens18>
May 01 18:00:05 debian11 NetworkManager[499]: <info>  [1651420805.5102] device (ens18>
May 01 18:00:05 debian11 NetworkManager[499]: <info>  [1651420805.5136] device (ens18>
May 01 18:00:05 debian11 NetworkManager[499]: <info>  [1651420805.5141] device (ens18>
May 01 18:00:05 debian11 NetworkManager[499]: <info>  [1651420805.5147] manager: Netw>
May 01 18:00:05 debian11 NetworkManager[499]: <info>  [1651420805.5157] manager: Netw>
May 01 18:00:05 debian11 NetworkManager[499]: <info>  [1651420805.5159] policy: set '>
May 01 18:00:05 debian11 NetworkManager[499]: <info>  [1651420805.6555] device (ens18>
May 01 18:00:05 debian11 NetworkManager[499]: <info>  [1651420805.6577] manager: Netw>
May 01 18:00:05 debian11 NetworkManager[499]: <info>  [1651420805.6604] manager: star>
lines 1-21/21 (END)
[q]
```

La commande **nmcli** (Network Manager Command Line Interface) est utilisée pour configurer NetworkManager.

Les options et les sous-commandes peuvent être consultées en utilisant les commandes suivantes :

```
root@debian11:~# nmcli help
Usage: nmcli [OPTIONS] OBJECT { COMMAND | help }

OPTIONS
  -a, --ask                                ask for missing parameters
  -c, --colors auto|yes|no                 whether to use colors in output
  -e, --escape yes|no                      escape columns separators in values
```

```
  -f, --fields <field,...>|all|common      specify fields to output
  -g, --get-values <field,...>|all|common  shortcut for -m tabular -t -f
  -h, --help                               print this help
  -m, --mode tabular|multiline             output mode
  -o, --overview                           overview mode
  -p, --pretty                             pretty output
  -s, --show-secrets                       allow displaying passwords
  -t, --terse                              terse output
  -v, --version                            show program version
  -w, --wait <seconds>                     set timeout waiting for finishing operations

OBJECT
  g[eneral]       NetworkManager's general status and operations
  n[etworking]    overall networking control
  r[adio]         NetworkManager radio switches
  c[onnection]    NetworkManager's connections
  d[evice]        devices managed by NetworkManager
  a[gent]         NetworkManager secret agent or polkit agent
  m[onitor]       monitor NetworkManager changes
```

1.1 - Connections et Profils

NetworkManager inclus la notion de **connections** ou **profils** permettant des configurations différentes en fonction de la localisation. Pour voir les connections actuelles, utilisez la commande **nmcli c** avec la sous-commande **show** :

```
root@debian11:~# nmcli c show
NAME                UUID                                  TYPE      DEVICE
Wired connection 1  77c569e6-3176-4c10-8008-40d7634d2504  ethernet  ens18
```

Créez donc un profil IP fixe rattaché au périphérique **ens18** :

```
root@debian11:~# nmcli connection add con-name ip_fixe ifname ens18 type ethernet ip4 10.0.2.41/24 gw4 10.0.2.1
```

```
Connection 'ip_fixe' (c52994fc-0918-4108-81d2-d86dade62c7a) successfully added.
```

Constatez sa présence :

```
root@debian11:~# nmcli c show
NAME                UUID                                  TYPE      DEVICE
Wired connection 1  77c569e6-3176-4c10-8008-40d7634d2504  ethernet  ens18
ip_fixe             c52994fc-0918-4108-81d2-d86dade62c7a  ethernet  --
```

Notez que la sortie n'indique pas que le profil **ip_fixe** soit associé au periphérique **ens18** car le profil **ip_fixe** n'est pas activé :

```
root@debian11:~# nmcli d show
GENERAL.DEVICE:                         ens18
GENERAL.TYPE:                           ethernet
GENERAL.HWADDR:                         F6:35:D1:39:09:72
GENERAL.MTU:                            1500
GENERAL.STATE:                          100 (connected)
GENERAL.CONNECTION:                     Wired connection 1
GENERAL.CON-PATH:                       /org/freedesktop/NetworkManager/ActiveConnect>
WIRED-PROPERTIES.CARRIER:               on
IP4.ADDRESS[1]:                         10.0.2.40/24
IP4.GATEWAY:                            10.0.2.1
IP4.ROUTE[1]:                           dst = 10.0.2.0/24, nh = 0.0.0.0, mt = 100
IP4.ROUTE[2]:                           dst = 0.0.0.0/0, nh = 10.0.2.1, mt = 100
IP4.DNS[1]:                             8.8.8.8
IP4.DNS[2]:                             8.8.4.4
IP6.ADDRESS[1]:                         fe80::f435:d1ff:fe39:972/64
IP6.GATEWAY:                            --
IP6.ROUTE[1]:                           dst = fe80::/64, nh = ::, mt = 100
IP6.ROUTE[2]:                           dst = ff00::/8, nh = ::, mt = 256, table=255

GENERAL.DEVICE:                         lo
GENERAL.TYPE:                           loopback
GENERAL.HWADDR:                         00:00:00:00:00:00
```

```
GENERAL.MTU:                            65536
lines 1-23
[q]
```

Pour activer le profil ip_fixe, utilisez la commande suivante :

```
[root@centos8 ~]# nmcli connection up ip_fixe
```

Notez que votre terminal est bloqué à cause du changement de l'adresse IP.

> **A faire** - Revenez à votre Gateway et re-connectez-vous à la VM en tant que trainee en utilisant l'adresse IP 10.0.2.41.

Le profil ip_fixe est maintenant activé tandis que le profil enp0s3 a été désactivé :

```
root@debian11:~# nmcli c show
NAME                UUID                                  TYPE      DEVICE
ip_fixe             c52994fc-0918-4108-81d2-d86dade62c7a  ethernet  ens18
Wired connection 1  77c569e6-3176-4c10-8008-40d7634d2504  ethernet  --
root@debian11:~# nmcli d show
GENERAL.DEVICE:                         ens18
GENERAL.TYPE:                           ethernet
GENERAL.HWADDR:                         F6:35:D1:39:09:72
GENERAL.MTU:                            1500
GENERAL.STATE:                          100 (connected)
GENERAL.CONNECTION:                     ip_fixe
GENERAL.CON-PATH:                       /org/freedesktop/NetworkManager/ActiveC>
WIRED-PROPERTIES.CARRIER:               on
IP4.ADDRESS[1]:                         10.0.2.41/24
IP4.GATEWAY:                            10.0.2.1
IP4.ROUTE[1]:                           dst = 10.0.2.0/24, nh = 0.0.0.0, mt = 1>
```

```
IP4.ROUTE[2]:                           dst = 0.0.0.0/0, nh = 10.0.2.1, mt = 100
IP6.ADDRESS[1]:                         fe80::7958:e23f:31e:62cd/64
IP6.GATEWAY:                            --
IP6.ROUTE[1]:                           dst = fe80::/64, nh = ::, mt = 100
IP6.ROUTE[2]:                           dst = ff00::/8, nh = ::, mt = 256, tabl>

GENERAL.DEVICE:                         lo
GENERAL.TYPE:                           loopback
lines 1-19
[q]
```

Pour consulter les paramètres du profil **Wired connection 1**, utilisez la commande suivante :

```
root@debian11:~# nmcli -p connection show "Wired connection 1"
===============================================================================
                Connection profile details (Wired connection 1)
===============================================================================
connection.id:                          Wired connection 1
connection.uuid:                        77c569e6-3176-4c10-8008-40d7634d2504
connection.stable-id:                   --
connection.type:                        802-3-ethernet
connection.interface-name:              --
connection.autoconnect:                 yes
connection.autoconnect-priority:        0
connection.autoconnect-retries:         -1 (default)
connection.multi-connect:               0 (default)
connection.auth-retries:                -1
connection.timestamp:                   1651494383
connection.read-only:                   no
connection.permissions:                 --
connection.zone:                        --
connection.master:                      --
connection.slave-type:                  --
connection.autoconnect-slaves:          -1 (default)
```

```
connection.secondaries:                 --
connection.gateway-ping-timeout:        0
connection.metered:                     unknown
connection.lldp:                        default
connection.mdns:                        -1 (default)
connection.llmnr:                       -1 (default)
connection.wait-device-timeout:         -1
---------------------------------------------------------------------------
802-3-ethernet.port:                    --
802-3-ethernet.speed:                   0
802-3-ethernet.duplex:                  --
802-3-ethernet.auto-negotiate:          no
802-3-ethernet.mac-address:             --
802-3-ethernet.cloned-mac-address:      --
802-3-ethernet.generate-mac-address-mask:--
802-3-ethernet.mac-address-blacklist:   --
802-3-ethernet.mtu:                     auto
802-3-ethernet.s390-subchannels:        --
802-3-ethernet.s390-nettype:            --
802-3-ethernet.s390-options:            --
802-3-ethernet.wake-on-lan:             default
802-3-ethernet.wake-on-lan-password:    --
---------------------------------------------------------------------------
ipv4.method:                            manual
ipv4.dns:                               8.8.8.8,8.8.4.4
ipv4.dns-search:                        --
ipv4.dns-options:                       --
ipv4.dns-priority:                      0
ipv4.addresses:                         10.0.2.40/24
ipv4.gateway:                           10.0.2.1
ipv4.routes:                            --
ipv4.route-metric:                      -1
ipv4.route-table:                       0 (unspec)
ipv4.routing-rules:                     --
```

```
ipv4.ignore-auto-routes:                no
ipv4.ignore-auto-dns:                   no
lines 1-56
[q]
```

De même, pour consulter les paramètres du profil **ip_fixe**, utilisez la commande suivante :

```
root@debian11:~# nmcli -p connection show ip_fixe
===============================================================================
                    Connection profile details (ip_fixe)
===============================================================================
connection.id:                          ip_fixe
connection.uuid:                        c52994fc-0918-4108-81d2-d86dade62c7a
connection.stable-id:                   --
connection.type:                        802-3-ethernet
connection.interface-name:              ens18
connection.autoconnect:                 yes
connection.autoconnect-priority:        0
connection.autoconnect-retries:         -1 (default)
connection.multi-connect:               0 (default)
connection.auth-retries:                -1
connection.timestamp:                   1651496105
connection.read-only:                   no
connection.permissions:                 --
connection.zone:                        --
connection.master:                      --
connection.slave-type:                  --
connection.autoconnect-slaves:          -1 (default)
connection.secondaries:                 --
connection.gateway-ping-timeout:        0
connection.metered:                     unknown
connection.lldp:                        default
connection.mdns:                        -1 (default)
connection.llmnr:                       -1 (default)
```

```
connection.wait-device-timeout:         -1
------------------------------------------------------------------------------
802-3-ethernet.port:                    --
802-3-ethernet.speed:                   0
802-3-ethernet.duplex:                  --
802-3-ethernet.auto-negotiate:          no
802-3-ethernet.mac-address:             --
802-3-ethernet.cloned-mac-address:      --
802-3-ethernet.generate-mac-address-mask:--
802-3-ethernet.mac-address-blacklist:   --
802-3-ethernet.mtu:                     auto
802-3-ethernet.s390-subchannels:        --
802-3-ethernet.s390-nettype:            --
802-3-ethernet.s390-options:            --
802-3-ethernet.wake-on-lan:             default
802-3-ethernet.wake-on-lan-password:    --
------------------------------------------------------------------------------
ipv4.method:                            manual
ipv4.dns:                               --
ipv4.dns-search:                        --
ipv4.dns-options:                       --
ipv4.dns-priority:                      0
ipv4.addresses:                         10.0.2.41/24
ipv4.gateway:                           10.0.2.1
ipv4.routes:                            --
ipv4.route-metric:                      -1
ipv4.route-table:                       0 (unspec)
ipv4.routing-rules:                     --
ipv4.ignore-auto-routes:                no
ipv4.ignore-auto-dns:                   no
lines 1-56
[q]
```

Pour consulter la liste profils associés à un périphérique, utilisez la commande suivante :

```
root@debian11:~# nmcli -f CONNECTIONS device show ens18
CONNECTIONS.AVAILABLE-CONNECTION-PATHS: /org/freedesktop/NetworkManager/Settings/1,/o>
CONNECTIONS.AVAILABLE-CONNECTIONS[1]:   77c569e6-3176-4c10-8008-40d7634d2504 | Wired >
CONNECTIONS.AVAILABLE-CONNECTIONS[2]:   c52994fc-0918-4108-81d2-d86dade62c7a | ip_fixe
lines 1-3/3 (END)
[q]
```

Les fichiers de configuration pour le periphérique **ens18** se trouvent dans le répertoire **/etc/NetworkManager/system-connections/** :

```
root@debian11:~# ls -l /etc/NetworkManager/system-connections
total 8
-rw------- 1 root root 284 May  2 14:23  ip_fixe.nmconnection
-rw------- 1 root root 249 Apr 25 07:01 'Wired connection 1'
```

1.2 - Résolution des Noms

L'étude du fichier **/etc/NetworkManager/system-connections/ip_fixe.nmconnection** démontre l'abscence de directives concernant les DNS :

```
root@debian11:~# cat /etc/NetworkManager/system-connections/ip_fixe.nmconnection
[connection]
id=ip_fixe
uuid=c52994fc-0918-4108-81d2-d86dade62c7a
type=ethernet
interface-name=ens18
permissions=

[ethernet]
mac-address-blacklist=

[ipv4]
address1=10.0.2.41/24,10.0.2.1
dns-search=
```

```
method=manual

[ipv6]
addr-gen-mode=stable-privacy
dns-search=
method=auto

[proxy]
```

La résolution des noms est donc inactive :

```
root@debian11:~# ping www.free.fr
ping: www.free.fr: Temporary failure in name resolution
```

Modifiez donc la configuration du profil **ip_fixe** :

```
root@debian11:~# nmcli connection mod ip_fixe ipv4.dns 8.8.8.8
```

L'étude du fichier **/etc/NetworkManager/system-connections/ip_fixe.nmconnection** démontre que la directive concernant le serveur DNS a été ajoutée :

```
root@debian11:~# cat /etc/NetworkManager/system-connections/ip_fixe.nmconnection
[connection]
id=ip_fixe
uuid=c52994fc-0918-4108-81d2-d86dade62c7a
type=ethernet
interface-name=ens18
permissions=
timestamp=1651499105

[ethernet]
mac-address-blacklist=

[ipv4]
```

```
address1=10.0.2.41/24,10.0.2.1
dns=8.8.8.8;
dns-search=
method=manual

[ipv6]
addr-gen-mode=stable-privacy
dns-search=
method=auto

[proxy]
```

Afin que la modification du serveur DNS soit prise en compte, re-démarrez le service NetworkManager :

```
root@debian11:~# systemctl restart NetworkManager.service
```

Vérifiez que le fichier **/etc/resolv.conf** ait été modifié par NetworkManager :

```
root@debian11:~# cat /etc/resolv.conf
# Generated by NetworkManager
nameserver 8.8.8.8
```

Dernièrement vérifiez la resolution des noms :

```
root@debian11:~# ping www.free.fr
PING www.free.fr (212.27.48.10) 56(84) bytes of data.
64 bytes from www.free.fr (212.27.48.10): icmp_seq=1 ttl=47 time=10.8 ms
64 bytes from www.free.fr (212.27.48.10): icmp_seq=2 ttl=47 time=11.1 ms
^C
--- www.free.fr ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1002ms
rtt min/avg/max/mdev = 10.804/10.931/11.058/0.127 ms
```

Important : Notez qu'il existe un front-end graphique en mode texte, **nmtui**, pour configurer NetworkManager.

1.3 - Ajouter une Deuxième Adresse IP à un Profil

Pour ajouter une deuxième adresse IP à un profil sous Debian 11, il convient d'utiliser la commande suivante :

```
root@debian11:~# nmcli connection mod ip_fixe +ipv4.addresses 192.168.1.2/24
```

Rechargez la configuration du profil :

```
root@debian11:~# nmcli con up ip_fixe
Connection successfully activated (D-Bus active path: /org/freedesktop/NetworkManager/ActiveConnection/2)
```

Saisissez ensuite la commande suivante :

```
root@debian11:~# nmcli connection show ip_fixe
connection.id:                          ip_fixe
connection.uuid:                        c52994fc-0918-4108-81d2-d86dade62c7a
connection.stable-id:                   --
connection.type:                        802-3-ethernet
connection.interface-name:              ens18
connection.autoconnect:                 yes
connection.autoconnect-priority:        0
connection.autoconnect-retries:         -1 (default)
connection.multi-connect:               0 (default)
connection.auth-retries:                -1
connection.timestamp:                   1651499367
connection.read-only:                   no
```

```
connection.permissions:                 --
connection.zone:                        --
connection.master:                      --
connection.slave-type:                  --
connection.autoconnect-slaves:          -1 (default)
connection.secondaries:                 --
connection.gateway-ping-timeout:        0
connection.metered:                     unknown
connection.lldp:                        default
connection.mdns:                        -1 (default)
connection.llmnr:                       -1 (default)
connection.wait-device-timeout:         -1
802-3-ethernet.port:                    --
802-3-ethernet.speed:                   0
802-3-ethernet.duplex:                  --
802-3-ethernet.auto-negotiate:          no
802-3-ethernet.mac-address:             --
802-3-ethernet.cloned-mac-address:      --
802-3-ethernet.generate-mac-address-mask:--
802-3-ethernet.mac-address-blacklist:   --
802-3-ethernet.mtu:                     auto
802-3-ethernet.s390-subchannels:        --
802-3-ethernet.s390-nettype:            --
802-3-ethernet.s390-options:            --
802-3-ethernet.wake-on-lan:             default
802-3-ethernet.wake-on-lan-password:    --
ipv4.method:                            manual
ipv4.dns:                               8.8.8.8
ipv4.dns-search:                        --
ipv4.dns-options:                       --
ipv4.dns-priority:                      0
ipv4.addresses:                         10.0.2.41/24, 192.168.1.2/24
ipv4.gateway:                           10.0.2.1
ipv4.routes:                            --
```

```
ipv4.route-metric:                      -1
ipv4.route-table:                       0 (unspec)
ipv4.routing-rules:                     --
ipv4.ignore-auto-routes:                no
ipv4.ignore-auto-dns:                   no
ipv4.dhcp-client-id:                    --
ipv4.dhcp-iaid:                         --
ipv4.dhcp-timeout:                      0 (default)
ipv4.dhcp-send-hostname:                yes
ipv4.dhcp-hostname:                     --
lines 1-56
[Space Bar]
IP4.ADDRESS[1]:                         10.0.2.41/24
IP4.ADDRESS[2]:                         192.168.1.2/24
IP4.GATEWAY:                            10.0.2.1
IP4.ROUTE[1]:                           dst = 10.0.2.0/24, nh = 0.0.0.0, mt = 100
IP4.ROUTE[2]:                           dst = 192.168.1.0/24, nh = 0.0.0.0, mt = 100
IP4.ROUTE[3]:                           dst = 0.0.0.0/0, nh = 10.0.2.1, mt = 100
IP4.DNS[1]:                             8.8.8.8
lines 57-112
[q]
```

Important : Notez l'ajout de l'adresse secondaire à la ligne **ipv4.addresses:** ainsi que l'ajout de la ligne **IP4.ADDRESS[2]:**.

Consultez maintenant le contenu du fichier **/etc/NetworkManager/system-connections/ip_fixe.nmconnection** :

```
root@debian11:~# cat /etc/NetworkManager/system-connections/ip_fixe.nmconnection
[connection]
id=ip_fixe
uuid=c52994fc-0918-4108-81d2-d86dade62c7a
```

```
type=ethernet
interface-name=ens18
permissions=
timestamp=1651499263

[ethernet]
mac-address-blacklist=

[ipv4]
address1=10.0.2.41/24,10.0.2.1
address2=192.168.1.2/24
dns=8.8.8.8;
dns-search=
method=manual

[ipv6]
addr-gen-mode=stable-privacy
dns-search=
method=auto

[proxy]
```

Important : Notez l'ajout de la ligne **address2=192.168.1.2/24**.

1.4 - La Commande hostname

La procédure de la modification du hostname est simplifiée et sa prise en compte est immédiate :

```
root@debian11:~# hostname
debian11
```

```
root@debian11:~# nmcli general hostname debian11.ittraining.loc

root@debian11:~# cat /etc/hostname
debian11.ittraining.loc

root@debian11:~# hostname
debian11.ittraining.loc
```

1.5 - La Commande ip

Sous Debian 11 la commande **ip** est préférée par rapport à la commande ifconfig :

```
root@debian11:~# ip address
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    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: ens18: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether f6:35:d1:39:09:72 brd ff:ff:ff:ff:ff:ff
    altname enp0s18
    inet 10.0.2.41/24 brd 10.0.2.255 scope global noprefixroute ens18
       valid_lft forever preferred_lft forever
    inet 192.168.1.2/24 brd 192.168.1.255 scope global noprefixroute ens18
       valid_lft forever preferred_lft forever
    inet6 fe80::7958:e23f:31e:62cd/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
```

En cas de besoin, pour extraire les adresses IP de cette sortie, utilisez les commandes suivantes :

```
root@debian11:~# ip addr show ens18 | grep "inet" | grep -v "inet6" | awk '{ print $2; }' | sed 's/\/.*$//'
10.0.2.41
```

```
192.168.1.2
```

Les options de cette commande sont :

```
root@debian11:~# ip --help
Usage: ip [ OPTIONS ] OBJECT { COMMAND | help }
       ip [ -force ] -batch filename
where  OBJECT := { link | address | addrlabel | route | rule | neigh | ntable |
                   tunnel | tuntap | maddress | mroute | mrule | monitor | xfrm |
                   netns | l2tp | fou | macsec | tcp_metrics | token | netconf | ila |
                   vrf | sr | nexthop | mptcp }
       OPTIONS := { -V[ersion] | -s[tatistics] | -d[etails] | -r[esolve] |
                    -h[uman-readable] | -iec | -j[son] | -p[retty] |
                    -f[amily] { inet | inet6 | mpls | bridge | link } |
                    -4 | -6 | -I | -D | -M | -B | -0 |
                    -l[oops] { maximum-addr-flush-attempts } | -br[ief] |
                    -o[neline] | -t[imestamp] | -ts[hort] | -b[atch] [filename] |
                    -rc[vbuf] [size] | -n[etns] name | -N[umeric] | -a[ll] |
                    -c[olor]}
```

1.6 - Activer/Désactiver une Interface Manuellement

Deux commandes existent pour désactiver et activer manuellement une interface réseau :

```
# nmcli device disconnect enp0s3
# nmcli device connect enp0s3
```

Important : Veuillez ne **PAS** exécuter ces deux commandes.

1.7 - Routage Statique

La commande ip

Sous Debian 11, pour supprimer la route vers le réseau 192.168.1.0 il convient d'utiliser la commande ip et non pas la commande route :

```
root@debian11:~# ip route
default via 10.0.2.1 dev ens18 proto static metric 100
10.0.2.0/24 dev ens18 proto kernel scope link src 10.0.2.41 metric 100
192.168.1.0/24 dev ens18 proto kernel scope link src 192.168.1.2 metric 100

root@debian11:~# ip route del 192.168.1.0/24 via 0.0.0.0

root@debian11:~# ip route
default via 10.0.2.1 dev ens18 proto static metric 100
10.0.2.0/24 dev ens18 proto kernel scope link src 10.0.2.41 metric 100
```

Pour ajouter la route vers le réseau 192.168.1.0 :

```
root@debian11:~# ip route add 192.168.1.0/24 via 10.0.2.1

root@debian11:~# ip route
default via 10.0.2.1 dev ens18 proto static metric 100
10.0.2.0/24 dev ens18 proto kernel scope link src 10.0.2.41 metric 100
192.168.1.0/24 via 10.0.2.1 dev ens18
```

Important - La commande utilisée pour ajouter une passerelle par défaut prend la forme suivante **ip route add default via *adresse ip***.

Activer le routage sur le serveur

Pour activer le routage sur le serveur, il convient d'activer la retransmission des paquets:

```
root@debian11:~# echo 1 > /proc/sys/net/ipv4/ip_forward
root@debian11:~# cat /proc/sys/net/ipv4/ip_forward
1
```

LAB #2 - Diagnostique du Réseau

2.1 - ping

Pour tester l'accessibilité d'une machine, vous devez utiliser la commande **ping** :

```
root@debian11:~# ping -c4 10.0.2.1
PING 10.0.2.1 (10.0.2.1) 56(84) bytes of data.
64 bytes from 10.0.2.1: icmp_seq=1 ttl=64 time=0.184 ms
64 bytes from 10.0.2.1: icmp_seq=2 ttl=64 time=0.167 ms
64 bytes from 10.0.2.1: icmp_seq=3 ttl=64 time=0.168 ms
^C
--- 10.0.2.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2036ms
rtt min/avg/max/mdev = 0.167/0.173/0.184/0.007 ms
```

Les options de cette commande sont :

```
root@debian11:~# ping --help
ping: invalid option -- '-'

Usage
  ping [options] <destination>
```

```
Options:
  <destination>      dns name or ip address
  -a                 use audible ping
  -A                 use adaptive ping
  -B                 sticky source address
  -c <count>         stop after <count> replies
  -D                 print timestamps
  -d                 use SO_DEBUG socket option
  -f                 flood ping
  -h                 print help and exit
  -I <interface>     either interface name or address
  -i <interval>      seconds between sending each packet
  -L                 suppress loopback of multicast packets
  -l <preload>       send <preload> number of packages while waiting replies
  -m <mark>          tag the packets going out
  -M <pmtud opt>     define mtu discovery, can be one of <do|dont|want>
  -n                 no dns name resolution
  -O                 report outstanding replies
  -p <pattern>       contents of padding byte
  -q                 quiet output
  -Q <tclass>        use quality of service <tclass> bits
  -s <size>          use <size> as number of data bytes to be sent
  -S <size>          use <size> as SO_SNDBUF socket option value
  -t <ttl>           define time to live
  -U                 print user-to-user latency
  -v                 verbose output
  -V                 print version and exit
  -w <deadline>      reply wait <deadline> in seconds
  -W <timeout>       time to wait for response

IPv4 options:
  -4                 use IPv4
  -b                 allow pinging broadcast
  -R                 record route
```

```
  -T <timestamp>     define timestamp, can be one of <tsonly|tsandaddr|tsprespec>

IPv6 options:
  -6                 use IPv6
  -F <flowlabel>     define flow label, default is random
  -N <nodeinfo opt>  use icmp6 node info query, try <help> as argument

For more details see ping(8).
```

2.2 - netstat -i

Pour visualiser les statistiques réseaux, vous disposez de la commande **netstat** :

```
root@debian11:~# netstat -i
-bash: netstat: command not found

root@debian11:~# apt -y install net-tools

root@debian11:~# netstat -i
Kernel Interface table
Iface      MTU    RX-OK RX-ERR RX-DRP RX-OVR    TX-OK TX-ERR TX-DRP TX-OVR Flg
ens18     1500     7861      0      0 0          7299      0      0      0 BMRU
lo       65536       82      0      0 0            82      0      0      0 LRU
```

Les options de cette commande sont :

```
root@debian11:~# netstat --help
usage: netstat [-vWeenNcCF] [<Af>] -r         netstat {-V|--version|-h|--help}
       netstat [-vWnNcaeol] [<Socket> ...]
       netstat { [-vWeenNac] -i | [-cnNe] -M | -s [-6tuw] }

        -r, --route              display routing table
        -i, --interfaces         display interface table
```

```
       -g, --groups              display multicast group memberships
       -s, --statistics          display networking statistics (like SNMP)
       -M, --masquerade          display masqueraded connections

       -v, --verbose             be verbose
       -W, --wide                don't truncate IP addresses
       -n, --numeric             don't resolve names
       --numeric-hosts           don't resolve host names
       --numeric-ports           don't resolve port names
       --numeric-users           don't resolve user names
       -N, --symbolic            resolve hardware names
       -e, --extend              display other/more information
       -p, --programs            display PID/Program name for sockets
       -o, --timers              display timers
       -c, --continuous          continuous listing

       -l, --listening           display listening server sockets
       -a, --all                 display all sockets (default: connected)
       -F, --fib                 display Forwarding Information Base (default)
       -C, --cache               display routing cache instead of FIB
       -Z, --context             display SELinux security context for sockets

<Socket>={-t|--tcp} {-u|--udp} {-U|--udplite} {-S|--sctp} {-w|--raw}
         {-x|--unix} --ax25 --ipx --netrom
<AF>=Use '-6|-4' or '-A <af>' or '--<af>'; default: inet
List of possible address families (which support routing):
  inet (DARPA Internet) inet6 (IPv6) ax25 (AMPR AX.25)
  netrom (AMPR NET/ROM) ipx (Novell IPX) ddp (Appletalk DDP)
  x25 (CCITT X.25)
```

2.3 - traceroute

La commande ping est à la base de la commande **traceroute**. Cette commande sert à découvrir la route empruntée pour accéder à un site donné :

```
root@debian11:~# traceroute www.free.fr
traceroute to www.free.fr (212.27.48.10), 30 hops max, 60 byte packets
 1  10.0.2.1 (10.0.2.1)  0.476 ms  0.440 ms  0.419 ms
 2  51.68.180.253 (51.68.180.253)  2.730 ms  2.786 ms  2.927 ms
 3  10.161.41.253 (10.161.41.253)  0.315 ms 10.161.41.252 (10.161.41.252)  0.293 ms  0.274 ms
 4  10.17.242.66 (10.17.242.66)  0.467 ms 10.17.242.62 (10.17.242.62)  0.620 ms 10.17.242.66 (10.17.242.66)
0.795 ms
 5  10.73.40.38 (10.73.40.38)  0.189 ms  2.403 ms 10.73.40.42 (10.73.40.42)  2.339 ms
 6  10.73.249.68 (10.73.249.68)  4.894 ms * *
 7  fra-fr5-sbb1-nc5.de.eu (91.121.215.116)  1.909 ms  1.491 ms fra-fr5-sbb2-nc5.de.eu (94.23.122.246)  1.475 ms
 8  be101.sbg-g1-nc5.fr.eu (94.23.122.136)  4.182 ms be101.sbg-g2-nc5.fr.eu (91.121.215.196)  4.504 ms  4.918 ms
 9  be103.par-gsw-sbb1-nc5.fr.eu (91.121.215.219)  10.471 ms be103.par-th2-sbb1-nc5.fr.eu (94.23.122.139)  10.448
ms  10.198 ms
10  10.200.2.65 (10.200.2.65)  10.174 ms 10.200.2.71 (10.200.2.71)  10.211 ms 10.200.2.65 (10.200.2.65)  10.111
ms
11  * * *
12  194.149.166.61 (194.149.166.61)  10.289 ms *  10.111 ms
13  * * *
14  * * *
15  * * *
16  * * *
17  * * *
18  * * *
19  * * *
20  * * *
21  * * *
22  * * *
23  * * *
24  * * *
25  * * *
26  * * *
27  * * *
28  * * *
29  * * *
```

```
30  * * *
```

Les options de cette commande sont :

```
root@debian11:~# traceroute --help
Usage:
  traceroute [ -46dFITnreAUDV ] [ -f first_ttl ] [ -g gate,... ] [ -i device ] [ -m max_ttl ] [ -N squeries ] [ -
p port ] [ -t tos ] [ -l flow_label ] [ -w MAX,HERE,NEAR ] [ -q nqueries ] [ -s src_addr ] [ -z sendwait ] [ --
fwmark=num ] host [ packetlen ]
Options:
  -4                          Use IPv4
  -6                          Use IPv6
  -d  --debug                 Enable socket level debugging
  -F  --dont-fragment         Do not fragment packets
  -f first_ttl  --first=first_ttl
                              Start from the first_ttl hop (instead from 1)
  -g gate,...  --gateway=gate,...
                              Route packets through the specified gateway
                              (maximum 8 for IPv4 and 127 for IPv6)
  -I  --icmp                  Use ICMP ECHO for tracerouting
  -T  --tcp                   Use TCP SYN for tracerouting (default port is 80)
  -i device  --interface=device
                              Specify a network interface to operate with
  -m max_ttl  --max-hops=max_ttl
                              Set the max number of hops (max TTL to be
                              reached). Default is 30
  -N squeries  --sim-queries=squeries
                              Set the number of probes to be tried
                              simultaneously (default is 16)
  -n                          Do not resolve IP addresses to their domain names
  -p port  --port=port        Set the destination port to use. It is either
                              initial udp port value for "default" method
                              (incremented by each probe, default is 33434), or
                              initial seq for "icmp" (incremented as well,
```

```
                              default from 1), or some constant destination
                              port for other methods (with default of 80 for
                              "tcp", 53 for "udp", etc.)
-t tos  --tos=tos             Set the TOS (IPv4 type of service) or TC (IPv6
                              traffic class) value for outgoing packets
-l flow_label  --flowlabel=flow_label
                              Use specified flow_label for IPv6 packets
-w MAX,HERE,NEAR  --wait=MAX,HERE,NEAR
                              Wait for a probe no more than HERE (default 3)
                              times longer than a response from the same hop,
                              or no more than NEAR (default 10) times than some
                              next hop, or MAX (default 5.0) seconds (float
                              point values allowed too)
-q nqueries  --queries=nqueries
                              Set the number of probes per each hop. Default is
                              3
-r                            Bypass the normal routing and send directly to a
                              host on an attached network
-s src_addr  --source=src_addr
                              Use source src_addr for outgoing packets
-z sendwait  --sendwait=sendwait
                              Minimal time interval between probes (default 0).
                              If the value is more than 10, then it specifies a
                              number in milliseconds, else it is a number of
                              seconds (float point values allowed too)
-e  --extensions              Show ICMP extensions (if present), including MPLS
-A  --as-path-lookups         Perform AS path lookups in routing registries and
                              print results directly after the corresponding
                              addresses
-M name  --module=name        Use specified module (either builtin or external)
                              for traceroute operations. Most methods have
                              their shortcuts (`-I' means `-M icmp' etc.)
-O OPTS,...  --options=OPTS,...
                              Use module-specific option OPTS for the
```

```
                              traceroute module. Several OPTS allowed,
                              separated by comma. If OPTS is "help", print info
                              about available options
  --sport=num                 Use source port num for outgoing packets. Implies
                              `-N 1'
  --fwmark=num                Set firewall mark for outgoing packets
  -U  --udp                   Use UDP to particular port for tracerouting
                              (instead of increasing the port per each probe),
                              default port is 53
  -UL                         Use UDPLITE for tracerouting (default dest port
                              is 53)
  -D  --dccp                  Use DCCP Request for tracerouting (default port
                              is 33434)
  -P prot  --protocol=prot    Use raw packet of protocol prot for tracerouting
  --mtu                       Discover MTU along the path being traced. Implies
                              `-F -N 1'
  --back                      Guess the number of hops in the backward path and
                              print if it differs
  -V  --version               Print version info and exit
  --help                      Read this help and exit

Arguments:
+     host          The host to traceroute to
      packetlen     The full packet length (default is the length of an IP
                    header plus 40). Can be ignored or increased to a minimal
                    allowed value
```

LAB #3 - Connexions à Distance

3.1 - Telnet

La commande **telnet** est utilisée pour établir une connexion à distance avec un serveur telnet :

```
  # telnet numero_ip
```

> **Important** - Le service telnet revient à une redirection des canaux standards d'entrée et de sortie. Notez que la connexion n'est **pas** sécurisée. Pour fermer la connexion, il faut saisir la commande **exit**. La commande telnet n'offre pas de services de transfert de fichiers. Pour cela, il convient d'utiliser la command **ftp**.

Les options de cette commande sont :

```
root@debian11:~# which telnet
/usr/bin/telnet
root@debian11:~# telnet --help
telnet: invalid option -- '-'
Usage: telnet [-4] [-6] [-8] [-E] [-L] [-a] [-d] [-e char] [-l user]
        [-n tracefile] [ -b addr ] [-r] [host-name [port]]
```

3.2 - wget

La commande **wget** est utilisée pour récupérer un fichier via http, https ou ftp :

```
root@debian11:~# wget https://www.dropbox.com/s/wk79lkfr6f12u9j/wget_file.txt
--2022-05-03 10:07:50--  https://www.dropbox.com/s/wk79lkfr6f12u9j/wget_file.txt
Resolving www.dropbox.com (www.dropbox.com)... 162.125.67.18, 2620:100:6023:18::a27d:4312
Connecting to www.dropbox.com (www.dropbox.com)|162.125.67.18|:443... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: /s/raw/wk79lkfr6f12u9j/wget_file.txt [following]
--2022-05-03 10:07:51--  https://www.dropbox.com/s/raw/wk79lkfr6f12u9j/wget_file.txt
Reusing existing connection to www.dropbox.com:443.
HTTP request sent, awaiting response... 302 Found
Location:
```

```
https://uc64dcd84ee4be2a0c2f111bd2ab.dl.dropboxusercontent.com/cd/0/inline/BkjB7WuIg8oRJsAnsjaq0YeQrof8LaM6svUdaz
V8cmiN5MWJqHVYZlMfbR33nIokof9ZHmK551ixoOU8CgteUybofFdo_vD62VcF-
WNi4ACD8o6JsjdD7kiiuge-8mfbD1unwz4UtpyHIuVPQ9sUrh0QNNTVtsj3kcjDJAgEDg-uKQ/file# [following]
--2022-05-03 10:07:51--
https://uc64dcd84ee4be2a0c2f111bd2ab.dl.dropboxusercontent.com/cd/0/inline/BkjB7WuIg8oRJsAnsjaq0YeQrof8LaM6svUdaz
V8cmiN5MWJqHVYZlMfbR33nIokof9ZHmK551ixoOU8CgteUybofFdo_vD62VcF-
WNi4ACD8o6JsjdD7kiiuge-8mfbD1unwz4UtpyHIuVPQ9sUrh0QNNTVtsj3kcjDJAgEDg-uKQ/file
Resolving uc64dcd84ee4be2a0c2f111bd2ab.dl.dropboxusercontent.com
(uc64dcd84ee4be2a0c2f111bd2ab.dl.dropboxusercontent.com)... 162.125.67.15, 2620:100:6023:15::a27d:430f
Connecting to uc64dcd84ee4be2a0c2f111bd2ab.dl.dropboxusercontent.com
(uc64dcd84ee4be2a0c2f111bd2ab.dl.dropboxusercontent.com)|162.125.67.15|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 46 [text/plain]
Saving to: ‘wget_file.txt’

wget_file.txt         100%[======================>]      46  --.-KB/s    in 0s

2022-05-03 10:07:51 (26.8 MB/s) - ‘wget_file.txt’ saved [46/46]

root@debian11:~# cat wget_file.txt
This is a file retrieved by the wget command.
```

Les options de cette commande sont :

```
root@debian11:~# wget --help | more
GNU Wget 1.21, a non-interactive network retriever.
Usage: wget [OPTION]... [URL]...

Mandatory arguments to long options are mandatory for short options too.

Startup:
  -V,  --version                   display the version of Wget and exit
  -h,  --help                      print this help
  -b,  --background                go to background after startup
```

```
  -e,  --execute=COMMAND           execute a `.wgetrc'-style command

Logging and input file:
  -o,  --output-file=FILE          log messages to FILE
  -a,  --append-output=FILE        append messages to FILE
  -d,  --debug                     print lots of debugging information
  -q,  --quiet                     quiet (no output)
  -v,  --verbose                   be verbose (this is the default)
  -nv, --no-verbose                turn off verboseness, without being quiet
       --report-speed=TYPE         output bandwidth as TYPE.  TYPE can be bits
  -i,  --input-file=FILE           download URLs found in local or external FILE
  -F,  --force-html                treat input file as HTML
  -B,  --base=URL                  resolves HTML input-file links (-i -F)
                                     relative to URL
       --config=FILE               specify config file to use
       --no-config                 do not read any config file
       --rejected-log=FILE         log reasons for URL rejection to FILE

Download:
  -t,  --tries=NUMBER              set number of retries to NUMBER (0 unlimits)
       --retry-connrefused         retry even if connection is refused
       --retry-on-http-error=ERRORS    comma-separated list of HTTP errors to retry
  -O,  --output-document=FILE      write documents to FILE
  -nc, --no-clobber                skip downloads that would download to
                                     existing files (overwriting them)
       --no-netrc                  don't try to obtain credentials from .netrc
  -c,  --continue                  resume getting a partially-downloaded file
       --start-pos=OFFSET          start downloading from zero-based position OFFSET
       --progress=TYPE             select progress gauge type
       --show-progress             display the progress bar in any verbosity mode
  -N,  --timestamping              don't re-retrieve files unless newer than
                                     local
       --no-if-modified-since      don't use conditional if-modified-since get
                                     requests in timestamping mode
```

```
       --no-use-server-timestamps  don't set the local file's timestamp by
                                     the one on the server
  -S,  --server-response           print server response
       --spider                    don't download anything
  -T,  --timeout=SECONDS           set all timeout values to SECONDS
       --dns-timeout=SECS          set the DNS lookup timeout to SECS
       --connect-timeout=SECS      set the connect timeout to SECS
       --read-timeout=SECS         set the read timeout to SECS
  -w,  --wait=SECONDS              wait SECONDS between retrievals
                                     (applies if more then 1 URL is to be retrieved)
--More--
[q]
```

3.3 - ftp

Important - Si la commande **ftp** n'est pas installée sous Debian 11, installez-le à l'aide de la commande **apt install ftp** en tant que root.

La commande **ftp** est utilisée pour le transfert de fichiers. Une fois connecté, il convient d'utiliser la commande **help** pour afficher la liste des commandes disponibles :

```
ftp> help
Commands may be abbreviated.  Commands are:

!               dir             mdelete         qc              site
$               disconnect      mdir            sendport        size
account         exit            mget            put             status
append          form            mkdir           pwd             struct
ascii           get             mls             quit            system
bell            glob            mode            quote           sunique
binary          hash            modtime         recv            tenex
```

```
bye             help            mput            reget           tick
case            idle            newer           rstatus         trace
cd              image           nmap            rhelp           type
cdup            ipany           nlist           rename          user
chmod           ipv4            ntrans          reset           umask
close           ipv6            open            restart         verbose
cr              lcd             prompt          rmdir           ?
delete          ls              passive         runique
debug           macdef          proxy           send
ftp>
```

Le caractère **!** permet d'exécuter une commande sur la machine cliente

```
ftp> !pwd
/root
```

Pour transférer un fichier vers le serveur, il convient d'utiliser la commande **put** :

```
ftp> put nom_fichier_local nom_fichier_distant
```

Vous pouvez également transférer plusieurs fichiers à la fois grâce à la commande **mput**. Dans ce cas précis, il convient de saisir la commande suivante:

```
ftp> mput nom*.*
```

Pour transférer un fichier du serveur, il convient d'utiliser la commande **get** :

```
ftp> get nom_fichier
```

Vous pouvez également transférer plusieurs fichiers à la fois grâce à la commande **mget** (voir la commande **mput** ci-dessus).

Pour supprimer un fichier sur le serveur, il convient d'utiliser la commande **del** :

```
ftp> del nom_fichier
```

Pour fermer la session, il convient d'utiliser la commande **quit** :

```
ftp> quit
root@debian11:~#
```

3.4 - SSH

Présentation

La commande **ssh** est le successeur et la remplaçante de la commande **rlogin**. Il permet d'établir des connexions sécurisées avec une machine distante. SSH comporte cinq acteurs :

- Le **serveur SSH**
  - le démon sshd, qui s'occupe des authentifications et autorisations des clients,
- Le **client SSH**
  - ssh ou scp, qui assure la connexion et le dialogue avec le serveur,
- La **session** qui représente la connexion courante et qui commence juste après l'authentification réussie,
- Les **clefs**
  - **Couple de clef utilisateur asymétriques** et persistantes qui assurent l'identité d'un utilisateur et qui sont stockés sur disque dur,
  - **Clef hôte asymétrique et persistante** garantissant l'identité du serveur er qui est conservé sur disque dur
  - **Clef serveur asymétrique et temporaire** utilisée par le protocole SSH1 qui sert au chiffrement de la clé de session,
  - **Clef de session symétrique qui est générée aléatoirement** et qui permet le chiiffrement de la communication entre le client et le serveur. Elle est détruite en fin de session. SSH-1 utilise une seule clef tandis que SSH-2 utilise une clef par direction de la communication,
- La **base de données des hôtes connus** qui stocke les clés des connexions précédentes.

SSH fonctionne de la manière suivante pour la la mise en place d'un canal sécurisé:

- Le client contacte le serveur sur son port 22,
- Les client et le serveur échangent leur version de SSH. En cas de non-compatibilité de versions, l'un des deux met fin au processus,
- Le serveur SSH s'identifie auprès du client en lui fournissant :

    - Sa clé hôte,
    - Sa clé serveur,
    - Une séquence aléatoire de huit octets à inclure dans les futures réponses du client,
    - Une liste de méthodes de chiffrage, compression et authentification,
- Le client et le serveur produisent un identifiant identique, un haché MD5 long de 128 bits contenant la clé hôte, la clé serveur et la séquence aléatoire,
- Le client génère sa clé de session symétrique et la chiffre deux fois de suite, une fois avec la clé hôte du serveur et la deuxième fois avec la clé serveur. Le client envoie cette clé au serveur accompagnée de la séquence aléatoire et un choix d'algorithmes supportés,
- Le serveur déchiffre la clé de session,
- Le client et le serveur mettent en place le canal sécurisé.

SSH-1

SSH-1 utilise une paire de clefs de type RSA1. Il assure l'intégrité des données par une **Contrôle de Redondance Cyclique** (CRC) et est un bloc dit **monolithique**.

Afin de s'identifier, le client essaie chacune des six méthodes suivantes :

- **Kerberos**,
- **Rhosts**,
- **RhostsRSA**,
- Par **clef asymétrique**,
- **TIS**,
- Par **mot de passe**.

SSH-2

SSH-2 utilise **DSA** ou **RSA**. Il assure l'intégrité des données par l'algorithme **HMAC**. SSH-2 est organisé en trois **couches** :

- **SSH-TRANS** – Transport Layer Protocol,
- **SSH-AUTH** – Authentification Protocol,
- **SSH-CONN** – Connection Protocol.

SSH-2 diffère de SSH-1 essentiellement dans la phase authentification.

Trois méthodes d'authentification :

- Par **clef asymétrique**,
  - Identique à SSH-1 sauf avec l'algorithme DSA,
- **RhostsRSA**,
- Par **mot de passe**.

Les options de cette commande sont :

```
root@debian11:~# ssh --help
unknown option -- -
usage: ssh [-46AaCfGgKkMNnqsTtVvXxYy] [-B bind_interface]
           [-b bind_address] [-c cipher_spec] [-D [bind_address:]port]
           [-E log_file] [-e escape_char] [-F configfile] [-I pkcs11]
           [-i identity_file] [-J [user@]host[:port]] [-L address]
           [-l login_name] [-m mac_spec] [-O ctl_cmd] [-o option] [-p port]
           [-Q query_option] [-R address] [-S ctl_path] [-W host:port]
           [-w local_tun[:remote_tun]] destination [command]
```

Authentification par mot de passe

L'utilisateur fournit un mot de passe au client ssh. Le client ssh le transmet de façon sécurisée au serveur ssh puis le serveur vérifie le mot de passe et l'accepte ou non.

Avantage:

- Aucune configuration de clef asymétrique n'est nécessaire.

Inconvénients:

- L'utilisateur doit fournir à chaque connexion un identifiant et un mot de passe,
- Moins sécurisé qu'un système par clef asymétrique.

Authentification par clef asymétrique

- Le **client** envoie au serveur une requête d'authentification par clé asymétrique qui contient le module de la clé à utiliser,
- Le **serveur** recherche une correspondance pour ce module dans le fichier des clés autorisés **~/.ssh/authorized_keys**,
  - Dans le cas où une correspondance n'est pas trouvée, le serveur met fin à la communication,
  - Dans le cas contraire le serveur génère une chaîne aléatoire de 256 bits appelée un **challenge** et la chiffre avec la **clé publique du client**,
- Le **client** reçoit le challenge et le décrypte avec la partie privée de sa clé. Il combine le challenge avec l'identifiant de session et chiffre le résultat. Ensuite il envoie le résultat chiffré au serveur.
- Le **serveur** génère le même haché et le compare avec celui reçu du client. Si les deux hachés sont identiques, l'authentification est réussie.

Configuration du Serveur

La configuration du serveur s'effectue dans le fichier **/etc/ssh/sshd_config** :

```
root@debian11:~# cat /etc/ssh/sshd_config
#       $OpenBSD: sshd_config,v 1.103 2018/04/09 20:41:22 tj Exp $

# This is the sshd server system-wide configuration file.  See
# sshd_config(5) for more information.

# This sshd was compiled with PATH=/usr/bin:/bin:/usr/sbin:/sbin

# The strategy used for options in the default sshd_config shipped with
# OpenSSH is to specify options with their default value where
# possible, but leave them commented.  Uncommented options override the
# default value.

Include /etc/ssh/sshd_config.d/*.conf

#Port 22
#AddressFamily any
```

```
#ListenAddress 0.0.0.0
#ListenAddress ::

#HostKey /etc/ssh/ssh_host_rsa_key
#HostKey /etc/ssh/ssh_host_ecdsa_key
#HostKey /etc/ssh/ssh_host_ed25519_key

# Ciphers and keying
#RekeyLimit default none

# Logging
#SyslogFacility AUTH
#LogLevel INFO

# Authentication:

#LoginGraceTime 2m
PermitRootLogin yes
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10

#PubkeyAuthentication yes

# Expect .ssh/authorized_keys2 to be disregarded by default in future.
#AuthorizedKeysFile     .ssh/authorized_keys .ssh/authorized_keys2

#AuthorizedPrincipalsFile none

#AuthorizedKeysCommand none
#AuthorizedKeysCommandUser nobody

# For this to work you will also need host keys in /etc/ssh/ssh_known_hosts
#HostbasedAuthentication no
```

```
# Change to yes if you don't trust ~/.ssh/known_hosts for
# HostbasedAuthentication
#IgnoreUserKnownHosts no
# Don't read the user's ~/.rhosts and ~/.shosts files
#IgnoreRhosts yes

# To disable tunneled clear text passwords, change to no here!
#PasswordAuthentication yes
#PermitEmptyPasswords no

# Change to yes to enable challenge-response passwords (beware issues with
# some PAM modules and threads)
ChallengeResponseAuthentication no

# Kerberos options
#KerberosAuthentication no
#KerberosOrLocalPasswd yes
#KerberosTicketCleanup yes
#KerberosGetAFSToken no

# GSSAPI options
#GSSAPIAuthentication no
#GSSAPICleanupCredentials yes
#GSSAPIStrictAcceptorCheck yes
#GSSAPIKeyExchange no

# Set this to 'yes' to enable PAM authentication, account processing,
# and session processing. If this is enabled, PAM authentication will
# be allowed through the ChallengeResponseAuthentication and
# PasswordAuthentication.  Depending on your PAM configuration,
# PAM authentication via ChallengeResponseAuthentication may bypass
# the setting of "PermitRootLogin without-password".
# If you just want the PAM account and session checks to run without
# PAM authentication, then enable this but set PasswordAuthentication
```

```
# and ChallengeResponseAuthentication to 'no'.
UsePAM yes

#AllowAgentForwarding yes
#AllowTcpForwarding yes
#GatewayPorts no
X11Forwarding yes
#X11DisplayOffset 10
#X11UseLocalhost yes
#PermitTTY yes
PrintMotd no
#PrintLastLog yes
#TCPKeepAlive yes
#PermitUserEnvironment no
#Compression delayed
#ClientAliveInterval 0
#ClientAliveCountMax 3
#UseDNS no
#PidFile /var/run/sshd.pid
#MaxStartups 10:30:100
#PermitTunnel no
#ChrootDirectory none
#VersionAddendum none

# no default banner path
#Banner none

# Allow client to pass locale environment variables
AcceptEnv LANG LC_*

# override default of no subsystems
Subsystem       sftp    /usr/lib/openssh/sftp-server

# Example of overriding settings on a per-user basis
```

```
#Match User anoncvs
#       X11Forwarding no
#       AllowTcpForwarding no
#       PermitTTY no
#       ForceCommand cvs server
```

Pour ôter les lignes de commentaires dans ce fichier, utilisez la commande suivante :

```
root@debian11:~# cd /tmp ; grep -E -v '^(#|$)'  /etc/ssh/sshd_config > sshd_config
root@debian11:/tmp# cat sshd_config
Include /etc/ssh/sshd_config.d/*.conf
PermitRootLogin yes
ChallengeResponseAuthentication no
UsePAM yes
X11Forwarding yes
PrintMotd no
AcceptEnv LANG LC_*
Subsystem       sftp    /usr/lib/openssh/sftp-server
```

Pour sécuriser le serveur ssh, ajoutez ou modifiez les directives suivantes :

```
AllowGroups adm
Banner /etc/issue.net
HostbasedAuthentication no
IgnoreRhosts yes
LoginGraceTime 60
LogLevel INFO
PermitEmptyPasswords no
PermitRootLogin no
PrintLastLog yes
Protocol 2
StrictModes yes
X11Forwarding no
```

Votre fichier ressemblera à celui-ci :

```
root@debian11:/tmp# vi sshd_config
root@debian11:/tmp# cat sshd_config
AllowGroups adm
Banner /etc/issue.net
HostbasedAuthentication no
IgnoreRhosts yes
LoginGraceTime 60
LogLevel INFO
PermitEmptyPasswords no
PermitRootLogin no
PrintLastLog yes
Protocol 2
StrictModes yes
X11Forwarding no
Include /etc/ssh/sshd_config.d/*.conf
ChallengeResponseAuthentication no
UsePAM yes
PrintMotd no
AcceptEnv LANG LC_*
Subsystem       sftp    /usr/lib/openssh/sftp-server
```

Renommez le fichier **/etc/ssh/sshd_config** en **/etc/ssh/sshd_config.old** :

```
[root@centos11 tmp]# cp /etc/ssh/sshd_config /etc/ssh/sshd_config.old
```

Copiez le fichier **/tmp/sshd_config** vers **/etc/ssh/** :

```
root@debian11:/tmp# cp /tmp/sshd_config /etc/ssh
```

Redémarrez le service sshd :

```
root@debian11:/tmp# systemctl restart sshd
```

```
root@debian11:/tmp# systemctl status sshd
● ssh.service - OpenBSD Secure Shell server
     Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: enabled)
     Active: active (running) since Tue 2022-05-03 11:01:24 CEST; 7s ago
       Docs: man:sshd(8)
             man:sshd_config(5)
    Process: 4885 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
   Main PID: 4888 (sshd)
      Tasks: 1 (limit: 4632)
     Memory: 1.1M
        CPU: 24ms
     CGroup: /system.slice/ssh.service
             └─4888 sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups

May 03 11:01:24 debian11.ittraining.loc sshd[4888]: Server listening on 0.0.0.0 port >
May 03 11:01:24 debian11.ittraining.loc systemd[1]: Stopping OpenBSD Secure Shell ser>
May 03 11:01:24 debian11.ittraining.loc sshd[4888]: Server listening on :: port 22.
May 03 11:01:24 debian11.ittraining.loc systemd[1]: ssh.service: Succeeded.
May 03 11:01:24 debian11.ittraining.loc systemd[1]: Stopped OpenBSD Secure Shell serv>
May 03 11:01:24 debian11.ittraining.loc systemd[1]: Starting OpenBSD Secure Shell ser>
May 03 11:01:24 debian11.ittraining.loc systemd[1]: Started OpenBSD Secure Shell serv>
lines 1-20/20 (END)
[q]
```

Mettez l'utilisateur **trainee** dans le groupe **adm** :

```
root@debian11:/tmp# groups trainee
trainee : trainee cdrom floppy audio dip src video plugdev netdev lpadmin scanner
root@debian11:/tmp# usermod -aG adm trainee
root@debian11:/tmp# groups trainee
trainee : trainee adm cdrom floppy audio dip src video plugdev netdev lpadmin scanner
```

Pour générer les clefs du serveur, saisissez la commande suivante en tant que **root**. Notez que la passphrase doit être **vide**.

```
root@debian11:/tmp# ssh-keygen -t dsa
Generating public/private dsa key pair.
Enter file in which to save the key (/root/.ssh/id_dsa): /etc/ssh/ssh_host_dsa_key
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /etc/ssh/ssh_host_dsa_key
Your public key has been saved in /etc/ssh/ssh_host_dsa_key.pub
The key fingerprint is:
SHA256:emwF3Bq/2H+JGk5dNXgKBLTtbuC04byZmFJDxkUxso4 root@debian11.ittraining.loc
The key's randomart image is:
+---[DSA 1024]----+
|       .o*o.     |
|       .oo=   .  |
|      ..=..o . o.|
|      o+ =. . o..|
|     Eo.S+o. ..  |
|       ==+=o .   |
|      o *==o.. . |
|     . ooo=o. o  |
|      .o +o...   |
+----[SHA256]-----+
root@debian11:/tmp# ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa): /etc/ssh/ssh_host_rsa_key
/etc/ssh/ssh_host_rsa_key already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /etc/ssh/ssh_host_rsa_key
Your public key has been saved in /etc/ssh/ssh_host_rsa_key.pub
The key fingerprint is:
SHA256:xrkpZ6lfF1hQIZEYJuH1L+Z2QGhQCf8Xwt9HPwTuT7Y root@debian11.ittraining.loc
The key's randomart image is:
+---[RSA 3072]----+
```

```
|      =+=+++oo   |
|     . *o+.o. .  |
|      . + = o. ..|
|       o + *.o...|
|        S * =..+o|
|       . * + .+.o|
|      . * + o  E |
|       = o o     |
|      ...        |
+----[SHA256]-----+
root@debian11:/tmp# ssh-keygen -t ecdsa
Generating public/private ecdsa key pair.
Enter file in which to save the key (/root/.ssh/id_ecdsa): /etc/ssh/ssh_host_ecdsa_key
/etc/ssh/ssh_host_ecdsa_key already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /etc/ssh/ssh_host_ecdsa_key
Your public key has been saved in /etc/ssh/ssh_host_ecdsa_key.pub
The key fingerprint is:
SHA256:38O9lqa1AHvviceNEbQ1AOUMspkYBpIpLlu/U05ymqo root@debian11.ittraining.loc
The key's randomart image is:
+---[ECDSA 256]---+
|   .o..o . ooo.  |
| . o. . o = + ...|
|. .    . +   + o.|
|...           o  |
|.o .    S .    . |
|.   o +  . = ..  |
|     X    o *.o= |
|    = .    . *B+.|
|E... .      o**  |
+----[SHA256]-----+
root@debian11:/tmp# ssh-keygen -t ed25519
```

```
Generating public/private ed25519 key pair.
Enter file in which to save the key (/root/.ssh/id_ed25519): /etc/ssh/ssh_host_ed25519_key
/etc/ssh/ssh_host_ed25519_key already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /etc/ssh/ssh_host_ed25519_key
Your public key has been saved in /etc/ssh/ssh_host_ed25519_key.pub
The key fingerprint is:
SHA256:f8arQ5MBRGNJoj4eARYapvxf/MLxFFMZcKf1eLkgeow root@debian11.ittraining.loc
The key's randomart image is:
+--[ED25519 256]--+
|..+.  .+*o.+oo   |
|++ . . oo.o.+ o .|
|o.  o    + o o + |
|  .. ..   B . o .|
|   .+  +SE =   . |
|   ..oo =.=.     |
|    .. o +..+    |
|        . .o .   |
|          .o.    |
+----[SHA256]-----+
```

Les clefs publiques générées possèdent l'extension **.pub**. Les clefs privées n'ont pas d'extension :

```
root@debian11:/tmp# ls /etc/ssh
moduli        sshd_config.d         ssh_host_ecdsa_key        ssh_host_rsa_key
ssh_config    sshd_config.old       ssh_host_ecdsa_key.pub    ssh_host_rsa_key.pub
ssh_config.d  ssh_host_dsa_key      ssh_host_ed25519_key
sshd_config   ssh_host_dsa_key.pub  ssh_host_ed25519_key.pub
```

Re-démarrez ensuite le service sshd :

```
root@debian11:/tmp# systemctl restart sshd.service
```

```
root@debian11:/tmp# systemctl status sshd.service
● ssh.service - OpenBSD Secure Shell server
     Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: enabled)
     Active: active (running) since Tue 2022-05-03 11:30:09 CEST; 8s ago
       Docs: man:sshd(8)
             man:sshd_config(5)
    Process: 4942 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
   Main PID: 4943 (sshd)
      Tasks: 1 (limit: 4632)
     Memory: 1.1M
        CPU: 24ms
     CGroup: /system.slice/ssh.service
             └─4943 sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups

May 03 11:30:09 debian11.ittraining.loc systemd[1]: Starting OpenBSD Secure Shell ser>
May 03 11:30:09 debian11.ittraining.loc sshd[4943]: Server listening on 0.0.0.0 port >
May 03 11:30:09 debian11.ittraining.loc sshd[4943]: Server listening on :: port 22.
May 03 11:30:09 debian11.ittraining.loc systemd[1]: Started OpenBSD Secure Shell serv>
lines 1-17/17 (END)
[q]
```

Configuration du Client

Saisissez maintenant les commandes suivantes en tant que **trainee** :

Important - Lors de la génération des clefs, la passphrase doit être **vide**.

```
root@debian11:/tmp# exit
logout
trainee@debian11:~$ ssh-keygen -t dsa
```

```
Generating public/private dsa key pair.
Enter file in which to save the key (/home/trainee/.ssh/id_dsa):
Created directory '/home/trainee/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/trainee/.ssh/id_dsa
Your public key has been saved in /home/trainee/.ssh/id_dsa.pub
The key fingerprint is:
SHA256:WijVQNwc9klBZxW4R8ZY5LwZK08/FaZCkWP0yKNj+20 trainee@debian11.ittraining.loc
The key's randomart image is:
+---[DSA 1024]----+
|     ooooo=++B=. |
|      .+oo.B*o+  |
|      . . +=.=+o |
|     . .  o o +=.|
|    . . S+ ..o= .|
|     . o. o .+ ..|
|      .  .    ...|
|          . .E  .|
|           ...   |
+----[SHA256]-----+
trainee@debian11:~$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/trainee/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/trainee/.ssh/id_rsa
Your public key has been saved in /home/trainee/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:r8id4Px97b9t7GMybe7OTaQU5lgaRjsTK/EAyMFdgjo trainee@debian11.ittraining.loc
The key's randomart image is:
+---[RSA 3072]----+
|     o.=oo+ o    |
|      = .. = +   |
```

```
|     .    . O +  |
|    E      o X . |
|     .  S   o o .|
|         .   . o |
|      .   .  .o..|
|     + + +  .oo*=|
|      =.= .. .XXB|
+----[SHA256]-----+
trainee@debian11:~$ ssh-keygen -t ecdsa
Generating public/private ecdsa key pair.
Enter file in which to save the key (/home/trainee/.ssh/id_ecdsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/trainee/.ssh/id_ecdsa
Your public key has been saved in /home/trainee/.ssh/id_ecdsa.pub
The key fingerprint is:
SHA256:IL44+ZzExDxBFCt9nZtMgCQB/iuq9UtJC6uq/RhrPvg trainee@debian11.ittraining.loc
The key's randomart image is:
+---[ECDSA 256]---+
|..o+=o.          |
|.  +.. o .       |
| .. = o +        |
|  .= + + o       |
|  ..B   S        |
|   O.=           |
| oB.B            |
|o++@ .           |
|X+EoB.           |
+----[SHA256]-----+
trainee@debian11:~$ ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/trainee/.ssh/id_ed25519):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
```

```
Your identification has been saved in /home/trainee/.ssh/id_ed25519
Your public key has been saved in /home/trainee/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:yQ9mtIx1nK7D1vSZjkkbofXHZsTDp5P0rywiJIMX35M trainee@debian11.ittraining.loc
The key's randomart image is:
+--[ED25519 256]--+
|                 |
|         . .     |
|        o +      |
|      .* =   o   |
|     ..oS.=.  * .|
|    . ++oOE+ * * |
|     . +* =.= O .|
|       ..o.*.= ..|
|         .+...o..|
+----[SHA256]-----+
```

Les clés générées seront placées dans le répertoire **~/.ssh/** :

```
trainee@debian11:~$ ls .ssh
id_dsa      id_ecdsa      id_ed25519      id_rsa
id_dsa.pub  id_ecdsa.pub  id_ed25519.pub  id_rsa.pub
```

Tunnels SSH

Le protocole SSH peut être utilisé pour sécuriser les protocoles tels telnet, pop3 etc.. En effet, on peut créer un *tunnel* SSH dans lequel passe les communications du protocole non-sécurisé.

La commande pour créer un tunnel ssh prend la forme suivante :

```
ssh -N -f compte@hôte -Lport-local:localhost:port_distant
```

Dans votre cas, vous allez créer un tunnel dans votre propre VM entre le port 15023 et le port 23 :

```
trainee@debian11:~$ su -
Password: fenestros
root@debian11:~# ssh -N -f trainee@localhost -L15023:localhost:23
The authenticity of host 'localhost (::1)' can't be established.
ECDSA key fingerprint is SHA256:38O9lqa1AHvviceNEbQ1AOUMspkYBpIpLlu/U05ymqo.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'localhost' (ECDSA) to the list of known hosts.
Debian GNU/Linux 11
trainee@localhost's password: trainee
```

Installez maintenant le serveur telnet :

```
root@debian11:~# apt -y install telnetd
```

Vérifiez que le service **inetd** est démarré :

```
root@debian11:~# systemctl status inetd
● inetd.service - Internet superserver
     Loaded: loaded (/lib/systemd/system/inetd.service; enabled; vendor preset: enabl>
     Active: active (running) since Tue 2022-05-03 11:55:27 CEST; 44s ago
       Docs: man:inetd(8)
   Main PID: 5110 (inetd)
      Tasks: 1 (limit: 4632)
     Memory: 576.0K
        CPU: 7ms
     CGroup: /system.slice/inetd.service
             └─5110 /usr/sbin/inetd

May 03 11:55:27 debian11.ittraining.loc systemd[1]: Starting Internet superserver...
May 03 11:55:27 debian11.ittraining.loc systemd[1]: Started Internet superserver.
```

Connectez-vous ensuite via telnet sur le port 15023, vous constaterez que votre connexion n'aboutit pas :

```
root@debian11:~# telnet localhost 15023
Trying ::1...
Connected to localhost.
Escape character is '^]'.
Debian GNU/Linux 11
debian11.ittraining.loc login: trainee
Password: trainee
Linux debian11.ittraining.loc 5.17.0-1-amd64 #1 SMP PREEMPT Debian 5.17.3-1 (2022-04-18) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Tue May  3 08:57:59 CEST 2022 from 10.0.2.1 on pts/0

trainee@debian11:~$ whoami
trainee

trainee@debian11:~$ pwd
/home/trainee

trainee@debian11:~$ exit
logout
Connection closed by foreign host.
```

Important - Notez bien que votre communication telnet passe par le tunnel SSH.

3.5 - SCP

Présentation

La commande **scp** est le successeur et la remplaçante de la commande **rcp** de la famille des commandes **remote**. Il permet de faire des transferts sécurisés à partir d'une machine distante :

```
$ scp compte@numero_ip(nom_de_machine):/chemin_distant/fichier_distant /chemin_local/fichier_local
```

ou vers une machine distante :

```
$ scp /chemin_local/fichier_local compte@numero_ip(nom_de_machine):/chemin_distant/fichier_distant
```

Utilisation

Nous allons maintenant utiliser **scp** pour chercher un fichier sur le «serveur» :

Créez le fichier **/home/trainee/scp_test** :

```
[trainee@centos8 ~]$ touch scp-test
[trainee@centos8 ~]$ exit
logout
Connection closed by foreign host.
[root@centos8 ~]#
```

Récupérez le fichier **scp_test** en utilisant scp :

```
[root@centos8 ~]# scp trainee@127.0.0.1:/home/trainee/scp-test .
The authenticity of host '127.0.0.1 (127.0.0.1)' can't be established.
ECDSA key fingerprint is SHA256:Q7T/CP0SLiMbMAIgVzTuEHegYS/spPE5zzQchCHD5Vw.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
```

```
Warning: Permanently added '127.0.0.1' (ECDSA) to the list of known hosts.
\S
Kernel \r on an \m
trainee@127.0.0.1's password: trainee
scp-test
100%    0     0.0KB/s   00:00

[root@centos8 ~]# ls -l
total 32
-rw-------. 1 root root 1358 Jun 16 06:40 anaconda-ks.cfg
drwxr-xr-x. 3 root root   21 Jun 16 06:39 home
-rw-r--r--. 1 root root 1749 Aug 24 11:20 I2TCH.asc
-rw-r--r--. 1 root root 1853 Jun 16 06:54 initial-setup-ks.cfg
-rw-r--r--. 1 root root   31 Aug 24 11:22 message.txt
-rw-r--r--. 1 root root  561 Aug 24 11:32 message.txt.asc
-rw-r--r--. 1 root root  367 Aug 24 11:30 message.txt.gpg
-rw-r--r--. 1 root root  329 Aug 24 11:23 message.txt.sig
-rw-r--r--. 1 root root    0 Aug 30 03:55 scp-test
-rw-r--r--. 1 root root   46 Aug 29 06:22 wget_file.txt
```

3.6 - Mise en Place des Clefs Asymétriques

Il convient maintenant de se connecter sur le «serveur» en utilisant ssh et vérifiez la présence du répertoire ~/.ssh :

```
[root@centos8 ~]# ssh -l trainee 127.0.0.1
\S
Kernel \r on an \m
trainee@127.0.0.1's password: trainee
Activate the web console with: systemctl enable --now cockpit.socket

[trainee@centos8 ~]$ ls -la | grep .ssh
drwx------.  2 trainee trainee      4096 Aug 30 02:26 .ssh
```

Important - Si le dossier distant .ssh n'existe pas dans le répertoire personnel de l'utilisateur connecté, il faut le créer avec des permissions de 700. Dans votre cas, puisque votre machine joue le rôle de serveur **et** du client, le dossier /home/trainee/.ssh existe **déjà**.

Ensuite, il convient de transférer le fichier local **.ssh/id_ecdsa.pub** du «client» vers le «serveur» en le renommant en **authorized_keys** :

```
[trainee@centos8 ~]$ exit
logout
Connection to 127.0.0.1 closed.

[root@centos8 ~]# exit
logout

[trainee@centos8 ~]$ scp .ssh/id_ecdsa.pub trainee@127.0.0.1:/home/trainee/.ssh/authorized_keys
The authenticity of host '127.0.0.1 (127.0.0.1)' can't be established.
ECDSA key fingerprint is SHA256:Q7T/CP0SLiMbMAIgVzTuEHegYS/spPE5zzQchCHD5Vw.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '127.0.0.1' (ECDSA) to the list of known hosts.
\S
Kernel \r on an \m
trainee@127.0.0.1's password: trainee
id_ecdsa.pub
100%  192   497.6KB/s   00:00
```

Connectez-vous via telnet :

```
[trainee@centos8 ~]$ ssh -l trainee localhost
The authenticity of host 'localhost (::1)' can't be established.
ECDSA key fingerprint is SHA256:Q7T/CP0SLiMbMAIgVzTuEHegYS/spPE5zzQchCHD5Vw.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
```

```
Warning: Permanently added 'localhost' (ECDSA) to the list of known hosts.
\S
Kernel \r on an \m
Activate the web console with: systemctl enable --now cockpit.socket

Last login: Mon Aug 30 03:57:14 2021 from 127.0.0.1
[trainee@centos8 ~]$
```

Important - Lors de la connexion au serveur, l'authentification utilise le couple de clefs asymétrique au format ecdsa et aucun mot de passe n'est requis.

Insérez maintenant les clefs publiques restantes dans le fichier .ssh/authorized_keys :

```
[trainee@centos8 ~]$ cd .ssh
[trainee@centos8 .ssh]$ ls
authorized_keys  id_dsa  id_dsa.pub  id_ecdsa  id_ecdsa.pub  id_ed25519  id_ed25519.pub  id_rsa  id_rsa.pub
known_hosts
[trainee@centos8 .ssh]$ cat authorized_keys
ecdsa-sha2-nistp256
AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBHDrzSXP+Ecxf/sQ18VwCRNm7rrSrrsaJmuIw/RgTH5puKF5E+Yy15cvAAKBX
pJPxUmrOaOyhab84PevV7XSHcI= trainee@centos8.ittraining.loc

[trainee@centos8 .ssh]$ cat id_rsa.pub >> authorized_keys
[trainee@centos8 .ssh]$ cat id_dsa.pub >> authorized_keys
[trainee@centos8 .ssh]$ cat id_ed25519.pub >> authorized_keys

[trainee@centos8 .ssh]$ cat authorized_keys
ecdsa-sha2-nistp256
AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBHDrzSXP+Ecxf/sQ18VwCRNm7rrSrrsaJmuIw/RgTH5puKF5E+Yy15cvAAKBX
pJPxUmrOaOyhab84PevV7XSHcI= trainee@centos8.ittraining.loc
ssh-rsa
```

```
AAAAB3NzaC1yc2EAAAADAQABAAABgQD3ZSMn/GIAHtaDFc6ZNnKJam9hzq8TxqMN5IopUr8QhwODyPadbB+FgH4r50qTux4ubwr1BlymgIdqRVWy3
2mE15M8tdtKc3j8DNMpUwPGEh+s/PT7GW+3E3shoyPvpLc1kKaKXKGl/JwfCK/8IYsubk2BmiiJYkzLECotPlaaxm4w1K0AtlnZQuLHt1HK3/rHCh
xo2o2w1t59/QwNcMLiKve1Z+zQ1POKo8VJ/DDrf90y2QWC28ejUs/ZjP6f6C4bn5Jjol4TbHls3ArMsbU6C1Ev5jqbzZ0kmognQ2CnRjeNM51YdFo
6nRsLoPQKpeLRMBpT/87HK1bPUOBVCyhXxSkqkVhMlgg8tgcD/MRlBaUVFoZ1wQ0L26Fe+q7c2Oykj54pdXCAK2ZTpCXGfhd/FxrfqGw7cSeKlGX4
QUzHMjMk2oHFC9h30BUk1gGN21KJPTh7/S1OZVrnCc3GUi5fXPvEpral0IU0sws+j03dj0sWm5ICQFKRkmZN11HCyT0=
trainee@centos8.ittraining.loc
ssh-dss
AAAAB3NzaC1kc3MAAACBALIdwEEqHrMWSUdzARm9ldsZK9ebbtZShtmwgdjphOk77fxymK0y6wV7QEmLL25LOcLb12uZ1F0LtRt/t2oqgrwqk3vUS
pCPLr09AXpcD/nxL9kc+rUxHyl6u1mHtyfCVLCPSvavCMR8TaA8egVMk3EwGRfHTiuDOKi7Iwus7gXPAAAAFQDHEQPGVRI7gVYKzCT6nrjDsQQ6jw
AAAIEAhhhH7fEjdldASXY0qTWkCvcs3cfK9/Ff315zByn47O02y9Vdo3QG5nOr1Oo8fc2xEkIBNmFr8Rr2g60cpvEev5hy4XZ1ghxnQ53iwKuiS72
ZATwhD6bZBrsiH0k1Et25gRcj5KCvDe/jHhbxCxsCuHUH2qvWsQNVwztE7hD0sxkAAACAQ8Dkpy8zXj7jW8o1txxf2W6J4r2+1lPldymA45ywZokN
4SCwvXlpPAuyBt0/HiU0R2PI9aqOAMosCLcy9WmnSwLQ2Z7QcD2i3XlAih2+1q9NJP22sPT3jSK9UZcdRjoZ/eNiz84sXZucNape32tFxjvcV4txo
bH/vD53q8g63fA= trainee@centos8.ittraining.loc
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIOfFQULLU8IZyKiSU63D2Zz6yGLqyHcBHnCRdSR9JSmc trainee@centos8.ittraining.loc
```

3.7 - Services réseaux

Quand un client émet une demande de connexion vers une application réseau sur un serveur, il utilise un socket attaché à un port local **supérieur à 1023**, alloué d'une manière dynamique. La requête contient le port de destination sur le serveur. Certaines applications serveurs se gèrent toutes seules, ce qui est la cas par exemple d'**apache2**. Par contre d'autres sont gérées par le service **xinetd**.

inetd

Le programme inetd est configuré via le fichier **/etc/inetd.conf** :

```
root@debian11:~# cat /etc/inetd.conf
# /etc/inetd.conf:  see inetd(8) for further informations.
#
# Internet superserver configuration database
#
#
```

```
# Lines starting with "#:LABEL:" or "#<off>#" should not
# be changed unless you know what you are doing!
#
# If you want to disable an entry so it isn't touched during
# package updates just comment it out with a single '#' character.
#
# Packages should modify this file by using update-inetd(8)
#
# <service_name> <sock_type> <proto> <flags> <user> <server_path> <args>
#
#:INTERNAL: Internal services
#discard                stream  tcp     nowait  root    internal
#discard                dgram   udp     wait    root    internal
#daytime                stream  tcp     nowait  root    internal
#time           stream  tcp     nowait  root    internal

#:STANDARD: These are standard services.
telnet          stream  tcp     nowait  telnetd /usr/sbin/tcpd  /usr/sbin/in.telnetd

#:BSD: Shell, login, exec and talk are BSD protocols.

#:MAIL: Mail, news and uucp services.

#:INFO: Info services

#:BOOT: TFTP service is provided primarily for booting.  Most sites
#       run this only on machines acting as "boot servers."

#:RPC: RPC based services

#:HAM-RADIO: amateur-radio services

#:OTHER: Other services
```

Les lignes de configuration des serveurs ressemblent à :

```
telnet          stream  tcp     nowait  telnetd /usr/sbin/tcpd  /usr/sbin/in.telnetd
```

Le premier champs de la ligne identifie le nom du port qui identifie l'application. Inetd lit ensuite le fichier **/etc/services** pour déduire le numéro de port concerné et se met à l'écoute de celui-ci.

Le deuxième et le troisième champs définissent le type de protocole, à savoir:

- stream tcp pour le tcp
- dgram udp pour l'udp

Le quatrième champs prend un de deux valeurs:

- nowait
  - indique qu'il y aura un serveur par client
- wait
  - indique qu'il y aura un seul serveur pour l'ensemble des clients

Le cinquième champs indique l'identité sous laquelle sera exécuté le serveur.

Le sixième champs indique l'exécutable. Dans ce cas c'est **/usr/sbin/tcpd**.

Le septième champs indique les arguments de l'application serveur dont l'argument **0** est le nom de l'application.

TCP Wrapper

Lors de l'utilisation d'inetd, **TCP Wrapper** est utilisé pour contrôler l'accès à des services réseaux grâce à des **ACL** :

```
telnet          stream  tcp     nowait  telnetd /usr/sbin/tcpd  /usr/sbin/in.telnetd
```

Quand une requête arrive pour le serveur telnet, inetd active le wrapper **tcpd** au lieu d'activer **/usr/sbin/in.telnetd** directement.

tcpd met à jour un journal et vérifie si le client a le droit d'utiliser le service concerné. Les ACL se trouvent dans deux fichiers:

```
root@debian11:~# cat /etc/hosts.allow
# /etc/hosts.allow: list of hosts that are allowed to access the system.
#                   See the manual pages hosts_access(5) and hosts_options(5).
#
# Example:    ALL: LOCAL @some_netgroup
#             ALL: .foobar.edu EXCEPT terminalserver.foobar.edu
#
# If you're going to protect the portmapper use the name "rpcbind" for the
# daemon name. See rpcbind(8) and rpc.mountd(8) for further information.
#

root@debian11:~# cat /etc/hosts.deny
# /etc/hosts.deny: list of hosts that are _not_ allowed to access the system.
#                  See the manual pages hosts_access(5) and hosts_options(5).
#
# Example:    ALL: some.host.name, .some.domain
#             ALL EXCEPT in.fingerd: other.host.name, .other.domain
#
# If you're going to protect the portmapper use the name "rpcbind" for the
# daemon name. See rpcbind(8) and rpc.mountd(8) for further information.
#
# The PARANOID wildcard matches any host whose name does not match its
# address.
#
# You may wish to enable this to ensure any programs that don't
# validate looked up hostnames still leave understandable logs. In past
# versions of Debian this has been the default.
# ALL: PARANOID
```

Il faut noter que si ces fichiers n'existent pas ou sont vides, il n'y a pas de contrôle d'accès.

Le format d'une ligne dans un de ces deux fichiers est:

```
démon: liste_de_clients
```

Par exemple dans le cas de notre serveur telnetd, une ligne dans le fichier **/etc/hosts.allow** similaire à:

```
in.telnetd: 192.168.1.10, .fenestros.com
```

implique que la machine dont le numéro IP est le 192.168.1.10 ainsi que les machines du domaine **fenestros.com** sont autorisées à utiliser le service.

Le mot clef **ALL** peut être utilisé pour indiquer tout. Par exemple, **ALL:ALL** dans le fichier **/etc/host.deny** bloque effectivement toute tentative de connexion à un service inetd sauf pour les ACL inclus dans le fichier /etc/host.allow.

Configuration du Réseau sous RHEL/CentOS 6

Configuration de TCP/IP

La configuration TCP/IP se trouve dans le répertoire **/etc/sysconfig**. Les fichiers importants sont :

DHCP

/etc/sysconfig/network

```
[root@centos6 ~]# cat /etc/sysconfig/network
NETWORKING=yes
HOSTNAME=centos6
```

Dans ce fichier vous pouvez constater les directives suivantes :

Directive	Description
NETWORKING	Indique que la prise en charge du réseau est activée
HOSTNAME	Indique le nom d'hôte de la machine

Ce fichier peut également contenir les directives suivantes :

Directive	Description
GATEWAY	Indique l'adresse IPv4 de la passerelle
GATEWAYDEV	Indique l'interface réseau utilisée pour accéder à la passerelle
NISDOMAIN	Indique le domaine NIS s'il en existe un
NETWORKING_IPV6	Active ou désactive le support IPv6

/etc/sysconfig/network-scripts/ifcfg-ethX (où X=0,1 ...)

ifcfg-eth0

```
[root@centos6 ~]# cat /etc/sysconfig/network-scripts/ifcfg-eth0
DEVICE="eth0"
NM_CONTROLLED="yes"
ONBOOT=yes
TYPE=Ethernet
BOOTPROTO=dhcp
DEFROUTE=yes
IPV4_FAILURE_FATAL=yes
IPV6INIT=no
NAME="System eth0"
UUID=5fb06bd0-0bb0-7ffb-45f1-d6edd65f3e03
HWADDR=08:00:27:48:7D:7F
PEERDNS=yes
PEERROUTES=yes
```

Dans ce fichier vous pouvez constater les directives suivantes :

Directive	Description
DEVICE	Indique le nom de l'interface
NM_CONTROLLED	Indique que le service NetworkManager est utilisé pour gérer les interfaces réseau
ONBOOT	Indique que l'interface est activée au démarrage de la machine

Directive	Description
TYPE	Indique que le type de réseau est ethernet. Les valeurs permises sont ethernet ou wireless
BOOTPROTO	Indique comment monter l'interface. Les valeurs permises sont dhcp, static ou bootp
DEFROUTE	Définit l'interface en tant que passerelle par défaut
IPV4_FAILURE_FATAL	Stipule que si IPv4 et IPv6 sont activés et la connexion IPv4 est perdue, la connexion IPv6 est considérée d'être perdue
IPV6INIT	Indique que le support IPv6 ne sera pas initialisé
NAME	Indique un nom descriptif de l'interface
UUID	Indique la valeur de l'UUID de l'interface
HWADDR	Indique l'adresse MAC de l'interface
PEERDNS	Indique que le fichier /etc/resolv.conf doit être modifié automatiquement pour contenir les adresses IP des DNS fournies par le serveur DHCP

Recherchez la définition de la directive **PEERROUTES=yes**.

IP Fixe

/etc/sysconfig/network

```
NETWORKING=yes
NETWORKING_IPV6=no
HOSTNAME=centos6.fenestros.loc
```

/etc/sysconfig/network-scripts/ifcfg-ethX (où X=0,1 ...)

ifcfg-eth0

```
DEVICE="eth0"
```

```
NM_CONTROLLED="no"
ONBOOT=yes
TYPE=Ethernet
BOOTPROTO=static
IPV6INIT=no
HWADDR="08:00:27:9B:55:B1"
NETMASK=255.255.255.0
IPADDR=10.0.2.15
GATEWAY=10.0.2.2
DNS1=8.8.8.8
DNS2=8.8.4.4
DOMAIN=fenestros.loc
USERCTL=yes
```

Dans ce fichier vous pouvez constater les nouvelles directives suivantes :

Directive	**Description**
NETMASK	Indique le masque de sous-réseau IPv4 associé à l'interface
IPADDR	Indique l'adresse IPv4 de l'interface
GATEWAY	Indique l'adresse IPv4 de la passerelle par défaut
DNS1	Indique le DNS primaire
DNS2	Indique le DNS secondaire
DOMAIN	Indique le nom du domaine local
USERCTL	Indique que les utilisateurs normaux peuvent activer/désactiver l'interface

Notez que VirtualBox fournit une passerelle par défaut (10.0.2.2).

Après avoir modifier les deux fichiers **/etc/sysconfig/network** et **/etc/sysconfig/network-scripts/ifcfg-eth0** vous devez désactiver le service **NetworkManager** utilisé pour la connexion DHCP et activer le service **network** :

```
[root@centos6 ~]# service NetworkManager stop
```

```
Arrêt du démon NetworkManager :                            [  OK  ]
[root@centos6 ~]# chkconfig --del NetworkManager
[root@centos6 ~]# service network start
Activation de l'interface loopback :                       [  OK  ]
Activation de l'interface eth0 :                           [  OK  ]
[root@centos6 ~]#
```

La Commande hostname

Lors du passage à une configuration en IPv4 fixe vous avez modifié la directive **HOSTNAME** du fichier **/etc/sysconfig/network** de **centos** à **centos.fenestros.loc**. Afin d'informer le système immédiatement de la modification du FQDN (*Fully Qualified Domain Name*), utilisez la commande **hostname** :

```
[root@centos6 ~]# hostname
centos6
[root@centos6 ~]# hostname centos6.fenestros.loc
[root@centos6 ~]# hostname
centos6.fenestros.loc
```

Pour afficher le FQDN du système vous pouvez également utiliser la commande suivante :

```
[root@centos6 ~]# uname -n
centos6.fenestros.loc
```

Options de la commande hostname

Les options de cette commande sont :

```
[root@centos6 ~]# hostname --help
Syntaxe : hostname [-v] {hôte|-F fichier}      définit le nom d'hôte (depuis le fichier)
       domainname [-v] {domaine_nis|-F fichier}   définit le domaine NIS (depuis le fichier)
```

```
       hostname [-v] [-d|-f|-s|-a|-i|-y]  display formatted name
       hostname [-v]                          affiche le nom d'hôte

       hostname -V|--version|-h|--help       affiche des infos et termine

    dnsdomainname=hostname -d, {yp,nis,}domainname=hostname -y

    -s, --short           nom d'hôte court
    -a, --alias           noms d'alias
    -i, --ip-address      adresses de l'hôte
    -f, --fqdn, --long    nom d'hôte long (FQDN)
    -d, --domain          nom de domaine DNS
    -y, --yp, --nis       nom de domaine NIS/YP
    -F, --file            read hostname or NIS domainname from given file

   This command can read or set the hostname or the NIS domainname. You can
   also read the DNS domain or the FQDN (fully qualified domain name).
   Unless you are using bind or NIS for host lookups you can change the
   FQDN (Fully Qualified Domain Name) and the DNS domain name (which is
   part of the FQDN) in the /etc/hosts file.
```

La Commande ifconfig

Pour afficher la configuration IP de la machine il faut saisir la commande suivante :

```
[root@centos6 ~]# ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:48:7D:7F
          inet adr:10.0.2.15  Bcast:10.0.2.255  Masque:255.255.255.0
          adr inet6: fe80::a00:27ff:fe48:7d7f/64 Scope:Lien
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:16765 errors:0 dropped:0 overruns:0 frame:0
          TX packets:15256 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 lg file transmission:1000
```

```
          RX bytes:12435171 (11.8 MiB)  TX bytes:4767389 (4.5 MiB)

lo        Link encap:Boucle locale
          inet adr:127.0.0.1  Masque:255.0.0.0
          adr inet6: ::1/128 Scope:Hôte
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:8 errors:0 dropped:0 overruns:0 frame:0
          TX packets:8 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 lg file transmission:0
          RX bytes:480 (480.0 b)  TX bytes:480 (480.0 b)
```

La commande ifconfig est également utilisée pour configurer une interface.

Créez maintenant une interface fictive ainsi :

```
[root@centos6 ~]# ifconfig eth0:1 192.168.1.2 netmask 255.255.255.0 broadcast 192.168.1.255
```

Constatez maintenant le résultat :

```
[root@centos6 ~]# ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:48:7D:7F
          inet adr:10.0.2.15  Bcast:10.0.2.255  Masque:255.255.255.0
          adr inet6: fe80::a00:27ff:fe48:7d7f/64 Scope:Lien
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:16904 errors:0 dropped:0 overruns:0 frame:0
          TX packets:15337 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 lg file transmission:1000
          RX bytes:12445250 (11.8 MiB)  TX bytes:4816855 (4.5 MiB)

eth0:1    Link encap:Ethernet  HWaddr 08:00:27:48:7D:7F
          inet adr:192.168.1.2  Bcast:192.168.1.255  Masque:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1

lo        Link encap:Boucle locale
```

```
          inet adr:127.0.0.1  Masque:255.0.0.0
          adr inet6: ::1/128 Scope:Hôte
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:8 errors:0 dropped:0 overruns:0 frame:0
          TX packets:8 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 lg file transmission:0
          RX bytes:480 (480.0 b)  TX bytes:480 (480.0 b)
```

Options de la commande ifconfig

Les options de cette commande sont :

```
[root@centos6 ~]# ifconfig --help
Usage:
  ifconfig [-a] [-v] [-s] <interface> [[<AF>] <address>]
  [add <adresse>[/<lg_prefixe>]]
  [del <adresse>[/<lg_prefixe>]]
  [[-]broadcast [<adresse>]]  [[-]pointopoint [<adresse>]]
  [netmask <address>]  [dstaddr <address>]  [tunnel <address>]
  [outfill <NN>] [keepalive <NN>]
  [hw <HW> <adresse>]  [metric <NN>]  [mtu <NN>]
  [[-]trailers]  [[-]arp]  [[-]allmulti]
  [multicast]  [[-]promisc]
  [mem_start <NN>]  [io_addr <NN>]  [irq <NN>]  [media <type>]
  [txqueuelen <NN>]
  [[-]dynamic]
  [up|down] ...

  <HW>=Type de matériel.
  Liste des types de matériels possibles:
    loop (Boucle locale) slip (IP ligne série) cslip (IP ligne série - VJ )
    slip6 (IP ligne série - 6 bits) cslip6 (IP ligne série - 6 bits VJ) adaptive (IP ligne série adaptative)
    strip (Metricom Starmode IP) ash (Ash) ether (Ethernet)
```

```
    tr (16/4 Mbps Token Ring) tr (16/4 Mbps Token Ring (New)) ax25 (AMPR AX.25)
    netrom (AMPR NET/ROM) rose (AMPR ROSE) tunnel (IPIP Tunnel)
    ppp (Protocole Point-à-Point) hdlc ((Cisco)-HDLC) lapb (LAPB)
    arcnet (ARCnet) dlci (Frame Relay DLCI) frad (Périphériue d'accès Frame Relay)
    sit (IPv6-dans-IPv4) fddi (Fiber Distributed Data Interface) hippi (HIPPI)
    irda (IrLAP) ec (Econet) x25 (generic X.25)
    infiniband (InfiniBand)
  <AF>=famille d'Adresses. Défaut: inet
  Liste des familles d'adresses possibles:
    unix (Domaine UNIX) inet (DARPA Internet) inet6 (IPv6)
    ax25 (AMPR AX.25) netrom (AMPR NET/ROM) rose (AMPR ROSE)
    ipx (Novell IPX) ddp (Appletalk DDP) ec (Econet)
    ash (Ash) x25 (CCITT X.25)
```

Activer/Désactiver une Interface Manuellement

Deux commandes existent pour activer et désactiver manuellement une interface réseau :

```
[root@centos6 ~]# ifdown eth0
[root@centos6 ~]# ifconfig
lo        Link encap:Boucle locale
          inet adr:127.0.0.1  Masque:255.0.0.0
          adr inet6: ::1/128 Scope:Hôte
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:384 errors:0 dropped:0 overruns:0 frame:0
          TX packets:384 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 lg file transmission:0
          RX bytes:32496 (31.7 KiB)  TX bytes:32496 (31.7 KiB)

[root@centos6 ~]# ifup eth0
État de connexion active : activation
État de chemin actif : /org/freedesktop/NetworkManager/ActiveConnection/0
état : activé
```

```
Connexion activée
[root@centos6 ~]# ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:9B:55:B1
          inet adr:10.0.2.15  Bcast:10.0.2.255  Masque:255.255.255.0
          adr inet6: fe80::a00:27ff:fe9b:55b1/64 Scope:Lien
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:27 errors:0 dropped:0 overruns:0 frame:0
          TX packets:42 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 lg file transmission:1000
          RX bytes:4271 (4.1 KiB)  TX bytes:6145 (6.0 KiB)

lo        Link encap:Boucle locale
          inet adr:127.0.0.1  Masque:255.0.0.0
          adr inet6: ::1/128 Scope:Hôte
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:392 errors:0 dropped:0 overruns:0 frame:0
          TX packets:392 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 lg file transmission:0
          RX bytes:33600 (32.8 KiB)  TX bytes:33600 (32.8 KiB)
```

/etc/networks

Ce fichier contient la correspondance entre des noms de réseaux et l'adresse IP du réseau :

```
[root@centos6 ~]# cat /etc/networks
default 0.0.0.0
loopback 127.0.0.0
link-local 169.254.0.0
```

Résolution d'adresses IP

La configuration DNS est stockée dans le fichier **/etc/resolv.conf**.

/etc/resolv.conf

La configuration DNS est stockée dans le fichier /etc/resolv.conf :

```
[root@centos6 ~]# cat /etc/resolv.conf
# Generated by NetworkManager


# No nameservers found; try putting DNS servers into your
# ifcfg files in /etc/sysconfig/network-scripts like so:
#
# DNS1=xxx.xxx.xxx.xxx
# DNS2=xxx.xxx.xxx.xxx
# DOMAIN=lab.foo.com bar.foo.com
nameserver 8.8.8.8
nameserver 8.8.4.4
search fenestros.loc
```

Notez que les DNS utilisés sont les serveurs DNS publics de Google.

/etc/nsswitch.conf

L'ordre de recherche des services de noms est stocké dans le fichier **/etc/nsswitch.conf**. Pour connaître l'ordre, saisissez la commande suivante :

```
[root@centos6 ~]# grep '^hosts:' /etc/nsswitch.conf
hosts:      files dns
```

/etc/hosts

Le mot **files** dans la sortie de la commande précédente fait référence au fichier **/etc/hosts** :

```
[root@centos6 ~]# cat /etc/hosts
10.0.2.15   centos6    # Added by NetworkManager
127.0.0.1   localhost.localdomain    localhost
::1 centos6    localhost6.localdomain6    localhost6
```

Pour tester le serveur DNS, deux commandes sont possibles :

```
[root@centos6 ~]# nslookup www.i2tch.com
Server:     8.8.8.8
Address:    8.8.8.8#53

Non-authoritative answer:
www.i2tch.com   canonical name = i2tch.com.
Name:   i2tch.com
Address: 90.119.37.144

[root@centos6 ~]# dig www.i2tch.com

; <<>> DiG 9.10.3-P4-Ubuntu <<>> www.i2tch.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 25061
;; flags: qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 65494
;; QUESTION SECTION:
;www.i2tch.com.         IN    A
```

```
;; ANSWER SECTION:
www.i2tch.com.      6563    IN    CNAME    i2tch.com.
i2tch.com.      50    IN    A    90.119.37.144

;; Query time: 1 msec
;; SERVER: 8.8.8.8#53(8.8.8.8)
;; WHEN: Mon Jan 22 18:00:27 CET 2018
;; MSG SIZE  rcvd: 72
```

Configuration du Réseau sous RHEL/CentOS 7

RHEL/CentOS 7 utilise exclusivement **Network Manager** pour gérer le réseau. Network Manager est composé de deux éléments :

- un service qui gère les connexions réseaux et rapporte leurs états,
- des front-ends qui passent par un API de configuration du service.

> **Important** : Notez qu'avec cette version de NetworkManager, IPv6 est activée par défaut.

Le service NetworkManager doit toujours être lancé :

```
[root@centos7 ~]# systemctl status NetworkManager.service
● NetworkManager.service - Network Manager
   Loaded: loaded (/usr/lib/systemd/system/NetworkManager.service; enabled; vendor preset: enabled)
   Active: active (running) since Sun 2016-08-07 09:18:20 CEST; 1 day 1h ago
 Main PID: 673 (NetworkManager)
   CGroup: /system.slice/NetworkManager.service
           ├─ 673 /usr/sbin/NetworkManager --no-daemon
           └─2673 /sbin/dhclient -d -q -sf /usr/libexec/nm-dhcp-helper -pf /var/run/dhclient-enp0s3.pid -lf
/var/lib/NetworkManager/dhclient-45b701c1-0a21-4d76-a795-...
```

```
Aug 08 11:03:55 centos7.fenestros.loc NetworkManager[673]: <info>    nameserver '8.8.8.8'
Aug 08 11:03:55 centos7.fenestros.loc NetworkManager[673]: <info>  (enp0s3): DHCPv4 state changed unknown ->
bound
Aug 08 11:03:55 centos7.fenestros.loc NetworkManager[673]: <info>  (enp0s3): device state change: ip-config ->
ip-check (reason 'none') [70 80 0]
Aug 08 11:03:55 centos7.fenestros.loc NetworkManager[673]: <info>  (enp0s3): device state change: ip-check ->
secondaries (reason 'none') [80 90 0]
Aug 08 11:03:55 centos7.fenestros.loc NetworkManager[673]: <info>  (enp0s3): device state change: secondaries ->
activated (reason 'none') [90 100 0]
Aug 08 11:03:55 centos7.fenestros.loc NetworkManager[673]: <info>  NetworkManager state is now CONNECTED_LOCAL
Aug 08 11:03:55 centos7.fenestros.loc NetworkManager[673]: <info>  NetworkManager state is now CONNECTED_GLOBAL
Aug 08 11:03:55 centos7.fenestros.loc NetworkManager[673]: <info>  Policy set 'Wired connection 1' (enp0s3) as
default for IPv4 routing and DNS.
Aug 08 11:03:55 centos7.fenestros.loc NetworkManager[673]: <info>  (enp0s3): Activation: successful, device
activated.
Aug 08 11:03:55 centos7.fenestros.loc dhclient[2673]: bound to 10.0.2.15 -- renewal in 39589 seconds.
```

La Commande nmcli

La commande **nmcli** (Network Manager Command Line Interface) est utilisée pour configurer NetworkManager.

Les options et les sous-commandes peuvent être consultées en utilisant les commandes suivantes :

```
[root@centos7 ~]# nmcli help
Usage: nmcli [OPTIONS] OBJECT { COMMAND | help }

OPTIONS
  -t[erse]                                    terse output
  -p[retty]                                   pretty output
  -m[ode] tabular|multiline                   output mode
  -f[ields] <field1,field2,...>|all|common    specify fields to output
  -e[scape] yes|no                            escape columns separators in values
  -n[ocheck]                                  don't check nmcli and NetworkManager versions
```

```
  -a[sk]                                    ask for missing parameters
  -w[ait] <seconds>                         set timeout waiting for finishing operations
  -v[ersion]                                show program version
  -h[elp]                                   print this help

OBJECT
  g[eneral]       NetworkManager's general status and operations
  n[etworking]    overall networking control
  r[adio]         NetworkManager radio switches
  c[onnection]    NetworkManager's connections
  d[evice]        devices managed by NetworkManager
  a[gent]         NetworkManager secret agent or polkit agent

[root@centos7 ~]# nmcli g help
Usage: nmcli general { COMMAND | help }

COMMAND := { status | hostname | permissions | logging }

  status

  hostname [<hostname>]

  permissions

  logging [level <log level>] [domains <log domains>]


[root@centos7 ~]# nmcli g status help
Usage: nmcli general status { help }

Show overall status of NetworkManager.
'status' is the default action, which means 'nmcli gen' calls 'nmcli gen status'
```

Connections et Profils

NetworkManager inclus la notion de **connections** ou **profils** permettant des configurations différentes en fonction de la localisation. Pour voir les connections actuelles, utilisez la commande **nmcli c** avec la sous-commande **show** :

```
[root@centos7 ~]# nmcli c show
NAME                UUID                                  TYPE            DEVICE
Wired connection 1  45b701c1-0a21-4d76-a795-2f2bcba86955  802-3-ethernet  enp0s3
```

Comme on peut constater ici, il n'existe pour le moment, qu'un seul profil.

Créez donc un profil IP fixe rattaché au périphérique **enp0s3** :

```
[root@centos7 ~]# nmcli connection add con-name ip_fixe ifname enp0s3 type ethernet ip4 10.0.2.16/24 gw4 10.0.2.2
Connection 'ip_fixe' (fb3a11d9-4e03-4032-b26e-09d1195d2bcd) successfully added.
```

Constatez sa présence :

```
[root@centos7 ~]# nmcli c show
NAME                UUID                                  TYPE            DEVICE
ip_fixe             fb3a11d9-4e03-4032-b26e-09d1195d2bcd  802-3-ethernet  --
Wired connection 1  45b701c1-0a21-4d76-a795-2f2bcba86955  802-3-ethernet  enp0s3
```

Notez que la sortie n'indique pas que le profil **ip_fixe** soit associé au periphérique **enp0s3** car le profil **ip_fixe** n'est pas activé :

```
[root@centos7 ~]# nmcli d show
GENERAL.DEVICE:                         enp0s3
GENERAL.TYPE:                           ethernet
GENERAL.HWADDR:                         08:00:27:03:97:DD
GENERAL.MTU:                            1500
GENERAL.STATE:                          100 (connected)
GENERAL.CONNECTION:                     Wired connection 1
GENERAL.CON-PATH:                       /org/freedesktop/NetworkManager/ActiveConnection/2
```

```
WIRED-PROPERTIES.CARRIER:               on
IP4.ADDRESS[1]:                         10.0.2.15/24
IP4.GATEWAY:                            10.0.2.2
IP4.DNS[1]:                             8.8.8.8
IP6.ADDRESS[1]:                         fe80::a00:27ff:fe03:97dd/64
IP6.GATEWAY:

GENERAL.DEVICE:                         lo
GENERAL.TYPE:                           loopback
GENERAL.HWADDR:                         00:00:00:00:00:00
GENERAL.MTU:                            65536
GENERAL.STATE:                          10 (unmanaged)
GENERAL.CONNECTION:                     --
GENERAL.CON-PATH:                       --
IP4.ADDRESS[1]:                         127.0.0.1/8
IP4.GATEWAY:
IP6.ADDRESS[1]:                         ::1/128
IP6.GATEWAY:
```

Pour activer le profil ip_fixe, utilisez la commande suivante :

```
[root@centos7 ~]# nmcli connection up ip_fixe
```

Le profil ip_fixe est maintenant activé tandis que le profil enp0s3 a été désactivé :

```
[root@centos7 ~]# nmcli c show
NAME                UUID                                  TYPE            DEVICE
ip_fixe             fb3a11d9-4e03-4032-b26e-09d1195d2bcd  802-3-ethernet  enp0s3
Wired connection 1  45b701c1-0a21-4d76-a795-2f2bcba86955  802-3-ethernet  --
[root@centos7 ~]# nmcli d show
GENERAL.DEVICE:                         enp0s3
GENERAL.TYPE:                           ethernet
GENERAL.HWADDR:                         08:00:27:03:97:DD
GENERAL.MTU:                            1500
```

```
GENERAL.STATE:                          100 (connected)
GENERAL.CONNECTION:                     ip_fixe
GENERAL.CON-PATH:                       /org/freedesktop/NetworkManager/ActiveConnection/3
WIRED-PROPERTIES.CARRIER:               on
IP4.ADDRESS[1]:                         10.0.2.16/24
IP4.GATEWAY:                            10.0.2.2
IP6.ADDRESS[1]:                         fe80::a00:27ff:fe03:97dd/64
IP6.GATEWAY:

GENERAL.DEVICE:                         lo
GENERAL.TYPE:                           loopback
GENERAL.HWADDR:                         00:00:00:00:00:00
GENERAL.MTU:                            65536
GENERAL.STATE:                          10 (unmanaged)
GENERAL.CONNECTION:                     --
GENERAL.CON-PATH:                       --
IP4.ADDRESS[1]:                         127.0.0.1/8
IP4.GATEWAY:
IP6.ADDRESS[1]:                         ::1/128
IP6.GATEWAY:
```

Pour consulter les paramètres d'un profil, utilisez la commande suivante :

```
[root@centos7 ~]# nmcli -p connection show "Wired connection 1"
===============================================================================
                Connection profile details (Wired connection 1)
===============================================================================
connection.id:                          Wired connection 1
connection.uuid:                        45b701c1-0a21-4d76-a795-2f2bcba86955
connection.interface-name:              --
connection.type:                        802-3-ethernet
connection.autoconnect:                 yes
connection.autoconnect-priority:        0
connection.timestamp:                   1470647387
```

```
connection.read-only:                   no
connection.permissions:
connection.zone:                        --
connection.master:                      --
connection.slave-type:                  --
connection.autoconnect-slaves:          -1 (default)
connection.secondaries:
connection.gateway-ping-timeout:        0
connection.metered:                     unknown
------------------------------------------------------------------------------
802-3-ethernet.port:                    --
802-3-ethernet.speed:                   0
802-3-ethernet.duplex:                  --
802-3-ethernet.auto-negotiate:          yes
802-3-ethernet.mac-address:             08:00:27:03:97:DD
802-3-ethernet.cloned-mac-address:      --
802-3-ethernet.mac-address-blacklist:
802-3-ethernet.mtu:                     auto
802-3-ethernet.s390-subchannels:
802-3-ethernet.s390-nettype:            --
802-3-ethernet.s390-options:
802-3-ethernet.wake-on-lan:             1 (default)
802-3-ethernet.wake-on-lan-password:    --
------------------------------------------------------------------------------
ipv4.method:                            auto
ipv4.dns:
ipv4.dns-search:
ipv4.addresses:
ipv4.gateway:                           --
ipv4.routes:
ipv4.route-metric:                      -1
ipv4.ignore-auto-routes:                no
ipv4.ignore-auto-dns:                   no
ipv4.dhcp-client-id:                    --
```

```
ipv4.dhcp-send-hostname:                yes
ipv4.dhcp-hostname:                     --
ipv4.never-default:                     no
ipv4.may-fail:                          yes
---------------------------------------------------------------------------
ipv6.method:                            auto
ipv6.dns:
ipv6.dns-search:
ipv6.addresses:
ipv6.gateway:                           --
ipv6.routes:
ipv6.route-metric:                      -1
ipv6.ignore-auto-routes:                no
ipv6.ignore-auto-dns:                   no
ipv6.never-default:                     no
ipv6.may-fail:                          yes
ipv6.ip6-privacy:                       -1 (unknown)
ipv6.dhcp-send-hostname:                yes
ipv6.dhcp-hostname:                     --
---------------------------------------------------------------------------
[root@centos7 ~]# nmcli -p connection show ip_fixe
===========================================================================
                    Connection profile details (ip_fixe)
===========================================================================
connection.id:                          ip_fixe
connection.uuid:                        fb3a11d9-4e03-4032-b26e-09d1195d2bcd
connection.interface-name:              enp0s3
connection.type:                        802-3-ethernet
connection.autoconnect:                 yes
connection.autoconnect-priority:        0
connection.timestamp:                   1470647577
connection.read-only:                   no
connection.permissions:
connection.zone:                        --
```

```
connection.master:                      --
connection.slave-type:                  --
connection.autoconnect-slaves:          -1 (default)
connection.secondaries:
connection.gateway-ping-timeout:        0
connection.metered:                     unknown
------------------------------------------------------------------------------
802-3-ethernet.port:                    --
802-3-ethernet.speed:                   0
802-3-ethernet.duplex:                  --
802-3-ethernet.auto-negotiate:          yes
802-3-ethernet.mac-address:             --
802-3-ethernet.cloned-mac-address:      --
802-3-ethernet.mac-address-blacklist:
802-3-ethernet.mtu:                     auto
802-3-ethernet.s390-subchannels:
802-3-ethernet.s390-nettype:            --
802-3-ethernet.s390-options:
802-3-ethernet.wake-on-lan:             1 (default)
802-3-ethernet.wake-on-lan-password:    --
------------------------------------------------------------------------------
ipv4.method:                            manual
ipv4.dns:
ipv4.dns-search:
ipv4.addresses:                         10.0.2.16/24
ipv4.gateway:                           10.0.2.2
ipv4.routes:
ipv4.route-metric:                      -1
ipv4.ignore-auto-routes:                no
ipv4.ignore-auto-dns:                   no
ipv4.dhcp-client-id:                    --
ipv4.dhcp-send-hostname:                yes
ipv4.dhcp-hostname:                     --
ipv4.never-default:                     no
```

```
ipv4.may-fail:                          yes
---------------------------------------------------------------------------
ipv6.method:                            auto
ipv6.dns:
ipv6.dns-search:
ipv6.addresses:
ipv6.gateway:                           --
ipv6.routes:
ipv6.route-metric:                      -1
ipv6.ignore-auto-routes:                no
ipv6.ignore-auto-dns:                   no
ipv6.never-default:                     no
ipv6.may-fail:                          yes
ipv6.ip6-privacy:                       -1 (unknown)
ipv6.dhcp-send-hostname:                yes
ipv6.dhcp-hostname:                     --
---------------------------------------------------------------------------
===========================================================================
      Activate connection details (fb3a11d9-4e03-4032-b26e-09d1195d2bcd)
===========================================================================
GENERAL.NAME:                           ip_fixe
GENERAL.UUID:                           fb3a11d9-4e03-4032-b26e-09d1195d2bcd
GENERAL.DEVICES:                        enp0s3
GENERAL.STATE:                          activated
GENERAL.DEFAULT:                        yes
GENERAL.DEFAULT6:                       no
GENERAL.VPN:                            no
GENERAL.ZONE:                           --
GENERAL.DBUS-PATH:                      /org/freedesktop/NetworkManager/ActiveConnection/3
GENERAL.CON-PATH:                       /org/freedesktop/NetworkManager/Settings/1
GENERAL.SPEC-OBJECT:                    /
GENERAL.MASTER-PATH:                    --
---------------------------------------------------------------------------
IP4.ADDRESS[1]:                         10.0.2.16/24
```

```
IP4.GATEWAY:                            10.0.2.2
------------------------------------------------------------------------------
IP6.ADDRESS[1]:                         fe80::a00:27ff:fe03:97dd/64
IP6.GATEWAY:
------------------------------------------------------------------------------
```

Pour consulter la liste profils associés à un périphérique, utilisez la commande suivante :

```
[root@centos7 ~]# nmcli -f CONNECTIONS device show enp0s3
CONNECTIONS.AVAILABLE-CONNECTION-PATHS: /org/freedesktop/NetworkManager/Settings/{0,1}
CONNECTIONS.AVAILABLE-CONNECTIONS[1]:   45b701c1-0a21-4d76-a795-2f2bcba86955 | Wired connection 1
CONNECTIONS.AVAILABLE-CONNECTIONS[2]:   fb3a11d9-4e03-4032-b26e-09d1195d2bcd | ip_fixe
```

Les fichiers de configuration pour le periphérique **enp0s3** se trouvent dans le répertoire **/etc/sysconfig/network-scripts/** :

```
[root@centos7 ~]# ls -l /etc/sysconfig/network-scripts/ | grep ifcfg
-rw-r--r--. 1 root root   296 Aug  8 11:08 ifcfg-ip_fixe
-rw-r--r--. 1 root root   254 Sep 16  2015 ifcfg-lo
```

L'étude du fichier **/etc/sysconfig/network-scripts/ifcfg-ip_fixe** démontre l'abscence de directives concernant les DNS :

```
[root@centos7 ~]# cat /etc/sysconfig/network-scripts/ifcfg-ip_fixe
TYPE=Ethernet
BOOTPROTO=none
IPADDR=10.0.2.16
PREFIX=24
GATEWAY=10.0.2.2
DEFROUTE=yes
IPV4_FAILURE_FATAL=no
IPV6INIT=yes
IPV6_AUTOCONF=yes
IPV6_DEFROUTE=yes
IPV6_PEERDNS=yes
IPV6_PEERROUTES=yes
```

```
IPV6_FAILURE_FATAL=no
NAME=ip_fixe
UUID=fb3a11d9-4e03-4032-b26e-09d1195d2bcd
DEVICE=enp0s3
ONBOOT=yes
```

La résolution des noms est donc inactive :

```
[root@centos7 ~]# ping www.free.fr
ping: unknown host www.free.fr
```

Modifiez donc la configuration du profil **ip_fixe** :

```
[root@centos7 ~]# nmcli connection mod ip_fixe ipv4.dns 8.8.8.8
```

L'étude du fichier **/etc/sysconfig/network-scripts/ifcfg-ip_fixe** démontre que la directive concernant le serveur DNS a été ajoutée :

```
[root@centos7 ~]# cat /etc/sysconfig/network-scripts/ifcfg-ip_fixe
TYPE=Ethernet
BOOTPROTO=none
DEFROUTE=yes
IPV4_FAILURE_FATAL=no
IPV6INIT=yes
IPV6_AUTOCONF=yes
IPV6_DEFROUTE=yes
IPV6_FAILURE_FATAL=no
NAME=ip_fixe
UUID=fb3a11d9-4e03-4032-b26e-09d1195d2bcd
DEVICE=enp0s3
ONBOOT=yes
IPADDR=10.0.2.16
PREFIX=24
GATEWAY=10.0.2.2
DNS1=8.8.8.8
```

```
IPV6_PEERDNS=yes
IPV6_PEERROUTES=yes
```

Afin que la modification du serveur DNS soit prise en compte, re-démarrez le service NetworkManager :

```
[root@centos7 ~]# systemctl restart NetworkManager.service
[root@centos7 ~]# systemctl status NetworkManager.service
● NetworkManager.service - Network Manager
   Loaded: loaded (/usr/lib/systemd/system/NetworkManager.service; enabled; vendor preset: enabled)
   Active: active (running) since Mon 2016-08-08 11:16:53 CEST; 7s ago
 Main PID: 8394 (NetworkManager)
   CGroup: /system.slice/NetworkManager.service
           └─8394 /usr/sbin/NetworkManager --no-daemon

Aug 08 11:16:53 centos7.fenestros.loc NetworkManager[8394]: <info>  (enp0s3): device state change: prepare ->
config (reason 'none') [40 50 0]
Aug 08 11:16:53 centos7.fenestros.loc NetworkManager[8394]: <info>  (enp0s3): device state change: config -> ip-
config (reason 'none') [50 70 0]
Aug 08 11:16:53 centos7.fenestros.loc NetworkManager[8394]: <info>  (enp0s3): device state change: ip-config ->
ip-check (reason 'none') [70 80 0]
Aug 08 11:16:53 centos7.fenestros.loc NetworkManager[8394]: <info>  (enp0s3): device state change: ip-check ->
secondaries (reason 'none') [80 90 0]
Aug 08 11:16:53 centos7.fenestros.loc NetworkManager[8394]: <info>  (enp0s3): device state change: secondaries ->
activated (reason 'none') [90 100 0]
Aug 08 11:16:53 centos7.fenestros.loc NetworkManager[8394]: <info>  NetworkManager state is now CONNECTED_LOCAL
Aug 08 11:16:53 centos7.fenestros.loc NetworkManager[8394]: <info>  NetworkManager state is now CONNECTED_GLOBAL
Aug 08 11:16:53 centos7.fenestros.loc NetworkManager[8394]: <info>  Policy set 'ip_fixe' (enp0s3) as default for
IPv4 routing and DNS.
Aug 08 11:16:53 centos7.fenestros.loc NetworkManager[8394]: <info>  (enp0s3): Activation: successful, device
activated.
Aug 08 11:16:53 centos7.fenestros.loc NetworkManager[8394]: <info>  wpa_supplicant running
```

Vérifiez que le fichier **/etc/resolv.conf** ait été modifié par NetworkManager :

```
[root@centos7 ~]# cat /etc/resolv.conf
# Generated by NetworkManager
search fenestros.loc
nameserver 8.8.8.8
```

Dernièrement vérifiez la resolution des noms :

```
[root@centos7 ~]# ping www.free.fr
PING www.free.fr (212.27.48.10) 56(84) bytes of data.
64 bytes from www.free.fr (212.27.48.10): icmp_seq=1 ttl=63 time=10.4 ms
64 bytes from www.free.fr (212.27.48.10): icmp_seq=2 ttl=63 time=9.44 ms
64 bytes from www.free.fr (212.27.48.10): icmp_seq=3 ttl=63 time=12.1 ms
^C
--- www.free.fr ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2002ms
rtt min/avg/max/mdev = 9.448/10.680/12.171/1.126 ms
```

Important : Notez qu'il existe un front-end graphique en mode texte, **nmtui**, pour configurer NetworkManager.

Ajouter une Deuxième Adresse IP à un Profil

Pour ajouter une deuxième adresse IP à un profil sous RHEL/CentOS 7, il convient d'utiliser la commande suivante :

```
[root@centos7 ~]# nmcli connection mod ip_fixe +ipv4.addresses 192.168.1.2/24
```

Redémarrez la machine virtuelle puis en tant que root saisissez la commande suivante :

```
[root@centos7 ~]# nmcli connection show ip_fixe
connection.id:                          ip_fixe
```

```
connection.uuid:                        fb3a11d9-4e03-4032-b26e-09d1195d2bcd
connection.interface-name:              enp0s3
connection.type:                        802-3-ethernet
connection.autoconnect:                 yes
connection.autoconnect-priority:        0
connection.timestamp:                   1470555543
connection.read-only:                   no
connection.permissions:
connection.zone:                        --
connection.master:                      --
connection.slave-type:                  --
connection.autoconnect-slaves:          -1 (default)
connection.secondaries:
connection.gateway-ping-timeout:        0
connection.metered:                     unknown
802-3-ethernet.port:                    --
802-3-ethernet.speed:                   0
802-3-ethernet.duplex:                  --
802-3-ethernet.auto-negotiate:          yes
802-3-ethernet.mac-address:             --
802-3-ethernet.cloned-mac-address:      --
802-3-ethernet.mac-address-blacklist:
802-3-ethernet.mtu:                     auto
802-3-ethernet.s390-subchannels:
802-3-ethernet.s390-nettype:            --
802-3-ethernet.s390-options:
802-3-ethernet.wake-on-lan:             1 (default)
802-3-ethernet.wake-on-lan-password:    --
ipv4.method:                            manual
ipv4.dns:                               8.8.8.8
ipv4.dns-search:
ipv4.addresses:                         10.0.2.16/24, 192.168.1.2/24
ipv4.gateway:                           10.0.2.2
ipv4.routes:
```

```
ipv4.route-metric:                      -1
ipv4.ignore-auto-routes:                no
ipv4.ignore-auto-dns:                   no
ipv4.dhcp-client-id:                    --
ipv4.dhcp-send-hostname:                yes
ipv4.dhcp-hostname:                     --
ipv4.never-default:                     no
ipv4.may-fail:                          yes
ipv6.method:                            auto
ipv6.dns:
ipv6.dns-search:
ipv6.addresses:
ipv6.gateway:                           --
ipv6.routes:
ipv6.route-metric:                      -1
ipv6.ignore-auto-routes:                no
ipv6.ignore-auto-dns:                   no
ipv6.never-default:                     no
ipv6.may-fail:                          yes
ipv6.ip6-privacy:                       -1 (unknown)
ipv6.dhcp-send-hostname:                yes
ipv6.dhcp-hostname:                     --
GENERAL.NAME:                           ip_fixe
GENERAL.UUID:                           fb3a11d9-4e03-4032-b26e-09d1195d2bcd
GENERAL.DEVICES:                        enp0s3
GENERAL.STATE:                          activated
GENERAL.DEFAULT:                        yes
GENERAL.DEFAULT6:                       no
GENERAL.VPN:                            no
GENERAL.ZONE:                           --
GENERAL.DBUS-PATH:                      /org/freedesktop/NetworkManager/ActiveConnection/0
GENERAL.CON-PATH:                       /org/freedesktop/NetworkManager/Settings/0
GENERAL.SPEC-OBJECT:                    /
GENERAL.MASTER-PATH:                    --
```

```
IP4.ADDRESS[1]:                         10.0.2.16/24
IP4.ADDRESS[2]:                         192.168.1.2/24
IP4.GATEWAY:                            10.0.2.2
IP4.DNS[1]:                             8.8.8.8
IP6.ADDRESS[1]:                         fe80::a00:27ff:fe03:97dd/64
IP6.GATEWAY:
```

Important : Notez l'ajout de la ligne **IP4.ADDRESS[2]:**.

Consultez maintenant le contenu du fichier **/etc/sysconfig/network-scripts/ifcfg-ip_fixe** :

```
[root@centos7 ~]# cat /etc/sysconfig/network-scripts/ifcfg-ip_fixe
TYPE=Ethernet
BOOTPROTO=none
DEFROUTE=yes
IPV4_FAILURE_FATAL=no
IPV6INIT=yes
IPV6_AUTOCONF=yes
IPV6_DEFROUTE=yes
IPV6_FAILURE_FATAL=no
NAME=ip_fixe
UUID=fb3a11d9-4e03-4032-b26e-09d1195d2bcd
DEVICE=enp0s3
ONBOOT=yes
DNS1=8.8.8.8
IPADDR=10.0.2.16
PREFIX=24
IPADDR1=192.168.1.2
PREFIX1=24
GATEWAY=10.0.2.2
IPV6_PEERDNS=yes
```

```
IPV6_PEERROUTES=yes
```

Important : Notez l'ajout de la ligne **IPADDR1=192.168.1.2**.

La Commande hostname

La procédure de la modification du hostname est simplifiée et sa prise en compte est immédiate :

```
[root@centos7 ~]# nmcli general hostname centos.fenestros.loc
[root@centos7 ~]# cat /etc/hostname
centos.fenestros.loc
[root@centos7 ~]# hostname
centos.fenestros.loc
[root@centos7 ~]# nmcli general hostname centos7.fenestros.loc
[root@centos7 ~]# cat /etc/hostname
centos7.fenestros.loc
[root@centos7 ~]# hostname
centos7.fenestros.loc
```

La Commande ip

Sous RHEL/CentOS 7 la commande **ip** est préférée par rapport à la commande ifconfig :

```
[root@centos7 ~]# ip address
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN
    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: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether 08:00:27:03:97:dd brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.16/24 brd 10.0.2.255 scope global enp0s3
       valid_lft forever preferred_lft forever
    inet 192.168.1.2/24 brd 192.168.1.255 scope global enp0s3
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe03:97dd/64 scope link
       valid_lft forever preferred_lft forever
[root@centos7 ~]# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN
    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: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether 08:00:27:03:97:dd brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.16/24 brd 10.0.2.255 scope global enp0s3
       valid_lft forever preferred_lft forever
    inet 192.168.1.2/24 brd 192.168.1.255 scope global enp0s3
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe03:97dd/64 scope link
       valid_lft forever preferred_lft forever
```

Options de la Commande ip

Les options de cette commande sont :

```
[root@centos7 ~]# ip --help
Usage: ip [ OPTIONS ] OBJECT { COMMAND | help }
       ip [ -force ] -batch filename
```

```
where  OBJECT := { link | addr | addrlabel | route | rule | neigh | ntable |
                   tunnel | tuntap | maddr | mroute | mrule | monitor | xfrm |
                   netns | l2tp | tcp_metrics | token }
       OPTIONS := { -V[ersion] | -s[tatistics] | -d[etails] | -r[esolve] |
                    -f[amily] { inet | inet6 | ipx | dnet | bridge | link } |
                    -4 | -6 | -I | -D | -B | -0 |
                    -l[oops] { maximum-addr-flush-attempts } |
                    -o[neline] | -t[imestamp] | -b[atch] [filename] |
                    -rc[vbuf] [size]}
```

Activer/Désactiver une Interface Manuellement

Deux commandes existent pour désactiver et activer manuellement une interface réseau :

```
[root@centos7 ~]# nmcli device disconnect enp0s3
[root@centos7 ~]# nmcli device connect enp0s3
```

Routage Statique

La commande ip

Sous RHEL/CentOS 7, pour supprimer la route vers le réseau 192.168.1.0 il convient d'utiliser la commande ip et non pas la commande route :

```
[root@centos7 ~]# ip route
default via 10.0.2.2 dev enp0s3  proto static  metric 100
10.0.2.0/24 dev enp0s3  proto kernel  scope link  src 10.0.2.16  metric 100
192.168.1.0/24 dev enp0s3  proto kernel  scope link  src 192.168.1.2  metric 100

[root@centos7 ~]# ip route del 192.168.1.0/24 via 0.0.0.0

[root@centos7 ~]# ip route
```

```
default via 10.0.2.2 dev enp0s3  proto static  metric 100
10.0.2.0/24 dev enp0s3  proto kernel  scope link  src 10.0.2.16  metric 100
```

Pour ajouter la route vers le réseau 192.168.1.0 :

```
[root@centos7 ~]# ip route add 192.168.1.0/24 via 10.0.2.2

[root@centos7 ~]# ip route
default via 10.0.2.2 dev enp0s3  proto static  metric 100
10.0.2.0/24 dev enp0s3  proto kernel  scope link  src 10.0.2.16  metric 100
192.168.1.0/24 via 10.0.2.2 dev enp0s3
```

La commande utilisée pour ajouter une passerelle par défaut prend la forme suivante **ip route add default via *adresse ip***.

Activer/désactiver le routage sur le serveur

Pour activer le routage sur le serveur, il convient d'activer la retransmission des paquets:

```
[root@centos7 ~]# echo 1 > /proc/sys/net/ipv4/ip_forward
[root@centos7 ~]# cat /proc/sys/net/ipv4/ip_forward
1
```

Pour désactiver le routage sur le serveur, il convient de désactiver la retransmission des paquets:

```
[root@centos7 ~]# echo 0 > /proc/sys/net/ipv4/ip_forward
[root@centos7 ~]# cat /proc/sys/net/ipv4/ip_forward
0
```

LAB #4 - Utilisation de nmap et de netcat

4.1 - nmap

Installation

Sous RHEL/CentOS 7, **nmap** n'est pas installé par défaut :

```
[root@centos7 ~]# which nmap
/usr/bin/which: no nmap in (/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin:/root/bin)
```

Installez donc nmap en utilisant yum :

```
[root@centos7 ~]# yum install nmap
Loaded plugins: fastestmirror, langpacks
Repodata is over 2 weeks old. Install yum-cron? Or run: yum makecache fast
adobe-linux-x86_64                                       | 2.9 kB     00:00
base                                                     | 3.6 kB     00:00
extras                                                   | 3.4 kB     00:00
updates                                                  | 3.4 kB     00:00
(1/3): adobe-linux-x86_64/primary_db                       | 2.7 kB   00:00
(2/3): extras/7/x86_64/primary_db                          | 191 kB   00:00
(3/3): updates/7/x86_64/primary_db                         | 7.8 MB   00:04
Determining fastest mirrors
 * base: ftp.rezopole.net
 * extras: ftp.rezopole.net
 * updates: ftp.rezopole.net
Resolving Dependencies
--> Running transaction check
---> Package nmap.x86_64 2:6.40-7.el7 will be installed
--> Processing Dependency: nmap-ncat = 2:6.40-7.el7 for package: 2:nmap-6.40-7.el7.x86_64
```

```
--> Running transaction check
---> Package nmap-ncat.x86_64 2:6.40-7.el7 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package            Arch            Version                  Repository     Size
================================================================================
Installing:
 nmap               x86_64          2:6.40-7.el7             base          4.0 M
Installing for dependencies:
 nmap-ncat          x86_64          2:6.40-7.el7             base          201 k

Transaction Summary
================================================================================
Install  1 Package (+1 Dependent package)

Total download size: 4.2 M
Installed size: 17 M
Is this ok [y/d/N]: y
```

Les options de cette commande sont :

```
[root@centos7 ~]# nmap --help
Nmap 6.40 ( http://nmap.org )
Usage: nmap [Scan Type(s)] [Options] {target specification}
TARGET SPECIFICATION:
  Can pass hostnames, IP addresses, networks, etc.
  Ex: scanme.nmap.org, microsoft.com/24, 192.168.0.1; 10.0.0-255.1-254
  -iL <inputfilename>: Input from list of hosts/networks
  -iR <num hosts>: Choose random targets
  --exclude <host1[,host2][,host3],...>: Exclude hosts/networks
  --excludefile <exclude_file>: Exclude list from file
```

```
HOST DISCOVERY:
  -sL: List Scan - simply list targets to scan
  -sn: Ping Scan - disable port scan
  -Pn: Treat all hosts as online -- skip host discovery
  -PS/PA/PU/PY[portlist]: TCP SYN/ACK, UDP or SCTP discovery to given ports
  -PE/PP/PM: ICMP echo, timestamp, and netmask request discovery probes
  -PO[protocol list]: IP Protocol Ping
  -n/-R: Never do DNS resolution/Always resolve [default: sometimes]
  --dns-servers <serv1[,serv2],...>: Specify custom DNS servers
  --system-dns: Use OS's DNS resolver
  --traceroute: Trace hop path to each host
SCAN TECHNIQUES:
  -sS/sT/sA/sW/sM: TCP SYN/Connect()/ACK/Window/Maimon scans
  -sU: UDP Scan
  -sN/sF/sX: TCP Null, FIN, and Xmas scans
  --scanflags <flags>: Customize TCP scan flags
  -sI <zombie host[:probeport]>: Idle scan
  -sY/sZ: SCTP INIT/COOKIE-ECHO scans
  -sO: IP protocol scan
  -b <FTP relay host>: FTP bounce scan
PORT SPECIFICATION AND SCAN ORDER:
  -p <port ranges>: Only scan specified ports
    Ex: -p22; -p1-65535; -p U:53,111,137,T:21-25,80,139,8080,S:9
  -F: Fast mode - Scan fewer ports than the default scan
  -r: Scan ports consecutively - don't randomize
  --top-ports <number>: Scan <number> most common ports
  --port-ratio <ratio>: Scan ports more common than <ratio>
SERVICE/VERSION DETECTION:
  -sV: Probe open ports to determine service/version info
  --version-intensity <level>: Set from 0 (light) to 9 (try all probes)
  --version-light: Limit to most likely probes (intensity 2)
  --version-all: Try every single probe (intensity 9)
  --version-trace: Show detailed version scan activity (for debugging)
SCRIPT SCAN:
```

```
  -sC: equivalent to --script=default
  --script=<Lua scripts>: <Lua scripts> is a comma separated list of
           directories, script-files or script-categories
  --script-args=<n1=v1,[n2=v2,...]>: provide arguments to scripts
  --script-args-file=filename: provide NSE script args in a file
  --script-trace: Show all data sent and received
  --script-updatedb: Update the script database.
  --script-help=<Lua scripts>: Show help about scripts.
           <Lua scripts> is a comma separted list of script-files or
           script-categories.
OS DETECTION:
  -O: Enable OS detection
  --osscan-limit: Limit OS detection to promising targets
  --osscan-guess: Guess OS more aggressively
TIMING AND PERFORMANCE:
  Options which take <time> are in seconds, or append 'ms' (milliseconds),
  's' (seconds), 'm' (minutes), or 'h' (hours) to the value (e.g. 30m).
  -T<0-5>: Set timing template (higher is faster)
  --min-hostgroup/max-hostgroup <size>: Parallel host scan group sizes
  --min-parallelism/max-parallelism <numprobes>: Probe parallelization
  --min-rtt-timeout/max-rtt-timeout/initial-rtt-timeout <time>: Specifies
      probe round trip time.
  --max-retries <tries>: Caps number of port scan probe retransmissions.
  --host-timeout <time>: Give up on target after this long
  --scan-delay/--max-scan-delay <time>: Adjust delay between probes
  --min-rate <number>: Send packets no slower than <number> per second
  --max-rate <number>: Send packets no faster than <number> per second
FIREWALL/IDS EVASION AND SPOOFING:
  -f; --mtu <val>: fragment packets (optionally w/given MTU)
  -D <decoy1,decoy2[,ME],...>: Cloak a scan with decoys
  -S <IP_Address>: Spoof source address
  -e <iface>: Use specified interface
  -g/--source-port <portnum>: Use given port number
  --data-length <num>: Append random data to sent packets
```

```
  --ip-options <options>: Send packets with specified ip options
  --ttl <val>: Set IP time-to-live field
  --spoof-mac <mac address/prefix/vendor name>: Spoof your MAC address
  --badsum: Send packets with a bogus TCP/UDP/SCTP checksum
OUTPUT:
  -oN/-oX/-oS/-oG <file>: Output scan in normal, XML, s|<rIpt kIddi3,
     and Grepable format, respectively, to the given filename.
  -oA <basename>: Output in the three major formats at once
  -v: Increase verbosity level (use -vv or more for greater effect)
  -d: Increase debugging level (use -dd or more for greater effect)
  --reason: Display the reason a port is in a particular state
  --open: Only show open (or possibly open) ports
  --packet-trace: Show all packets sent and received
  --iflist: Print host interfaces and routes (for debugging)
  --log-errors: Log errors/warnings to the normal-format output file
  --append-output: Append to rather than clobber specified output files
  --resume <filename>: Resume an aborted scan
  --stylesheet <path/URL>: XSL stylesheet to transform XML output to HTML
  --webxml: Reference stylesheet from Nmap.Org for more portable XML
  --no-stylesheet: Prevent associating of XSL stylesheet w/XML output
MISC:
  -6: Enable IPv6 scanning
  -A: Enable OS detection, version detection, script scanning, and traceroute
  --datadir <dirname>: Specify custom Nmap data file location
  --send-eth/--send-ip: Send using raw ethernet frames or IP packets
  --privileged: Assume that the user is fully privileged
  --unprivileged: Assume the user lacks raw socket privileges
  -V: Print version number
  -h: Print this help summary page.
EXAMPLES:
  nmap -v -A scanme.nmap.org
  nmap -v -sn 192.168.0.0/16 10.0.0.0/8
  nmap -v -iR 10000 -Pn -p 80
SEE THE MAN PAGE (http://nmap.org/book/man.html) FOR MORE OPTIONS AND EXAMPLES
```

Utilisation

Pour connaître la liste des ports ouverts sur votre machine virtuelle, saisissez la commande suivante :

```
[root@centos7 ~]# nmap 127.0.0.1

Starting Nmap 6.40 ( http://nmap.org ) at 2017-08-05 14:17 CEST
Nmap scan report for localhost.localdomain (127.0.0.1)
Host is up (-2100s latency).
Not shown: 996 closed ports
PORT    STATE SERVICE
22/tcp  open  ssh
25/tcp  open  smtp
111/tcp open  rpcbind
631/tcp open  ipp

Nmap done: 1 IP address (1 host up) scanned in 0.06 seconds
```

> **Important** - Pour connaître les ports ouverts sur une machine distante, la procédure est identique sauf que vous devez utiliser l'adresse IP de votre cible.

Fichiers de Configuration

nmap utilise un fichier spécifique pour identifier les ports. Ce fichier est **/usr/share/nmap/nmap-services**:

```
[root@centos7 ~]# more /usr/share/nmap/nmap-services
# THIS FILE IS GENERATED AUTOMATICALLY FROM A MASTER - DO NOT EDIT.
# EDIT /nmap-private-dev/nmap-services-all IN SVN INSTEAD.
```

```
# Well known service port numbers -*- mode: fundamental; -*-
# From the Nmap Security Scanner ( http://nmap.org )
#
# $Id: nmap-services 31220 2013-07-03 04:30:43Z david $
#
# Derived from IANA data and our own research
#
# This collection of service data is (C) 1996-2011 by Insecure.Com
# LLC.  It is distributed under the Nmap Open Source license as
# provided in the COPYING file of the source distribution or at
# http://nmap.org/data/COPYING .  Note that this license
# requires you to license your own work under a compatable open source
# license.  If you wish to embed Nmap technology into proprietary
# software, we sell alternative licenses (contact sales@insecure.com).
# Dozens of software vendors already license Nmap technology such as
# host discovery, port scanning, OS detection, and version detection.
# For more details, see http://nmap.org/book/man-legal.html
#
# Fields in this file are: Service name, portnum/protocol, open-frequency, optional comments
#
tcpmux  1/tcp    0.001995    # TCP Port Service Multiplexer [rfc-1078]
tcpmux  1/udp    0.001236    # TCP Port Service Multiplexer
compressnet 2/tcp    0.000013    # Management Utility
compressnet 2/udp    0.001845    # Management Utility
compressnet 3/tcp    0.001242    # Compression Process
compressnet 3/udp    0.001532    # Compression Process
unknown 4/tcp    0.000477
rje 5/udp    0.000593    # Remote Job Entry
unknown 6/tcp    0.000502
echo    7/sctp    0.000000
echo    7/tcp    0.004855
echo    7/udp    0.024679
unknown 8/tcp    0.000013
--More--(0%)
```

Le répertoire **/usr/share/nmap** contient d'autres fichiers importants :

```
[root@centos7 ~]# ls -l /usr/share/nmap
total 6548
-rw-r--r--. 1 root root   10546 Nov 20  2015 nmap.dtd
-rw-r--r--. 1 root root  455371 Nov 20  2015 nmap-mac-prefixes
-rw-r--r--. 1 root root 3694559 Nov 20  2015 nmap-os-db
-rw-r--r--. 1 root root   11749 Nov 20  2015 nmap-payloads
-rw-r--r--. 1 root root    6631 Nov 20  2015 nmap-protocols
-rw-r--r--. 1 root root   49243 Nov 20  2015 nmap-rpc
-rw-r--r--. 1 root root 1727204 Nov 20  2015 nmap-service-probes
-rw-r--r--. 1 root root  622039 Nov 20  2015 nmap-services
-rw-r--r--. 1 root root   31935 Nov 20  2015 nmap.xsl
drwxr-xr-x. 3 root root    4096 Aug  5 14:16 nselib
-rw-r--r--. 1 root root   47190 Nov 20  2015 nse_main.lua
drwxr-xr-x. 2 root root   20480 Aug  5 14:16 scripts
```

Voici la liste des fichiers les plus importants :

Fichier	**Description**
/usr/share/nmap/nmap-protocols	Contient la liste des protocols reconnus par **nmap**.
/usr/share/nmap/nmap-service-probes	Contient les règles de balayage utilisées par **nmap** pour identifier le service actif sur un port donné.
/usr/share/nmap/nmap-mac-prefixes	Contient une liste de préfix d'adresses MAC par fabricant reconnu par **nmap**.
/usr/share/nmap/nmap-rpc	Contient une liste des services RPC reconnus par **nmap**.

Scripts

nmap utilise des scripts pour accomplir certaines tâches allant de la découverte simple de ports ouverts jusqu'à l'intrusion :

```
[root@centos7 ~]# ls /usr/share/nmap/scripts/
acarsd-info.nse                     http-domino-enum-passwords.nse          ndmp-version.nse
address-info.nse                    http-drupal-enum-users.nse              nessus-brute.nse
afp-brute.nse                       http-drupal-modules.nse                 nessus-xmlrpc-brute.nse
```

```
afp-ls.nse                          http-email-harvest.nse                  netbus-auth-bypass.nse
afp-path-vuln.nse                   http-enum.nse                           netbus-brute.nse
afp-serverinfo.nse                  http-exif-spider.nse                    netbus-info.nse
afp-showmount.nse                   http-favicon.nse                        netbus-version.nse
ajp-auth.nse                        http-fileupload-exploiter.nse           nexpose-brute.nse
ajp-brute.nse                       http-form-brute.nse                     nfs-ls.nse
ajp-headers.nse                     http-form-fuzzer.nse                    nfs-showmount.nse
ajp-methods.nse                     http-frontpage-login.nse                nfs-statfs.nse
ajp-request.nse                     http-generator.nse                      nping-brute.nse
amqp-info.nse                       http-git.nse                            nrpe-enum.nse
asn-query.nse                       http-gitweb-projects-enum.nse           ntp-info.nse
auth-owners.nse                     http-google-malware.nse                 ntp-monlist.nse
auth-spoof.nse                      http-grep.nse                           omp2-brute.nse
backorifice-brute.nse               http-headers.nse                        omp2-enum-targets.nse
backorifice-info.nse                http-huawei-hg5xx-vuln.nse              openlookup-info.nse
banner.nse                          http-icloud-findmyiphone.nse            openvas-otp-brute.nse
bitcoin-getaddr.nse                 http-icloud-sendmsg.nse                 oracle-brute.nse
bitcoin-info.nse                    http-iis-webdav-vuln.nse                oracle-brute-stealth.nse
bitcoinrpc-info.nse                 http-joomla-brute.nse                   oracle-enum-users.nse
bittorrent-discovery.nse            http-litespeed-sourcecode-download.nse  oracle-sid-brute.nse
bjnp-discover.nse                   http-majordomo2-dir-traversal.nse       ovs-agent-version.nse
broadcast-ataoe-discover.nse        http-malware-host.nse                   p2p-conficker.nse
broadcast-avahi-dos.nse             http-methods.nse                        path-mtu.nse
broadcast-bjnp-discover.nse         http-method-tamper.nse                  pcanywhere-brute.nse
broadcast-db2-discover.nse          http-open-proxy.nse                     pgsql-brute.nse
broadcast-dhcp6-discover.nse        http-open-redirect.nse                  pjl-ready-message.nse
broadcast-dhcp-discover.nse         http-passwd.nse                         pop3-brute.nse
broadcast-dns-service-discovery.nse http-phpmyadmin-dir-traversal.nse       pop3-capabilities.nse
broadcast-dropbox-listener.nse      http-phpself-xss.nse                    pptp-version.nse
broadcast-eigrp-discovery.nse       http-php-version.nse                    qscan.nse
broadcast-igmp-discovery.nse        http-proxy-brute.nse                    quake3-info.nse
broadcast-listener.nse              http-put.nse                            quake3-master-getservers.nse
broadcast-ms-sql-discover.nse       http-qnap-nas-info.nse                  rdp-enum-encryption.nse
broadcast-netbios-master-browser.nse http-rfi-spider.nse                    rdp-vuln-ms12-020.nse
```

```
broadcast-networker-discover.nse
broadcast-novell-locate.nse
broadcast-pc-anywhere.nse
broadcast-pc-duo.nse
broadcast-pim-discovery.nse
broadcast-ping.nse
broadcast-pppoe-discover.nse
broadcast-rip-discover.nse
broadcast-ripng-discover.nse
broadcast-sybase-asa-discover.nse
broadcast-tellstick-discover.nse
broadcast-upnp-info.nse
broadcast-versant-locate.nse
broadcast-wake-on-lan.nse
broadcast-wpad-discover.nse
broadcast-wsdd-discover.nse
broadcast-xdmcp-discover.nse
cassandra-brute.nse
cassandra-info.nse
cccam-version.nse
citrix-brute-xml.nse
citrix-enum-apps.nse
citrix-enum-apps-xml.nse
citrix-enum-servers.nse
citrix-enum-servers-xml.nse
couchdb-databases.nse
couchdb-stats.nse
creds-summary.nse
cups-info.nse
cups-queue-info.nse
cvs-brute.nse
cvs-brute-repository.nse
daap-get-library.nse
daytime.nse
http-robots.txt.nse
http-robtex-reverse-ip.nse
http-robtex-shared-ns.nse
http-sitemap-generator.nse
http-slowloris-check.nse
http-slowloris.nse
http-sql-injection.nse
http-stored-xss.nse
http-title.nse
http-tplink-dir-traversal.nse
http-trace.nse
http-traceroute.nse
http-unsafe-output-escaping.nse
http-userdir-enum.nse
http-vhosts.nse
http-virustotal.nse
http-vlcstreamer-ls.nse
http-vmware-path-vuln.nse
http-vuln-cve2009-3960.nse
http-vuln-cve2010-0738.nse
http-vuln-cve2010-2861.nse
http-vuln-cve2011-3192.nse
http-vuln-cve2011-3368.nse
http-vuln-cve2012-1823.nse
http-vuln-cve2013-0156.nse
http-waf-detect.nse
http-waf-fingerprint.nse
http-wordpress-brute.nse
http-wordpress-enum.nse
http-wordpress-plugins.nse
iax2-brute.nse
iax2-version.nse
icap-info.nse
ike-version.nse
realvnc-auth-bypass.nse
redis-brute.nse
redis-info.nse
resolveall.nse
reverse-index.nse
rexec-brute.nse
riak-http-info.nse
rlogin-brute.nse
rmi-dumpregistry.nse
rmi-vuln-classloader.nse
rpcap-brute.nse
rpcap-info.nse
rpc-grind.nse
rpcinfo.nse
rsync-brute.nse
rsync-list-modules.nse
rtsp-methods.nse
rtsp-url-brute.nse
samba-vuln-cve-2012-1182.nse
script.db
servicetags.nse
sip-brute.nse
sip-call-spoof.nse
sip-enum-users.nse
sip-methods.nse
skypev2-version.nse
smb-brute.nse
smb-check-vulns.nse
smb-enum-domains.nse
smb-enum-groups.nse
smb-enum-processes.nse
smb-enum-sessions.nse
smb-enum-shares.nse
smb-enum-users.nse
```

```
db2-das-info.nse
db2-discover.nse
dhcp-discover.nse
dict-info.nse
distcc-cve2004-2687.nse
dns-blacklist.nse
dns-brute.nse
dns-cache-snoop.nse
dns-check-zone.nse
dns-client-subnet-scan.nse
dns-fuzz.nse
dns-ip6-arpa-scan.nse
dns-nsec3-enum.nse
dns-nsec-enum.nse
dns-nsid.nse
dns-random-srcport.nse
dns-random-txid.nse
dns-recursion.nse
dns-service-discovery.nse
dns-srv-enum.nse
dns-update.nse
dns-zeustracker.nse
dns-zone-transfer.nse
domcon-brute.nse
domcon-cmd.nse
domino-enum-users.nse
dpap-brute.nse
drda-brute.nse
drda-info.nse
duplicates.nse
eap-info.nse
epmd-info.nse
eppc-enum-processes.nse
finger.nse
imap-brute.nse
imap-capabilities.nse
informix-brute.nse
informix-query.nse
informix-tables.nse
ip-forwarding.nse
ip-geolocation-geobytes.nse
ip-geolocation-geoplugin.nse
ip-geolocation-ipinfodb.nse
ip-geolocation-maxmind.nse
ipidseq.nse
ipv6-node-info.nse
ipv6-ra-flood.nse
irc-botnet-channels.nse
irc-brute.nse
irc-info.nse
irc-sasl-brute.nse
irc-unrealircd-backdoor.nse
iscsi-brute.nse
iscsi-info.nse
isns-info.nse
jdwp-exec.nse
jdwp-info.nse
jdwp-inject.nse
jdwp-version.nse
krb5-enum-users.nse
ldap-brute.nse
ldap-novell-getpass.nse
ldap-rootdse.nse
ldap-search.nse
lexmark-config.nse
llmnr-resolve.nse
lltd-discovery.nse
maxdb-info.nse
smb-flood.nse
smb-ls.nse
smb-mbenum.nse
smb-os-discovery.nse
smb-print-text.nse
smb-psexec.nse
smb-security-mode.nse
smb-server-stats.nse
smb-system-info.nse
smbv2-enabled.nse
smb-vuln-ms10-054.nse
smb-vuln-ms10-061.nse
smtp-brute.nse
smtp-commands.nse
smtp-enum-users.nse
smtp-open-relay.nse
smtp-strangeport.nse
smtp-vuln-cve2010-4344.nse
smtp-vuln-cve2011-1720.nse
smtp-vuln-cve2011-1764.nse
sniffer-detect.nse
snmp-brute.nse
snmp-hh3c-logins.nse
snmp-interfaces.nse
snmp-ios-config.nse
snmp-netstat.nse
snmp-processes.nse
snmp-sysdescr.nse
snmp-win32-services.nse
snmp-win32-shares.nse
snmp-win32-software.nse
snmp-win32-users.nse
socks-auth-info.nse
socks-brute.nse
```

```
firewalk.nse                        mcafee-epo-agent.nse                socks-open-proxy.nse
firewall-bypass.nse                 membase-brute.nse                   ssh2-enum-algos.nse
flume-master-info.nse               membase-http-info.nse               ssh-hostkey.nse
ftp-anon.nse                        memcached-info.nse                  sshv1.nse
ftp-bounce.nse                      metasploit-info.nse                 ssl-cert.nse
ftp-brute.nse                       metasploit-msgrpc-brute.nse         ssl-date.nse
ftp-libopie.nse                     metasploit-xmlrpc-brute.nse         ssl-enum-ciphers.nse
ftp-proftpd-backdoor.nse            mmouse-brute.nse                    ssl-google-cert-catalog.nse
ftp-vsftpd-backdoor.nse             mmouse-exec.nse                     ssl-known-key.nse
ftp-vuln-cve2010-4221.nse           modbus-discover.nse                 sslv2.nse
ganglia-info.nse                    mongodb-brute.nse                   stun-info.nse
giop-info.nse                       mongodb-databases.nse               stun-version.nse
gkrellm-info.nse                    mongodb-info.nse                    stuxnet-detect.nse
gopher-ls.nse                       mrinfo.nse                          svn-brute.nse
gpsd-info.nse                       msrpc-enum.nse                      targets-asn.nse
hadoop-datanode-info.nse            ms-sql-brute.nse                    targets-ipv6-multicast-echo.nse
hadoop-jobtracker-info.nse          ms-sql-config.nse                   targets-ipv6-multicast-invalid-
dst.nse
hadoop-namenode-info.nse            ms-sql-dac.nse                      targets-ipv6-multicast-mld.nse
hadoop-secondary-namenode-info.nse  ms-sql-dump-hashes.nse              targets-ipv6-multicast-slaac.nse
hadoop-tasktracker-info.nse         ms-sql-empty-password.nse           targets-sniffer.nse
hbase-master-info.nse               ms-sql-hasdbaccess.nse              targets-traceroute.nse
hbase-region-info.nse               ms-sql-info.nse                     teamspeak2-version.nse
hddtemp-info.nse                    ms-sql-query.nse                    telnet-brute.nse
hostmap-bfk.nse                     ms-sql-tables.nse                   telnet-encryption.nse
hostmap-ip2hosts.nse                ms-sql-xp-cmdshell.nse              tftp-enum.nse
hostmap-robtex.nse                  mtrace.nse                          tls-nextprotoneg.nse
http-adobe-coldfusion-apsa1301.nse  murmur-version.nse                  traceroute-geolocation.nse
http-affiliate-id.nse               mysql-audit.nse                     unusual-port.nse
http-apache-negotiation.nse         mysql-brute.nse                     upnp-info.nse
http-auth-finder.nse                mysql-databases.nse                 url-snarf.nse
http-auth.nse                       mysql-dump-hashes.nse               ventrilo-info.nse
http-awstatstotals-exec.nse         mysql-empty-password.nse            versant-info.nse
http-axis2-dir-traversal.nse        mysql-enum.nse                      vmauthd-brute.nse
```

```
http-backup-finder.nse              mysql-info.nse                  vnc-brute.nse
http-barracuda-dir-traversal.nse    mysql-query.nse                 vnc-info.nse
http-brute.nse                      mysql-users.nse                 voldemort-info.nse
http-cakephp-version.nse            mysql-variables.nse             vuze-dht-info.nse
http-chrono.nse                     mysql-vuln-cve2012-2122.nse     wdb-version.nse
http-coldfusion-subzero.nse         nat-pmp-info.nse                whois.nse
http-comments-displayer.nse         nat-pmp-mapport.nse             wsdd-discover.nse
http-config-backup.nse              nbstat.nse                      x11-access.nse
http-cors.nse                       ncp-enum-users.nse              xdmcp-discover.nse
http-date.nse                       ncp-serverinfo.nse              xmpp-brute.nse
http-default-accounts.nse           ndmp-fs-info.nse                xmpp-info.nse
```

Les scripts sont regroupés dans des catégories : **auth**, **broadcast**, **brute**, **default**, **discovery**, **dos**, **exploit**, **external**, **fuzzer**, **intrusive**, **malware**, **safe**, **version** and **vuln**.

> **Important** - Pour plus d'informations concernant ces catégories, consultez cette page.

La catégorie la plus utilisée est **default** qui est appelée par l'utilisation de l'option **-sC**. Cette catégorie contient une liste de scripts par défaut.

```
[root@centos7 ~]# nmap -v -sC localhost

Starting Nmap 6.40 ( http://nmap.org ) at 2017-08-05 14:20 CEST
NSE: Loaded 95 scripts for scanning.
NSE: Script Pre-scanning.
Initiating SYN Stealth Scan at 14:20
Scanning localhost (127.0.0.1) [1000 ports]
Discovered open port 22/tcp on 127.0.0.1
adjust_timeouts2: packet supposedly had rtt of -1500757317045342 microseconds.  Ignoring time.
adjust_timeouts2: packet supposedly had rtt of -1500757317045342 microseconds.  Ignoring time.
Discovered open port 25/tcp on 127.0.0.1
```

```
adjust_timeouts2: packet supposedly had rtt of -1500757317045486 microseconds.  Ignoring time.
adjust_timeouts2: packet supposedly had rtt of -1500757317045486 microseconds.  Ignoring time.
Discovered open port 111/tcp on 127.0.0.1
adjust_timeouts2: packet supposedly had rtt of -1500757317045504 microseconds.  Ignoring time.
adjust_timeouts2: packet supposedly had rtt of -1500757317045504 microseconds.  Ignoring time.
Discovered open port 631/tcp on 127.0.0.1
adjust_timeouts2: packet supposedly had rtt of -1500757274107480 microseconds.  Ignoring time.
adjust_timeouts2: packet supposedly had rtt of -1500757274107480 microseconds.  Ignoring time.
Completed SYN Stealth Scan at 14:20, 0.01s elapsed (1000 total ports)
NSE: Script scanning 127.0.0.1.
Initiating NSE at 14:20
Completed NSE at 14:20, 0.28s elapsed
Nmap scan report for localhost (127.0.0.1)
Host is up (0.0000060s latency).
rDNS record for 127.0.0.1: localhost.localdomain
Not shown: 996 closed ports
PORT    STATE SERVICE
22/tcp  open  ssh
| ssh-hostkey: 2048 17:21:e0:43:b1:66:22:22:b6:f8:2b:cc:08:68:38:59 (RSA)
|_256 19:cd:05:58:af:2c:10:82:52:ba:e3:31:df:bd:72:54 (ECDSA)
25/tcp  open  smtp
|_smtp-commands: centos7.fenestros.loc, PIPELINING, SIZE 10240000, VRFY, ETRN, ENHANCEDSTATUSCODES, 8BITMIME,
DSN,
111/tcp open  rpcbind
| rpcinfo:
|   program version   port/proto  service
|   100000  2,3,4        111/tcp  rpcbind
|_  100000  2,3,4        111/udp  rpcbind
631/tcp open  ipp
| http-methods: GET HEAD OPTIONS POST PUT
| Potentially risky methods: PUT
|_See http://nmap.org/nsedoc/scripts/http-methods.html
| http-robots.txt: 1 disallowed entry
|_/
```

```
|_http-title: Home - CUPS 1.6.3

NSE: Script Post-scanning.
Initiating NSE at 14:20
Completed NSE at 14:20, 0.00s elapsed
Read data files from: /usr/bin/../share/nmap
Nmap done: 1 IP address (1 host up) scanned in 0.43 seconds
           Raw packets sent: 1000 (44.000KB) | Rcvd: 2004 (84.176KB)
```

Attention - La catégorie par défaut **default** contient certains scripts de la catégorie **intrusive**. Vous ne devez donc jamais utiliser cette option sur un réseau sans avoir obtenu un accord au préalable.

4.2 - netcat

netcat est un couteau suisse. Il permet non seulement de scanner des ports mais aussi de lancer la connexion lors de la découverte d'un port ouvert.

Les options de cette commande sont :

```
[root@centos7 ~]# nc --help
Ncat 6.40 ( http://nmap.org/ncat )
Usage: ncat [options] [hostname] [port]

Options taking a time assume seconds. Append 'ms' for milliseconds,
's' for seconds, 'm' for minutes, or 'h' for hours (e.g. 500ms).
  -4                         Use IPv4 only
  -6                         Use IPv6 only
  -U, --unixsock             Use Unix domain sockets only
  -C, --crlf                 Use CRLF for EOL sequence
  -c, --sh-exec <command>    Executes the given command via /bin/sh
```

```
  -e, --exec <command>       Executes the given command
      --lua-exec <filename>  Executes the given Lua script
  -g hop1[,hop2,...]         Loose source routing hop points (8 max)
  -G <n>                     Loose source routing hop pointer (4, 8, 12, ...)
  -m, --max-conns <n>        Maximum <n> simultaneous connections
  -h, --help                 Display this help screen
  -d, --delay <time>         Wait between read/writes
  -o, --output <filename>    Dump session data to a file
  -x, --hex-dump <filename>  Dump session data as hex to a file
  -i, --idle-timeout <time>  Idle read/write timeout
  -p, --source-port port     Specify source port to use
  -s, --source addr          Specify source address to use (doesn't affect -l)
  -l, --listen               Bind and listen for incoming connections
  -k, --keep-open            Accept multiple connections in listen mode
  -n, --nodns                Do not resolve hostnames via DNS
  -t, --telnet               Answer Telnet negotiations
  -u, --udp                  Use UDP instead of default TCP
      --sctp                 Use SCTP instead of default TCP
  -v, --verbose              Set verbosity level (can be used several times)
  -w, --wait <time>          Connect timeout
      --append-output        Append rather than clobber specified output files
      --send-only            Only send data, ignoring received; quit on EOF
      --recv-only            Only receive data, never send anything
      --allow                Allow only given hosts to connect to Ncat
      --allowfile            A file of hosts allowed to connect to Ncat
      --deny                 Deny given hosts from connecting to Ncat
      --denyfile             A file of hosts denied from connecting to Ncat
      --broker               Enable Ncat's connection brokering mode
      --chat                 Start a simple Ncat chat server
      --proxy <addr[:port]>  Specify address of host to proxy through
      --proxy-type <type>    Specify proxy type ("http" or "socks4")
      --proxy-auth <auth>    Authenticate with HTTP or SOCKS proxy server
      --ssl                  Connect or listen with SSL
      --ssl-cert             Specify SSL certificate file (PEM) for listening
```

```
      --ssl-key               Specify SSL private key (PEM) for listening
      --ssl-verify            Verify trust and domain name of certificates
      --ssl-trustfile         PEM file containing trusted SSL certificates
      --version               Display Ncat's version information and exit

See the ncat(1) manpage for full options, descriptions and usage examples
```

Utilisation

Dans l'exemple qui suit, un scan est lancé sur le port 80 puis sur le port 25 :

```
[root@centos7 ~]# nc 127.0.0.1 80 -w 1 -vv
Ncat: Version 6.40 ( http://nmap.org/ncat )
libnsock nsi_new2(): nsi_new (IOD #1)
libnsock nsock_connect_tcp(): TCP connection requested to 127.0.0.1:80 (IOD #1) EID 8
libnsock nsock_trace_handler_callback(): Callback: CONNECT ERROR [Connection refused (111)] for EID 8
[127.0.0.1:80]
Ncat: Connection refused.

[root@centos7 ~]# nc 127.0.0.1 25 -w 1 -vv
Ncat: Version 6.40 ( http://nmap.org/ncat )
libnsock nsi_new2(): nsi_new (IOD #1)
libnsock nsock_connect_tcp(): TCP connection requested to 127.0.0.1:25 (IOD #1) EID 8
libnsock nsock_trace_handler_callback(): Callback: CONNECT SUCCESS for EID 8 [127.0.0.1:25]
Ncat: Connected to 127.0.0.1:25.
libnsock nsi_new2(): nsi_new (IOD #2)
libnsock nsock_read(): Read request from IOD #1 [127.0.0.1:25] (timeout: -1ms) EID 18
libnsock nsock_readbytes(): Read request for 0 bytes from IOD #2 [peer unspecified] EID 26
libnsock nsock_trace_handler_callback(): Callback: READ SUCCESS for EID 18 [127.0.0.1:25] (41 bytes): 220
centos7.fenestros.loc ESMTP Postfix..
220 centos7.fenestros.loc ESMTP Postfix
libnsock nsock_readbytes(): Read request for 0 bytes from IOD #1 [127.0.0.1:25] EID 34
```

```
^C
```

Important - Notez que **netcat** se connecte au port 25 qui est ouvert.

LAB #5 - Utilisation de tcpdump

Le logiciel **tcpdump** sert à écouter le réseau en interceptant les paquets.

Les options de cette commande sont :

```
[root@centos7 ~]# tcpdump --help
tcpdump version 4.9.2
libpcap version 1.5.3
OpenSSL 1.0.2k-fips  26 Jan 2017
Usage: tcpdump [-aAbdDefhHIJKlLnNOpqStuUvxX#] [ -B size ] [ -c count ]
        [ -C file_size ] [ -E algo:secret ] [ -F file ] [ -G seconds ]
        [ -i interface ] [ -j tstamptype ] [ -M secret ] [ --number ]
        [ -Q|-P in|out|inout ]
        [ -r file ] [ -s snaplen ] [ --time-stamp-precision precision ]
        [ --immediate-mode ] [ -T type ] [ --version ] [ -V file ]
        [ -w file ] [ -W filecount ] [ -y datalinktype ] [ -z postrotate-command ]
        [ -Z user ] [ expression ]
```

5.1 - Utilisation

L'option -i

Pour écouter sur une **interface spécifique**, utilisez l'option **-i** :

```
[root@centos7 ~]# tcpdump -i enp0s3
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp0s3, link-type EN10MB (Ethernet), capture size 262144 bytes
01:32:57.800710 IP centos7.fenestros.loc.ssh > gateway.36360: Flags [P.], seq 2207704927:2207705115, ack
23445380, win 40096, length 188
01:32:57.801785 IP gateway.36360 > centos7.fenestros.loc.ssh: Flags [.], ack 188, win 65535, length 0
01:32:57.805318 IP centos7.fenestros.loc.45319 > google-public-dns-a.google.com.domain: 62187+ PTR? 2.2.0.10.in-
addr.arpa. (39)
01:32:57.862866 IP google-public-dns-a.google.com.domain > centos7.fenestros.loc.45319: 62187 NXDomain 0/0/0 (39)
01:32:57.873506 IP centos7.fenestros.loc.54658 > google-public-dns-a.google.com.domain: 6854+ PTR? 15.2.0.10.in-
addr.arpa. (40)
01:32:57.934593 IP google-public-dns-a.google.com.domain > centos7.fenestros.loc.54658: 6854 NXDomain 0/0/0 (40)
01:32:57.947943 IP centos7.fenestros.loc.53477 > google-public-dns-a.google.com.domain: 63054+ PTR? 8.8.8.8.in-
addr.arpa. (38)
01:32:57.948649 IP centos7.fenestros.loc.ssh > gateway.36360: Flags [P.], seq 188:464, ack 1, win 40096, length
276
01:32:57.958724 IP gateway.36360 > centos7.fenestros.loc.ssh: Flags [.], ack 464, win 65535, length 0
01:32:58.025768 IP google-public-dns-a.google.com.domain > centos7.fenestros.loc.53477: 63054 1/0/0 PTR google-
public-dns-a.google.com. (82)
01:32:58.026959 IP centos7.fenestros.loc.ssh > gateway.36360: Flags [P.], seq 464:1476, ack 1, win 40096, length
1012
01:32:58.027640 IP centos7.fenestros.loc.ssh > gateway.36360: Flags [P.], seq 1476:1632, ack 1, win 40096, length
156
01:32:58.028108 IP gateway.36360 > centos7.fenestros.loc.ssh: Flags [.], ack 1476, win 65535, length 0
01:32:58.028146 IP gateway.36360 > centos7.fenestros.loc.ssh: Flags [.], ack 1632, win 65535, length 0
01:32:58.028313 IP centos7.fenestros.loc.ssh > gateway.36360: Flags [P.], seq 1632:1788, ack 1, win 40096, length
156
01:32:58.028822 IP centos7.fenestros.loc.ssh > gateway.36360: Flags [P.], seq 1788:2152, ack 1, win 40096, length
364
01:32:58.029240 IP gateway.36360 > centos7.fenestros.loc.ssh: Flags [.], ack 1788, win 65535, length 0
01:32:58.029273 IP gateway.36360 > centos7.fenestros.loc.ssh: Flags [.], ack 2152, win 65535, length 0
01:32:58.029710 IP centos7.fenestros.loc.ssh > gateway.36360: Flags [P.], seq 2152:2516, ack 1, win 40096, length
```

```
364
01:32:58.030217 IP gateway.36360 > centos7.fenestros.loc.ssh: Flags [.], ack 2516, win 65535, length 0
01:32:58.030773 IP centos7.fenestros.loc.ssh > gateway.36360: Flags [P.], seq 2516:2776, ack 1, win 40096, length
260
01:32:58.034485 IP gateway.36360 > centos7.fenestros.loc.ssh: Flags [.], ack 2776, win 65535, length 0
...
```

Notez qu'à la fin, un résumé vous est présenté, par exemple :

```
...
^C
767 packets captured
767 packets received by filter
0 packets dropped by kernel
```

L'option -x

Pour écouter sur une interface spécifique et voir le contenu en Hexadécimal, utilisez les options -i et **-x** :

```
[root@centos7 ~]# tcpdump -i enp0s3 -x
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp0s3, link-type EN10MB (Ethernet), capture size 262144 bytes
01:34:55.540011 IP centos7.fenestros.loc.ssh > gateway.36360: Flags [P.], seq 2208166651:2208166839, ack
23445916, win 40096, length 188
    0x0000:  4510 00e4 851c 4000 4006 9cd7 0a00 020f
    0x0010:  0a00 0202 0016 8e08 839d f2fb 0165 c19c
    0x0020:  5018 9ca0 18e7 0000 96d1 7d89 0022 fb31
    0x0030:  dec8 d6f4 1227 ceb3 4df9 1ed8 691b b0e4
    0x0040:  4561 6454 7f80 8928 1704 99a9 d4fe e565
    0x0050:  7d2e e55f eadc 7e0e 352c 2f65 dd2a ff02
    0x0060:  3a66 d1fa 3a15 9a4e 0054 91fe 7fe5 ce35
    0x0070:  7df4 371a 6363 1302 4037 2a7f dde1 3d76
```

```
    0x0080:  6ce5 069a 5855 cff6 18f5 dcf3 afff e525
    0x0090:  ec03 c13d c23c 9d4b e0b9 0661 4f59 cd11
    0x00a0:  cffb 1447 d6e6 7ab7 d7a8 d357 8fa6 ac7a
    0x00b0:  bcf5 257f 41dd 5064 ba0b 189b 1563 06fa
    0x00c0:  5364 9d6a 4f7c e99c 7a27 c3fc 3a2e 2618
    0x00d0:  1357 7e4c b62a 9e44 8350 71fa e9b8 a17f
    0x00e0:  a2be c731
01:34:55.540618 IP gateway.36360 > centos7.fenestros.loc.ssh: Flags [.], ack 188, win 65535, length 0
    0x0000:  4500 0028 061f 0000 4006 5ca1 0a00 0202
    0x0010:  0a00 020f 8e08 0016 0165 c19c 839d f3b7
    0x0020:  5010 ffff cf4e 0000 0000 0000 0000
^C01:34:55.543115 IP centos7.fenestros.loc.47568 > google-public-dns-a.google.com.domain: 37698+ PTR?
2.2.0.10.in-addr.arpa. (39)
    0x0000:  4500 0043 8ca3 4000 4011 91e8 0a00 020f
    0x0010:  0808 0808 b9d0 0035 002f 1c5f 9342 0100
    0x0020:  0001 0000 0000 0000 0132 0132 0130 0231
    0x0030:  3007 696e 2d61 6464 7204 6172 7061 0000
    0x0040:  0c00 01

3 packets captured
21 packets received by filter
0 packets dropped by kernel
```

L'option -X

Pour écouter sur une interface spécifique et voir le contenu en Hexadécimal et en ASCII, utilisez les options -i et **-X** :

```
[root@centos7 ~]# tcpdump -i enp0s3 -X
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp0s3, link-type EN10MB (Ethernet), capture size 262144 bytes
01:35:56.671522 IP centos7.fenestros.loc.ssh > gateway.36360: Flags [P.], seq 2208168643:2208168831, ack
23446048, win 40096, length 188
    0x0000:  4510 00e4 8522 4000 4006 9cd1 0a00 020f  E....."@.@.......
```

```
    0x0010:  0a00 0202 0016 8e08 839d fac3 0165 c220  ............e..
    0x0020:  5018 9ca0 18e7 0000 d8cc 9fe4 3055 351c  P...........0U5.
    0x0030:  96a0 b2c0 650e 44e7 0016 b72e c990 4929  ....e.D.......I)
    0x0040:  0d05 e79b ef5b ccef aafc 607b b9a2 a714  .....[....`{....
    0x0050:  e1a0 36a1 b5f3 4a0a 5cdd 90bd 96cf b75d  ..6...J.\......]
    0x0060:  efa5 cc0b d195 d69a e933 48f2 74a5 ca6b  .........3H.t..k
    0x0070:  c803 fe4e 5832 2253 d6d9 178b c63a df8e  ...NX2"S.....:..
    0x0080:  62a3 6e67 91f8 cc50 f330 51da a285 1c3a  b.ng...P.0Q....:
    0x0090:  b074 a98f 95d6 f528 03e2 308a 31e8 5aa2  .t.....(..0.1.Z.
    0x00a0:  988b e3b1 444e ab24 e1b0 3eb8 ffa7 ae3f  ....DN.$..>....?
    0x00b0:  332f 221d 3bec d17a 1c8f 741a eaaf 5444  3/".;..z..t...TD
    0x00c0:  ea38 320e ea7f e9d2 035a 2531 e8f1 0757  .82......Z%1...W
    0x00d0:  0259 70b1 464b 51ea 9e6b 1b93 48ea 6f25  .Yp.FKQ..k..H.o%
    0x00e0:  d918 9027                                ...'
01:35:56.672680 IP gateway.36360 > centos7.fenestros.loc.ssh: Flags [.], ack 188, win 65535, length 0
    0x0000:  4500 0028 06cc 0000 4006 5bf4 0a00 0202  E..(....@.[.....
    0x0010:  0a00 020f 8e08 0016 0165 c220 839d fb7f  .........e......
    0x0020:  5010 ffff c702 0000 0000 0000 0000       P.............
^C01:35:56.674310 IP centos7.fenestros.loc.57986 > google-public-dns-a.google.com.domain: 33529+ PTR?
2.2.0.10.in-addr.arpa. (39)
    0x0000:  4500 0043 188e 4000 4011 05fe 0a00 020f  E..C..@.@.......
    0x0010:  0808 0808 e282 0035 002f 1c5f 82f9 0100  .......5./._....
    0x0020:  0001 0000 0000 0000 0132 0132 0130 0231  .........2.2.0.1
    0x0030:  3007 696e 2d61 6464 7204 6172 7061 0000  0.in-addr.arpa..
    0x0040:  0c00 01                                  ...

3 packets captured
13 packets received by filter
0 packets dropped by kernel
```

L'option -w

Pour écouter sur une interface spécifique et envoyer la sortie dans un fichier, utilisez les options -i et **-w** :

```
[root@centos7 ~]# tcpdump -i enp0s3 -w log.dump
tcpdump: listening on enp0s3, link-type EN10MB (Ethernet), capture size 262144 bytes
^C45 packets captured
45 packets received by filter
0 packets dropped by kernel

[root@centos7 ~]# ls -l log.dump
-rw-r--r--. 1 tcpdump tcpdump 8685 Jun  9 01:37 log.dump
```

Important - Pour générer le trafic, réactualisez simplement cette page web. Arrêtez la sortie de la commande à l'aide des touches **^C**.

Notez que le fichier log.dump est au format **libpcap** et non au format texte. Il est donc inutile d'essayer de lire son contenu :

```
[root@centos7 ~]# file log.dump
log.dump: tcpdump capture file (little-endian) - version 2.4 (Ethernet, capture length 262144)
```

L'option -v

Tcpdump peut être utilisé avec un de trois modes verbose.

Mode	Option
Light verbose	-v
Medium verbose	-vv
Full verbose	-vvv

```
[root@centos7 ~]# tcpdump -i enp0s3 -v
tcpdump: listening on enp0s3, link-type EN10MB (Ethernet), capture size 262144 bytes
01:39:13.094781 IP (tos 0x10, ttl 64, id 39762, offset 0, flags [DF], proto TCP (6), length 164)
```

```
^C    centos7.fenestros.loc.ssh > gateway.36360: Flags [P.], cksum 0x18a7 (incorrect -> 0xd270), seq
2210981047:2210981171, ack 23448932, win 40096, length 124

1 packet captured
11 packets received by filter
0 packets dropped by kernel
```

5.2 - Filtrage à l'écoute

Tcpdump peut effectuer du filtrage lors de l'écoute.

Pour uniquement écouter les paquets en provenance de l'adresse IP 192.168.1.11, utilisez la condition **src host** :

```
# tcpdump src host 192.168.1.11 [Entrée]
```

Pour uniquement écouter les paquets en provenance de l'adresse IP 192.168.1.11 et vers l'adresse 192.168.1.2, utilisez les conditions src host et **dst host** :

```
# tcpdump src host 192.168.1.11 and dst host 192.168.1.2 [Entrée]
```

Pour uniquement écouter les paquets d'un port précis, utilisez la condition **port** :

```
# tcpdump -i eth0 port 80 [Entrée]
```

Pour uniquement écouter les paquets d'un protocole précis, utilisez une condition telle **ip**, **icmp**, **arp**, **rarp**, **udp** ou **tcp**:

```
# tcpdump -i eth0 udp [Entrée]
```

Pour uniquement écouter les paquets d'une taille inférieure à 100 octets, utilisez la condition **less** :

```
# tcpdump -i eth0 less 100 [Entrée]
```

Pour uniquement écouter les paquets d'une taille supérieure à 100 octets, utilisez la condition **great** :

```
# tcpdump -i eth0 greater 100 [Entrée]
```

L'utilisation des ses options et conditions peut être combinée pour donner des commandes telles :

```
# tcpdump -i eth0 -X src host 192.168.1.11 and dst host 192.168.1.2 and port 21 and ftp [Entrée]
```

LAB #6 - Mise en place d'un VPN avec OpenVPN

6.1 - Présentation

OpenVPN permet à des pairs de s'authentifier entre eux à l'aide :

- d'une **clé privée partagée** à l'avance,
- de **certificats** ou,
- à partir de la version 2.0 et à condition que le serveur possède un certificat, de **couples de noms d'utilisateur/mot de passe** sans besoin d'un certificat client

OpenVPN :

- utilise de manière intensive la bibliothèque d'authentification **OpenSSL** ainsi que le protocole **SSLv3/TLSv1**,
- n'est pas compatible avec IPsec ou d'autres logiciels VPN.

6.2 - Configuration commune au client et au serveur

Commencez par vérifiez si le paquet **openssl** est bien installé :

```
[root@centos7 ~]# rpm -q openssl
openssl-1.0.2k-8.el7.x86_64
```

Installez ensuite le paquet openvpn :

```
[root@centos7 ~]# yum install openvpn
```

Naviguez au répertoire **/etc/openvpn** et créez la clef partagée :

```
[root@centos7 ~]# cd /etc/openvpn/
[root@centos7 openvpn]# openvpn --genkey --secret static.key
[root@centos7 openvpn]# cat static.key
#
# 2048 bit OpenVPN static key
#
-----BEGIN OpenVPN Static key V1-----
54f96ea50dbef7d5341efeda459b05ad
5af134bf915bbd867fdd6310f4f0b72b
331a82cdc6080622a7861e8c30cd0ffb
6b35c143e5c715077247270bdb610fc8
4c536f34742ba23f2bfe9ab148b3fa04
20d1f6e5a20d58db30cce56ce1ca5744
3028353a7e5e47b3f630738b71b04a1e
e388b5e986826ce481ff457157b3492e
61c147cd3d4373e283ad91c8ac44c0e8
3b593d342cd0a2600db7b3e7cd0efa89
d38dd861c1e4fc566e5e50004b102c7f
b444795e2691cd59dfbb51e79996339d
7e54d002aa4d5c63b3c155fbcc20f696
fe148128f2e94e509c39c72c117a684b
9fa8c7e159c451a7c52f42b2260d62c9
586d66a454319ba538559c143643e434
-----END OpenVPN Static key V1-----
```

L'architecture réseau sera donc la suivante :

```
                                    serveur <------------Votre réseau-------------> client
```

```
                                    |                                   |
                                    |                                   |
                                eth0@ifxxx                          eth0@ifxxx
                                172.yy.0.3                          172.yy.0.4
```

6.3 - Configuration du client

Créez le fichier **/etc/openvpn/client.conf** :

```
[root@centos7 ~]# vi /etc/openvpn/client.conf
[root@centos7 ~]# cat /etc/openvpn/client.conf
remote 10.0.2.15
dev tun
port 1194
proto udp
comp-lzo
ifconfig 10.0.0.2 10.0.0.1
secret /etc/openvpn/static.key
```

Important - Trouvez la signification de chacune des directives dans ce fichier.

Arrêtez le service **firewalld** :

```
[root@centos7 ~]# systemctl stop firewalld
```

Lancez openvpn en ligne de commande et en arrière plan en spécifiant une journalisation :

```
[root@centos7 ~]# openvpn --config /etc/openvpn/client.conf > /var/log/vpn 2>&1 &
```

Vérifiez ensuite que le **socket** d'openvpn soit ouvert :

```
[root@centos7 ~]# netstat -an | grep 1194
udp        0      0 0.0.0.0:1194                0.0.0.0:*
```

Constatez ensuite la table de routage :

```
[root@centos7 ~]# netstat -ar
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Fenêtre irtt Iface
default         gateway         0.0.0.0         UG        0 0          0 enp0s3
10.0.0.1        0.0.0.0         255.255.255.255 UH        0 0          0 tun0
10.0.2.0        0.0.0.0         255.255.255.0   U         0 0          0 enp0s3
```

Notez la présence de la route via **tun0**.

Constatez ensuite le montage du tunnel en regardant le contenu du fichier de journalisation **/var/log/vpn** :

```
[root@centos7 ~]# tail /var/log/vpn
```

6.4 - Configuration du serveur

Créez le fichier **/etc/openvpn/server.conf** :

```
[root@centos7 ~]# vi /etc/openvpn/server.conf
[root@centos7 ~]# cat /etc/openvpn/server.conf
dev tun
ifconfig 10.0.0.1 10.0.0.2
secret /etc/openvpn/static.key
port 1194
proto udp
user nobody
```

```
group nobody
daemon
comp-lzo
keepalive 10 60
ping-timer-rem
persist-tun
persist-key
log /var/log/vpn
verb 1
```

Important - Trouvez la signification de chacune des directives dans ce fichier.

Arrêtez le service **firewalld** :

```
[root@centos7 ~]# systemctl stop firewalld
```

Lancez openvpn en ligne de commande et en arrière plan en spécifiant une journalisation :

```
[root@centos7 ~]# openvpn --config /etc/openvpn/server.conf > /var/log/vpn 2>&1 &
[1] 7751
```

Vérifiez ensuite que le **socket** d'openvpn soit ouvert :

```
[root@centos7 ~]# netstat -an | grep 1194
udp        0      0 0.0.0.0:1194                0.0.0.0:*
```

Constatez ensuite la table de routage :

```
[root@centos7 ~]# netstat -ar
Kernel IP routing table
```

```
Destination     Gateway         Genmask         Flags   MSS Fenêtre irtt Iface
0.0.0.0         10.0.2.2        0.0.0.0         UG        0 0          0 enp0s3
10.0.0.1        0.0.0.0         255.255.255.255 UH        0 0          0 tun0
10.0.2.0        0.0.0.0         255.255.255.0   U         0 0          0 enp0s3
```

Constatez ensuite le montage du tunnel en regardant le contenu du fichier de journalisation **/var/log/vpn** :

```
[root@centos7 ~]# tail /var/log/vpn
```

6.5 - Tests

Du client vers le serveur

Sur le client, utilisez la commande ping pour envoyer des paquets dans le tunnel :

```
[root@centos7 ~]# ping -c3 10.0.0.1
PING 10.0.0.1 (10.0.0.1) 56(84) bytes of data.
64 bytes from 10.0.0.1: icmp_seq=1 ttl=64 time=7.62 ms
64 bytes from 10.0.0.1: icmp_seq=2 ttl=64 time=1.35 ms
64 bytes from 10.0.0.1: icmp_seq=3 ttl=64 time=0.000 ms

--- 10.0.0.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2047ms
rtt min/avg/max/mdev = 0.000/2.994/7.629/3.323 ms
```

Du serveur vers le client

Sur le serveur, utilisez la commande ping pour envoyer des paquets dans le tunnel :

```
[root@centos7 ~]# ping -c5 10.0.0.2
```

```
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=2.59 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=9.08 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=7.24 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=7.03 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=4.08 ms

--- 10.0.0.2 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4034ms
rtt min/avg/max/mdev = 2.597/6.008/9.084/2.340 ms
```

Annexe #1 - Comprendre les Réseaux

Présentation des Réseaux

La définition d'un réseau peut être résumé ainsi :

- un ensemble d'**Equipements** (systèmes et périphériques) communiquant entre eux,
- une entité destinée au transport de données dans différents environnements.

Pour que la communication soit efficace, elle doit respecter les critères suivants :

- présenter des informations compréhensibles par tous les participants,
- être compatible avec un maximum d'interlocuteurs différents (dans le cas d'un réseau, les interlocuteurs sont des équipements : imprimantes, ordinateurs, clients, serveurs, téléphones...),
- si l'interlocuteur n'est pas disponible, les informations ne doivent pas se perdre,
- permettre une réduction des coûts (par ex. interconnexion à bas coût),
- permettre une productivité accrue (par ex. interconnexion à haut débit),
- être sécurisée si les informations à transmettre sont dites sensibles,
- garantir l'**unicité** et de l'**universalité** de l'**accès à l'information**.

On peut distinguer deux familles d'**Equipements** - les **Eléments Passifs** et les **Eléments Actifs**.

Les **Eléments Passifs** transmettent le signal d'un point à un autre :

- **Les Infrastructures ou Supports** - des câbles, de l'atmosphère ou des fibres optiques permettant de relier **physiquement** des équipements,
- **La Topologie** - l'architecture d'un réseau définissant les connexions entre les **Equipements** et, éventuellement, la hiérarchie entre eux.

Les **Eléments Actifs** sont des équipements qui consomment de l'énergie en traitant ou en interprétant le signal. Les **Equipements** sont classés selon leurs fonctions :

- **Equipement de Distribution Interne au Réseau** - Répartiteur (Hub, Switch, Commutateur etc.), Borne d'accès (Hotspot), Convertisseur de signal (Transciever), Amplificateur (Répéteur) ...,
- **Equipement d'Interconnexion de Réseaux** - Routeurs, Ponts ...,
- **Nœuds** et **Interfaces Réseaux** - postes informatiques, équipements en réseau ....

Un **Nœud** est une extrémité de connexion qui peut être une intersection de plusieurs connexions ou de plusieurs **Equipements**.

Une **Interface Réseau** est une prise ou élément d'un **Equipement Actif** faisant la connexion vers d'autres **Equipements** réseaux et qui reçoit et émet des données.

> Dans le cas d'un mélange d'**Equipements** non-homogènes en termes de performances au sein du même réseau, c'est la loi du plus faible qui emporte.

Tous les **Equipements** connectés au même support doivent respecter un ensemble de règles appelé une **Protocole de Communication**.

Les **Protocoles de Communication** définissent de façon formelle et interopérable la manière dont les informations sont échangées entre les **Equipements**.

Des **Logiciels**, dédiés à la gestion de ces **Protocoles de Communication**, sont installés sur des **Equipements d'Interconnexion** afin de fournir des fonctions de contrôle permettant une communication entre les **Equipements**.

Se basant sur des **Protocoles de Communication**, des **Services** fournissent des fonctionnalités accessibles aux utilisateurs ou d'autres programmes.

L'ensemble des **Equipements**, **Logiciels** et **Protocoles de Communication** constitue l'**Architecture Réseau**.

Classification des Réseaux

Les réseaux peuvent être classifiés de trois façon différentes :

- par **Mode de Transmission**,
- par **Topologie**,
- par **Étendue**.

Classification par Mode de Transmission

Il existe deux **Classes** de réseaux dans cette classification :

- les **Réseaux en Mode de Diffusion**,
  - utilise un seul support de transmission,
  - le message est envoyé sur tout le réseau à l'adresse d'**un** destinataire,
- les **Réseaux en Mode Point à Point**,
  - une seule liaison entre deux équipements,
  - les nœuds permettent de choisir la route en fonction de l'adresse du destinataire,
  - quand deux nœuds non directement connectés entre eux veulent communiquer ils le font par l'intermédiaire des autres noeuds du réseau.

Classification par Topologie

> La **Topologie Physique** d'un réseau décrit l'organisation de ce dernier en termes de câblage. La **Topologie Logique** d'un réseau décrit comment les données circulent sur le réseau. En effet c'est le choix des concentrateurs ainsi que les connections des câbles qui déterminent la topologie logique.

La Topologie Physique

Il existe 6 topologies physiques de réseau :

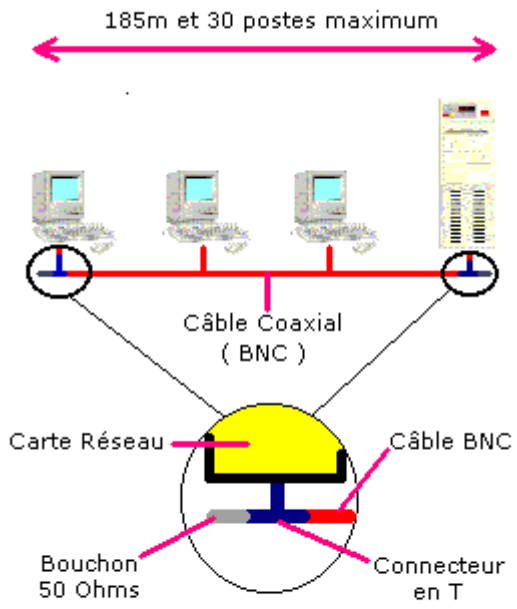
- La Topologie en Ligne,
- La Topologie en Bus,
- La Topologie en Etoile,
- La Topologie en Anneau,
- La Topologie en Arbre,
- La Topologie Maillée.

La Topologie en Ligne

Tous les nœuds sont connectés à un seul support. L'inconvénient de cette topologie est que dans le cas d'une défaillance d'une station, le réseau se trouve coupé en deux sous-réseaux.

La Topologie en Bus

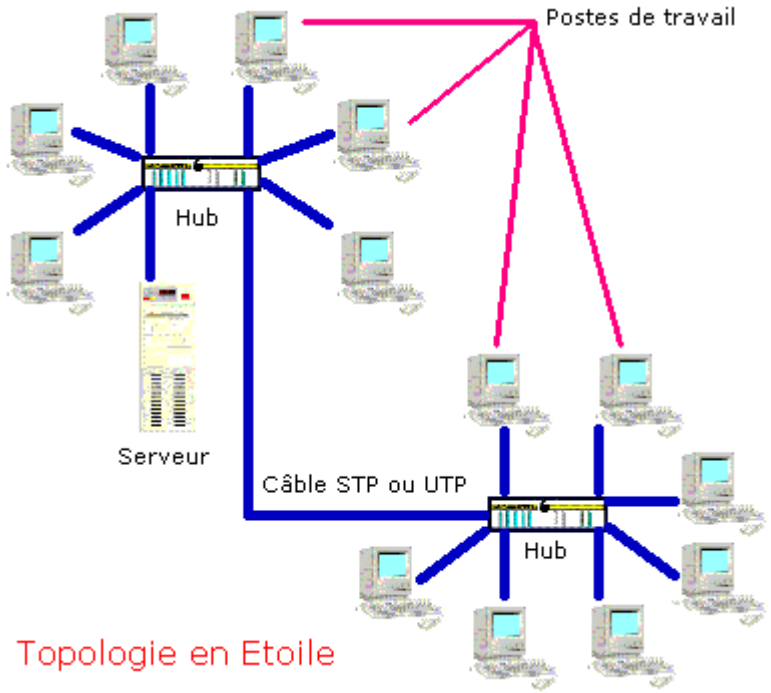
Tous les nœuds sont connectés à un seul support (un câble BNC en T) avec des bouchons à chaque extrémité. La longueur du bus est limitée à **185m**. Le nombre de stations de travail est limité à **30**. Les Stations sont reliées au Bus par des 'T'. Les bouchons sont des terminateurs qui sont des résistances de **50 Ohms**. Quand le support tombe en panne, le réseau ne fonctionne plus. Quand une station tombe en panne, elle ne perturbe pas le fonctionnement de l'ensemble du réseau. Les Stations étant reliés à un suel support, ce type de topologie necessite un **Protocole d'Accès** pour gérer le tour de parole des Stations afin d'éviter des conflits.



Topologie en Bus

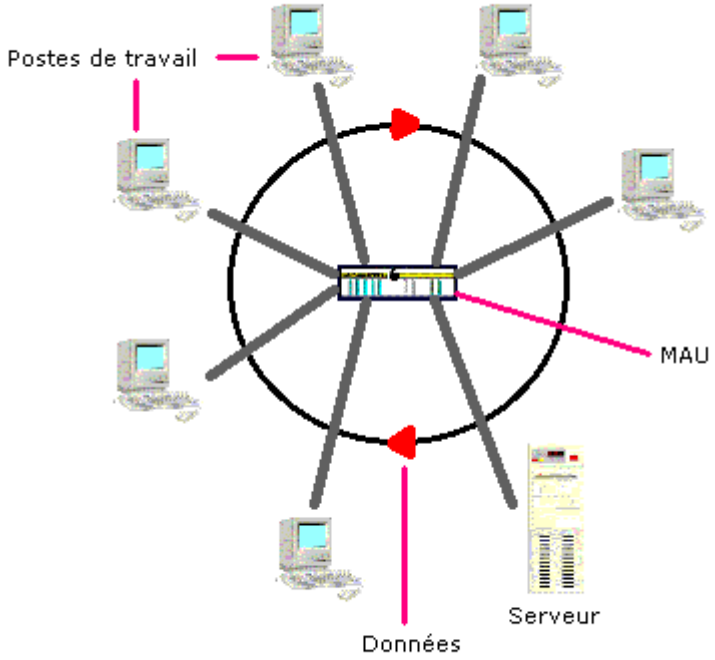
La Topologie en Étoile

Chaque nœud est connecté à un périphérique central appelé un **Hub** (**Concentrateur**) ou un **Switch** (**Commutateur**). Un Hub ou un Switch est prévu pour 4, 8, 16, 32 ... stations. En cas d'un réseau d'un plus grand nombre de stations, plusieurs Hubs ou Switches sont connectés ensemble. Quand une station tombe en panne, elle ne perturbe pas le fonctionnement de l'ensemble du réseau. Le point faible de cette topologie est l'équipement central.



La Topologie en Anneau

Chaque nœud est relié directement à ses deux voisins dans une topologie logique de cercle ininterrompu et une topologie physique en étoile car les stations sont reliées à un type de hub spécial, appelé un **Multistation Access Unit** (MAU).



Topologie en Anneau

Les stations sont reliées à la MAU par un câble 'IBM' munie d'une prise **AUI** du côté de la carte et une prise **Hermaphrodite** du coté de la MAU. Les données sont échangées dans un sens unidirectionnel. Une trame, appelée un **jeton**, circule en permanence. Si l'anneau est brisé, l'ensemble du réseau s'arrête. Pour cette raison, il est courant de voir deux anneaux contre-rotatifs.

La Topologie en Arbre

La Topologie en Arbre est utilisée dans un réseau hierarchique où le sommet, aussi appelé la **racine**, est connecté à plusieurs noeuds de niveau inférieur. Ces neouds peuvent à leur tour être connectés à d'autres noeuds inférieurs. L'ensemble forme une arborescence. Le point faible de cette topologie est sa racine. En cas de défaillance, le réseau est coupé en deux.

La Topologie Maillée

Cette Topologie est utilisée pour des grands réseaux de distribution tels Internet ou le WIFI. Chaque noeud à tous les autres via des liaisons point à point. Le nombre de liaisons devient très rapidement important en cas d'un grand nombre de noeuds. Par exemple dans le cas de 100 Stations (N), le nombre de liaisons est obtenu par la formule suivante :

```
N(N-1)/2 = 100(100-1)/2 = 4 950
```

La **Topologie Physique** la plus répandue est la **Topologie en Etoile**.

Classification par Etendue

La classification par étendue nous fournit 4 réseaux principaux :

Nom	Description	Traduction	Taille Approximative (M)
PAN	Personal Area Network	Réseau Personnel	1 -10
LAN	Local Area Network	Réseau Local Entreprise (RLE)	5 - 1 200
MAN	Métropolitain Area Network	Réseau Urbain	900 - 100 000
WAN	Wide Area Network	Réseau Long Distance (RLD)	50 000 et au delà

Cependant, d'autres classification existent :

CAN	Campus Area Network	Réseau de Campus
GAN	Global Area Network	Réseau Global
TAN	Tiny Area Network	Réseau Minuscule
FAN	Family Area Network	Réseau Familial
SAN	Storage Area Network	Réseau de Stockage

Etant donné que les WANs sont gérés par des opérateurs de télécommunications qui doivent demander une licence à l'état mais que les LANs ont été historiquement mis en oeuvre dans les entreprises, ces derniers sont en majorité issus du monde informatique.

Les Types de LAN

Il existe deux types de LAN :

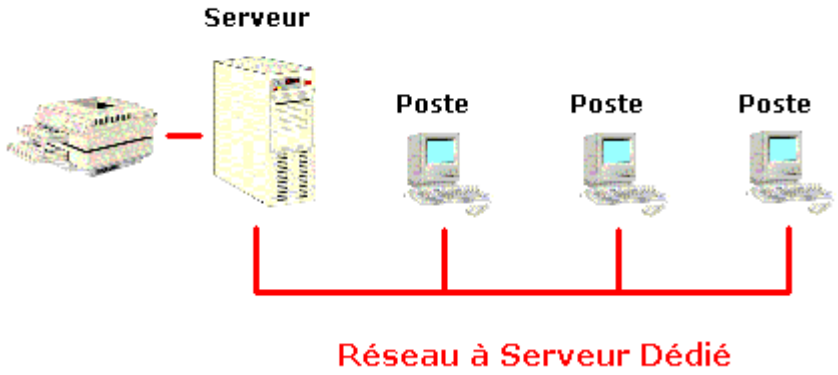
- le réseau à serveur dédié,
- le réseau poste à poste.

Réseau à Serveur Dédié

Le réseau à serveur dédié est caractérisé par le fait que toutes les ressources (imprimantes, applications, lecteurs etc.) sont gérées par le serveur. Les autres micro-ordinateurs ne jouent le rôle de client.

Des exemples des systèmes d'exploitation du réseau à serveur dédié sont :

- Windows NT Server,
- Windows 2000 Server,
- Windows 2003 Server,
- Windows 2008 Server,
- Linux,
- Unix.



Réseau Poste-à-Poste

Le réseau poste à poste est caractérisé par le fait que tous les ordinateurs peuvent jouer le rôle de client et de serveur :

- Windows 95,
- Windows 98,
- Windows NT Workstation.

Le Modèle Client/Serveur

Le modèle Client/Serveur est une des modalités des architectures informatiques distribuées. Dans ce modèle un serveur est tout **Logiciel** fournissant un **Service**.

Le serveur est aussi :

- passif, c'est-à-dire en attente permenante d'une demande, appelée une requête d'un client,
- capable de traiter plusieurs requêtes simultanément en utilisant le **multi-threading**,
- garant de l'intégrité globale.

Le client est, par contre **actif**, étant à l'origine des requêtes.

Il existe trois types de modèle client/serveur :

- **Plat** - tous les clients communiques avec un seul serveur,
- **Hiérarchique** - les clients n'ont de contact qu'avec les serveurs de plus haut niveau qu'eux,
- **Peer-to-Peer** - les équipements sont à la fois client **et** serveur en même temps.

Modèles de Communication

Les réseaux sont bâtis sur des technologies et des modèles. Le modèle **théorique** le plus important est le modèle **O**pen **S**ystem **Interconnection** créé par l'**I**nternational **Organization** for **S**tandardization tandis que le modèle pratique le plus important est le modèle **TCP/IP**.

Le modèle OSI

Le modèle OSI qui a été proposé par l'ISO est devenu le standard en termes de modèle pour décrire l'échange de données entre ordinateurs. Cette norme se repose sur sept couches, de la une - la Couche Physique, à la sept - la Couche d'Application, appelés des services. La communication entre les différentes couches est synchronisée entre le poste émetteur et le poste récepteur grâce à ce que l'on appelle un protocole.

Ce modèle repose sur trois termes :

- Les **Couches**,
- Les **Protocoles**,
- Les **Interfaces**.

Les Couches

Des sept couches :

- Les couches 1 à 3 sont les **Couches Basses** orientées **Transmission**,
- La couche 4 est la **Couche Charnière** entre les **Couches Basses** et les **Couches Hautes**,

- Les couches 5 à 7 sont les **Couches Hautes** orientées **Traitement**.

La couche du même niveau du système **A** parle avec son homologue du système **B**.

- **La Couche Physique** (Couche 1) est responsable :
  - du transfert de données binaires sur le câble physique ou virtuel
  - de la définition de tout aspect physique allant du connecteur jusqu'au câble en passant par la carte réseau, y compris l'organisation même du réseau
  - de la définition des tensions électriques sur le câble pour obtenir le 0 et le 1 binaires
- **La Couche de Liaison** (Couche 2) est responsable :
  - de la réception des données de la couche physique
  - de l'organisation des données en fragments, appelés des trames qui ont un format différent selon s'il s'agit d'un réseau basé sur la technologie Ethernet ou la technologie Token-Ring
  - de la préparation, émission et réception des trames
  - de la gestion de l'accès au réseau
  - de la communication nœud à nœud
  - de la gestion des erreurs
    - avant la transmission, le nœud émetteur calcule un code appelé un CRC et l'incorpore dans les données envoyées
    - le nœud récepteur recalcule un CRC en fonction du contenu de la trame reçue et le compare à celui incorporé avec l'envoi
    - en cas de deux CRC identique, le nœud récepteur envoie un accusé de réception au nœud émetteur
  - de la réception de l'accusé de réception
  - éventuellement de le ré-émission des données
  - En prenant ce modèle, l'IEEE (Institute of Electrical and Eletronics Engineers) l'a étendu avec le Modèle IEEE (802).
    - Dans ce modèle la Couche de Liaison est divisée en deux sous-couches importantes :
      - La **Sous-Couche LLC** (Logical Link Control) qui :
        - gère les accusés de réception
        - gère le flux de trames
      - La **Sous-Couche MAC** (Media Access Control) qui :
        - gère la méthode d'accès au réseau
        - le CSMA/CD dans un réseau basé sur la technologie Ethernet
        - l'accès au jeton dans un réseau basé sur la technologie Token-Ring
        - gère les erreurs
- **La Couche de Réseau** (Couche 3) est responsable de la gestion de la bonne distribution des différentes informations aux bonnes adresses en :

    - identifiant le chemin à emprunter d'un nœud donné à un autre
    - appliquant une conversion des adresses logiques (des noms) en adresses physiques
    - ajoutant des information adressage aux envois
    - détectant des paquets trop volumineux avant l'envoi et en les divisant en trames de données de tailles autorisées

- **La Couche de Transport** (Couche 4) est responsable de veiller à ce que les données soient envoyées correctement en :
    - constituant des paquets de données corrects
    - les envoyant dans le bon ordre
    - vérifiant que les données sont traités dans le même ordre que l'ordre d'émission
    - permettant à un processus sur un nœud de communiquer avec un autre nœud et d'échanger des messages avec lui

- **La Couche de Session** (Couche 5) est responsable :
    - de l'établissement, du maintien, et de la mise à fin de la communication entre deux noeuds distants, c'est-à-dire, de la session
    - de la conversation entre deux processus de vérification de la réception des messages envoyés en séquences, c'est-à-dire, le point de contrôle

- de la sécurité lors de l'ouverture de la session, c'est-à-dire, les droits d'utilisateurs etc.

- **La Couche de Présentation** (Couche 6) est responsable :
    - du formatage et de la mise en forme des données
    - des conversions de données telles le cryptage/décryptage

- **La Couche d'Application** (Couche 7) est responsable :
    - du dialogue homme/machine via des messages affichés
    - du partage des ressources
    - de la messagerie

Les Protocoles

Un **protocole** est un langage commun utilisé par dexu entités en communication pour pouvoir se comprendre. La nature du Protocole dépends directement de la nature de la communication. Cette bature dépend du **paradigme** de communication que l'application nécessite. Le paradigme est un modèle abstrait d'un problème ou d'une situation. Dans le paradigme de la diffusion, l'émetteur envoie dans informations au récepteur sans se soucier de ce que le récepteur va en faire. C'est la responsabilité du récepteur de comprendre et d'utiliser les informations.

Les Interfaces

Chaque couche rend des **services** à la couche immédiatement supérieure et utilise les services de la couche immédiatement inférieure. L'ensemble des services s'appelle une **Interface**. Les services sont composés de **S**ervice **D**ata **U**nits et sont disponibles par un **S**service **A**ccess **P**oint.

Protocol Data Units

L'**Unité de Données** ou *Protocol Data Unit* pour chaque couche comporte un nom spécifique :

- **Application Protocol Data Units** pour la couche **Application**,
- **Présentation Protocol Data Units** pour la couche **Présentation**,
- **Session Protocol Data Units** pour la couche **Session**,
- **Transport Protocol Data Units** pour la couche **Transport**.

Or, pour les **Couches Basses** on parle de :

- **Paquets** pour la couche **Réseau**,
- **Trames** pour la couche **Liaison**,
- **Bits** pouyr la couche **Physique**.

Encapsulation et Désencapsulation

Lorque les données sont communiqueés par le système A au système B, celles-ci commencent au niveau de la couche d'Application. Le couche d'Application ajoute une en-tête à l'unité de données qui contient des **informations de contrôle du protocole**. Au passage de chaque couche, celle-ci ajoute sa propre en-tête. De cette façon, lors de sa descente vers la couche physique, les données et l'entête de la couche supérieure sont encapulsulées :

Couche Système A	**Encapsulation**
Application	Application Header (AH) + Unité de Données (UD)
Présentation	Présentation Header (PH) + AH + UD
Session	Session Header (SH) + PH + AH + UD

Couche Système A	Encapsulation
Transport	Transport Header (TH) + SH + PH + AH + UD
Réseau	Network Header (NH) + TH + SH + PH + AH + UD
Liaison	Liaison Header (DH) + NH + TH + SH + PH + AH + UD

Lors de son voyage de la couche Physique vers la couche Application dans le système B, les en-têtes sont supprimées par chaque couche correspondante. On parle alors de **désencapsulation** :

Couche Système B	Encapsulation
Liaison	Liaison Header (DH) + NH + TH + SH + PH + AH + UD
Réseau	Network Header (NH) + TH + SH + PH + AH + UD
Transport	Transport Header (TH) + SH + PH + AH + UD
Session	Session Header (SH) + PH + AH + UD
Présentation	Présentation Header (PH) + AH + UD
Application	Application Header (AH) + Unité de Données (UD)

Spécification NDIS et le Modèle ODI

<note tip> **Cliquez ici pour ouvrir le schéma Simplifié du Modèle OSI incluant la spécification NDIS** </note>

La spécification NDIS (Network Driver Interface Specification) a été introduite conjointement par les sociétés Microsoft et 3Com. Cette spécification ainsi que son homologue, le modèle ODI (Open Datalink Interface) introduit conjointement par les sociétés Novell et Apple à la même époque, définit des standards pour les pilotes de cartes réseau afin qu'ils puissent être indépendants des protocoles utilisées et les systèmes d'exploitation sur les machines. Des deux 'standards', la spécification NDIS est le plus répandu, intervenant a niveau de la sous-couche MAC et l a couche de liaison. Elle spécifie :

- l'interface pilote-matériel
- l'interface pilote-protocole
- l'interface pilote - système d'exploitation

Le modèle TCP/IP

<note tip> **Cliquez ici pour voir le modèle OSI incluant la suite des protocoles et services TCP/IP** </note>

La suite des protocoles TCP/IP (Transmission Control Protocol / Internet Protocol) est issu de la DOD (Dept. Américain de la Défense) et le travail de l'ARPA (Advanced Research Project Agency).

- La suite des protocoles TCP/IP
  - a été introduite en 1974
  - a été utilisée dans l'ARPAnet en 1975
  - permet la communication entre des réseaux à base de systèmes d'exploitation, architectures et technologies différents
  - est très proche du modèle OSI en termes d'architecture et se place au niveau de la couche d'Application jusqu'à la couche Réseau.
  - est, en réalité, une suite de protocoles et de services :
    - **IP** (Internet Protocol)
      - le protocole IP s'intègre dans la couche Réseau du modèle OSI en assurant la communication entre les systèmes. Bien qu'il puisse découper des messages en fragments ou datagrammes et les reconstituer dans le bon ordre à l'arrivée, il ne garantit pas la réception.
    - **ICMP** (Internet Control Message Protocol)
      - le protocole ICMP produit des messages de contrôle aidant à synchroniser le réseau. Un exemple de ceci est la commande ping.
    - **TCP** (Transmission Control Protocol)
      - le protocole TCP se trouve au niveau de la couche de Transport du modèle OSI et s'occupe de la transmission des données entre noeuds.
    - **UDP** (User Datagram Protocol)
      - le protocole UDP n'est pas orienté connexion. Il est utilisé pour la transmission rapide de messages entre nœuds sans garantir leur acheminement.
    - **Telnet**
      - le protocole Telnet est utilisé pour établir une connexion de terminal à distance. Il se trouve dans la couche d'Application du modèle OSI.
    - **Ftp** (File Transfer Protocol)
      - le protocole ftp est utilisé pour le transfert de fichiers. Il se trouve dans la couche d'Application du modèle OSI.
    - **SMTP** (Simple Message Transfer Protocol)
      - le service SMTP est utilisé pour le transfert de courrier électronique. Il se trouve dans la couche d'Application du modèle OSI.
    - **DNS** (Domain Name Service)
      - le service DNS est utilisé pour le résolution de noms en adresses IP. Il se trouve dans la couche d'Application du modèle OSI.
    - **SNMP** (Simple Network Management Protocol)

        - le protocole SNMP est composé d'un agent et un gestionnaire. L'agent SNMP collecte des informations sur les périphériques, les configurations et les performances tandis que le gestionnaire SNMP reçois ses informations et réagit en conséquence.
    - **NFS** (Network File System)
        - le NFS a été mis au point par Sun Microsystems
        - le NFS génère un lien virtuel entre les lecteurs et les disques durs permettant de monter dans un disque virtuel local un disque distant
    - et aussi POP3, NNTP, IMAP etc ...

<note tip> **Cliquez ici pour voir les modèles TCP/IP et OSI** </note>

Le modèle TCP/IP est composé de 4 couches :

- La couche d'Accès Réseau
    - Cette couche spécifie la forme sous laquelle les données doivent être acheminées, quelque soit le type de réseau utilisé.
- La couche Internet
    - Cette couche est chargée de fournir le paquet de données.
- La couche de Transport
    - Cette couche assure l'acheminement des données et se charge des mécanismes permettant de connaître l'état de la transmission.
- La couche d'Application
    - Cette couche englobe les applications standards de réseau telles ftp, telnet, ssh, etc..

Les noms des Unités de Données sont différents selon le protocole utilisé et la couche du modèle TCP/IP :

Couche	**TCP**	**UDP**
Application	Stream	Message
Transport	Segment	Packet
Internet	Datagram	Datagram
Réseau	Frame	Frame

Les Raccordements

Les Modes de Transmission

On peut distinguer 3 modes de transmission :

- La **Liaison Simplex**,
  - Les données ne circulent que dans un **seul** sens de l'émetteur ver le récepteur,
  - La liaison nécessite deux canaux de transmissions,
- La **Liaison Half-Duplex** aussi appelée la **Liaison à l'Alternat** ou encore la **Liaison Semi-Duplex**,
  - Les données circulent dans un sens ou l'autre mais jamais dans les deux sens en même temps. Chaque extrémité émet donc à son tour,
  - La liaison permet d'avoir une liaison bi-directionnelle qui utilise la totalité de la banse passante,
- La **Liaison Full-Duplex** dans les deux sens en **même** temps. Chaque extrémité peut émettre et recevoir simultanément,
  - La liaison est caractérisée par une bande passante divisée par deux pour chaque sens des émissions.

Les Câbles

Le Câble Coaxial

En partant de l'extérieur, le câble coaxial est composé :

- d'une **Gaine** en caoutchouc, PVC ou Téflon pour protéger le câble,
- d'un **Blindage** en métal pour diminuer le bruit du aux parasites,
- d'un **Isolant** (diélectrique) pour éviter le contact entre le blindage et l'âme et ainsi éviter des courts-circuits,
- d'un **Âme** en cuivre ou torsadés pour transporter les données.

Avantages :

- **Peux coûteux**,
- Facilement **manipulable**,
- Peut être utilisé pour de **longues distances**,
- A un débit de 10 Mbit/s dans un LAN et 100 Mbit/s dans un WAN.

Inconvénients :

- Fragile,
- Instable,

- Vulnérable aux interférences,
- Half-Duplex.

Le Câble Paire Torsadée

Ce câble existe sous deux formes selon son utilisation :

- **Monobrin** pour du câblage **horizontal** (**Capillaire**),
  - chaque fil est composé d'un seul conducteur en cuivre,
  - la distance ne doit pas dépassée 90m.
- **Multibrin** pour des **cordons de brassage** :
  - chaque fil est composé de plusieurs brins en cuivre,
  - câble souple.

Avantages :

- Un débit de 10 Mbit/s à 10 GBit/s,
- A une bande passante plus large,
- Pas d'interruption par coupure du câble,
- Permet le **câblage universel** (téléphonie, fax, données ...),
- Full-Duplex.

Inconvénients :

- Nombre de câbles > câble coaxial,
- Plus cher,
- Plus encombrant dans les gaines techniques.

Catagories de Blindage

Il existe trois catagories de blindage :

- **Twisted** ou Torsadé,
- **Foiled** ou Entouré,
- **Shielded** ou Avec Ecran.

De ce fait, il existe 5 catagories de câbles Paire Torsadée :

Nom anglais ^ Appelation Ancienne ^ Nouvelle Appelation ^

Unshielded Twisted Pair	UTP	U/UTP
Foiled Twisted Pair	FTP	F/UTP
Shield Twisted Pair	STP	S/UTP
Shield Foiled Twisted Pair	SFTP	SF/UTP
Shield Shield Twisted Pair	S/STP	SS/STP3

Ces catégories donnent lieu à des **Classes** :

Classe	**Débit**	**Nombre de Paires Torsadées**	**Connecteur**	**Commentaires**
3	10 Mbit/s	4	RJ11	
4	16 Mbit/s	4	S/O	Non-utilisée de nos jours
5	100 Mbit/s	4	RJ45	Obsolète
5e/D	1 Gbit/s sur 100m	4	RJ45	S/O
6/E	2.5 Gbit/s sur 100m ou 10 Gbit/s sur 25m à 55m	4	Idéal pour PoE	
7/F	10 Gbit/s sur 100m	4	GG45 ou Tera	Paires individuellement et collectivement blindées. Problème de compatibilité avec les classes precédentes due au connecteur.

La Prise RJ45

Une prise RJ45 comporte 8 broches. Un câble peut être **droit** quand la broche 1 d'une extremité est connectée à la broche 1 de la prise RJ45 à l'autre extrémité, la broche 2 d'une extremité est connectée à la broche 2 de la prise RJ45 à l'autre extrémité et ainsi de suite ou bien **croisé** quand le brochage est inversé.

Les câbles croisés sont utilisés lors du branchement de deux équipements identiques (PC à PC, Hub à Hub, Routeur à Routeur).

Channel Link et Basic Link

Le **Channel Link** ou **Canal** est l'ensemble du **Basic Link** ou **Lien** de base et les cordons de brassage et de raccordement des équipements qui sont limités en distance à 10m.

Le **Basic Link** est le lien entre la prise RJ45 murale et la baie de brassage. Il est limité à 90m en classe 5D.

La Fibre Optique

La **Fibre Optique** est un fil de **Silice** permettant le transfert de la lumière. De ce fait elle est caractérisée par :

- des meilleures performances que le cuivre,
- de plus de communications simultanément,
- de la capacité de relier de plus grandes distances,
- une insensibilité aux perturbations,
- une résistance à la corrosion.

Qui plus est, elle ne produit aucune perturbation.

Elle est composée :

- d'un coeur de 10, de 50/125 ou de 62.50 micron,
- d'une gaine de 125 micron,
- d'une protection de 230 micron.

Il existe deux types de fibres, la **Fibre Monomode** et la **Fibre Multimodes**.

La Fibre Monomode :

- a un coeur de 8 à 10 Microns,
- est divisée en sous-catégories de distance,

    - 10 Km,
    - 15 Km,
    - 20 Km,
    - 50 Km,
    - 80 Km,
    - 100 Km.

La Fibre Multimode :

- a un coeur de 62,50 micron ou de 50/125 micron avec une gaine orange,
- permet plusieurs trajets lumineux appelés **modes** en même temps en Full Duplex,
- est utilisée pour de bas débits ou de courtes distances,
    - 2 Km pour 100 Mbit/s,
    - 500 m pour 1 Gbit/s.

Les Réseaux sans Fils

Les réseaux sans fils sans basés sur une liaison qui utilise des ondes radio-électriques (radio et infra-rouges).

Il existe des technologies différentes en fonction de la fréquence utilisée et de la portée des transmissions :

- Réseaux Personnels sans Fils - Bluetooth, HomeRF,
- Réseaux Locaux sans Fils - LiFI, WiFI,
- Réseaux Métropolitains sans Fil - wImax,
- Réseaux Etendus sans Fils - GSM, GPRS, UMTS.

Les principales ondes utilisées pour la transmission des données sont :

- Ondes GSM - Ondes Hertziennes repeosant sur des micro-ondes à basse fréquence avec une portée d'une dizaine de kilomètres,
- Ondes Wi-Fi - Ondes Hertziennes reposant sur des micro-ondes à haute fréquence avec une portée de 20 à 50 mètres,
- Ondes Satellitaires - Ondes Hertziennes longues portées.

Le Courant Porteur en Ligne

Le CPL utilise le réseau électrique domestique, le réseau moyenne et basse tension pour transmettre des informations numériques.

Le CPL superpose un signal à plus haute fréquence au signal électrique.

Seuls donc, les fils conducteurs transportent les signaux CPL.

Le coupleur intégré en entrée des boîtiers CPL élimine les composants basses fréquences pour isoler le signal CPL.

Le CPL utilise la phase électrique et le neutre. De ce fait, une installation triphasée fournit 3 réseaux CPL différents.

Le signal CPL ne s'arrête pas necéssairement aux limites de l'installation électrique. En effet en cas de compteurs non-numériques le signal les traversent.

Les normes CPL sont :

Norme	Débit Théorique	Débit Pratique	Temps pour copier 1 Go
Homeplug 1.01	14 Mbps	5.4 Mbps	25m 20s
Homeplug 1.1	85 Mbps	12 Mbps	11m 20s
PréUPA 200	200 Mbps	30 Mbps	4m 30s

Technologies

Il existe plusieurs technologies de réseau :

- Ethernet,
- Token-Ring,
- ARCnet,
- etc..

Nous détaillerons ici les deux technologies les plus répandues, à savoir Ethernet et Token-Ring.

Ethernet

La technologie Ethernet se repose sur :

- une topologie logique de bus,
- une topologie physique de bus ou étoile.

L'accès au bus utilise le **CSMA/CD**, Carrier Sense Multiple Access / Collision Detection (Accès Multiple à Détection de Porteuse / Détection de Collisions).

Il faut noter que :

- les données sont transmises à chaque nœud - c'est la méthode d'**accès multiple**,
- chaque nœud qui veut émettre écoute le réseau - c'est la **détection de porteuse**,
- quand le réseau est silencieux une trame est émise dans laquelle se trouvent les données ainsi que l'adresse du destinataire,
- le système est dit donc **aléatoire** ou **non-déterministe**,
- quand deux nœuds émettent en même temps, il y a **collision de données**,
- les deux nœuds vont donc cesser d'émettre, se mettant en attente jusqu'à ce qu'ils commencent à émettre de nouveau.

Token-Ring

La technologie Token-Ring se repose sur :

- une topologie logique en anneau,
- une topologie physique en étoile.

Token-Ring se traduit par **Anneau à Jeton**. Il n'est pas aussi répandu que l'Ethernet pour des raisons de coûts. En effet le rajout d'un nœud en Token-Ring peut coûter jusqu'à **4 fois plus cher qu'en Ethernet**.

Il faut noter que :

- les données sont transmises dans le réseau par un système appelé **méthode de passage de jeton**,
- le jeton est une **trame numérique vide** de données qui tourne en permanence dans l'anneau,
- quand un nœud souhaite émettre, il saisit le jeton, y dépose des données avec l'adresse du destinataire et ensuite laisse poursuivre son chemin

jusqu'à sa destination,
- pendant son voyage, aucun autre nœud ne peut émettre,
- une fois arrivé à sa destination, le jeton dépose ses données et retourne à l'émetteur pour confirmer la livraison,
- ce système est appelé **déterministe**.

L'intérêt de la technologie Token-Ring se trouve dans le fait :

- qu'il **évite des collisions**,
- qu'il est **possible de déterminer avec exactitude le temps que prenne l'acheminement des données**.

La technologie Token-Ring est donc idéale, voire obligatoire, dans des installations où chaque nœud doit disposer d'une opportunité à intervalle fixe d'émettre des données.

Périphériques Réseaux Spéciaux

En plus du câblage, les périphériques de réseau spéciaux sont des éléments primordiaux tant au niveau de la topologie physique que la topologie logique.

Les périphériques de réseau spéciaux sont :

- les Concentrateurs ou *Hubs*,
- les Répéteurs ou *Repeaters*,
- les Ponts ou *Bridges*,
- les Commutateurs ou *Switches*,
- les Routeurs ou *Routers*,
- les Passerelles ou *Gateways*.

L'objectif ici est de vous permettre de comprendre le rôle de chaque périphérique.

Les Concentrateurs

Les Concentrateurs permettent une connectivité entre les nœuds en topologie en étoile. Selon leur configuration, la topologie logique peut être en

étoile, en bus ou en anneau. Il existe de multiples types de Concentrateurs allant du plus simple au Concentrateur intelligent.

- **Le Concentrateur Simple**
  - est une boîte de raccordement centrale,
  - joue le rôle de récepteur et du réémetteur des signaux sans accélération ni gestion de ceux-ci,
  - est un périphérique utilisé pour des groupes de travail.
- **Le Concentrateur Évolué**
  - est un Concentrateur simple qui offre en plus l'amplification des signaux, la gestion du type de topologie logique grâce à des capacités d'être configurés à l'aide d'un logiciel ainsi que l'homogénéisation du réseau en offrant des ports pour un câblage différent. Par exemple, 8 ports en paire torsadée non-blindée et un port BNC.
- **Le Concentrateur Intelligent**
  - est un Concentrateur évolué qui offre en plus la détection automatique des pannes, la connectique avec un Pont ou un Routeur ainsi que le diagnostic et la génération de rapports.

Les Répéteurs

Un Répéteur est un périphérique réseau simple. Il est utilisé pour amplifier le signal quand :

- la longueur du câble dépasse la limite autorisée,
- le câble passe par une zone ou les interférences sont importantes.

Éventuellement, et uniquement dans le cas où le Répéteur serait muni d'une telle fonction, celui-ci peut être utiliser pour connecter deux réseaux ayant un câblage différent.

Les Ponts

Un Pont est **Répéteur intelligent**. Outre sa capacité d'amplifier les signaux, le Pont analyse le trafic qui passe par lui et met à jour une liste d'adresses des cartes réseau, appelée **une table de routage**, n'autorisant que les transmissions destinées à d'autres segments du réseau.

Les **diffusions** sont néanmoins autorisées.

Comme un Pont doit être intelligent, on utilise souvent un micro-ordinateur comme Pont. Forcément équipé de 2 cartes réseau, le Pont peut également jouer le rôle de serveur de fichiers.

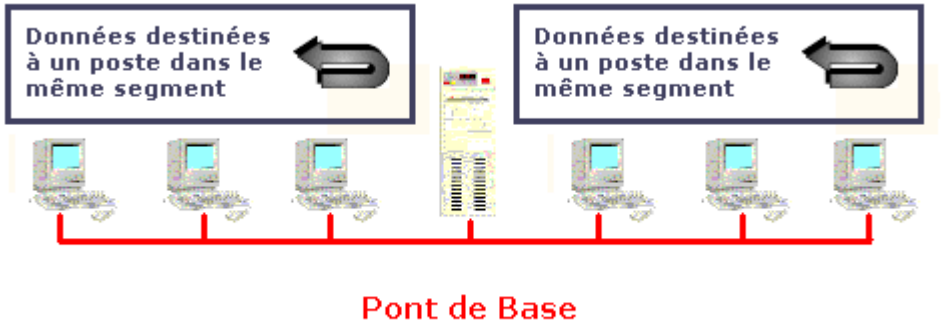
Le Pont sert donc à isoler des segments du réseau pour des raisons de :

- **sécurité** afin d'éviter à ce que des données sensibles soient propagées sur tout le réseau,
- **performance** afin qu'une partie du réseau trop chargée ralentisse le réseau entier,
- **fiabilité** afin par exemple qu'une carte en panne ne gène pas le reste du réseau avec une diffusion.

Il existe trois types de configuration de Ponts

Le Pont de Base

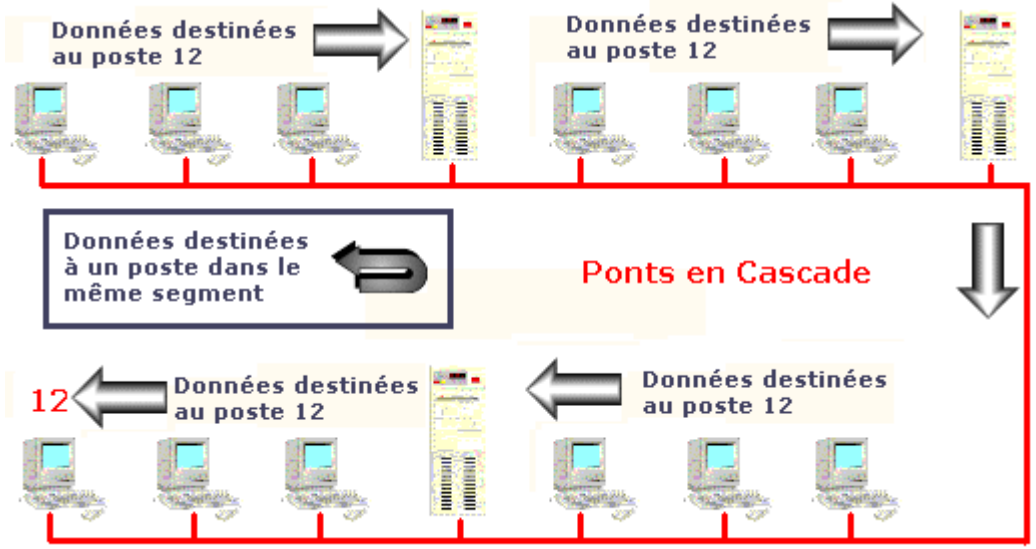
Le Pont de Base est utilisé très rarement pour isoler deux segments.



;#; ;#;

Le Pont en Cascade

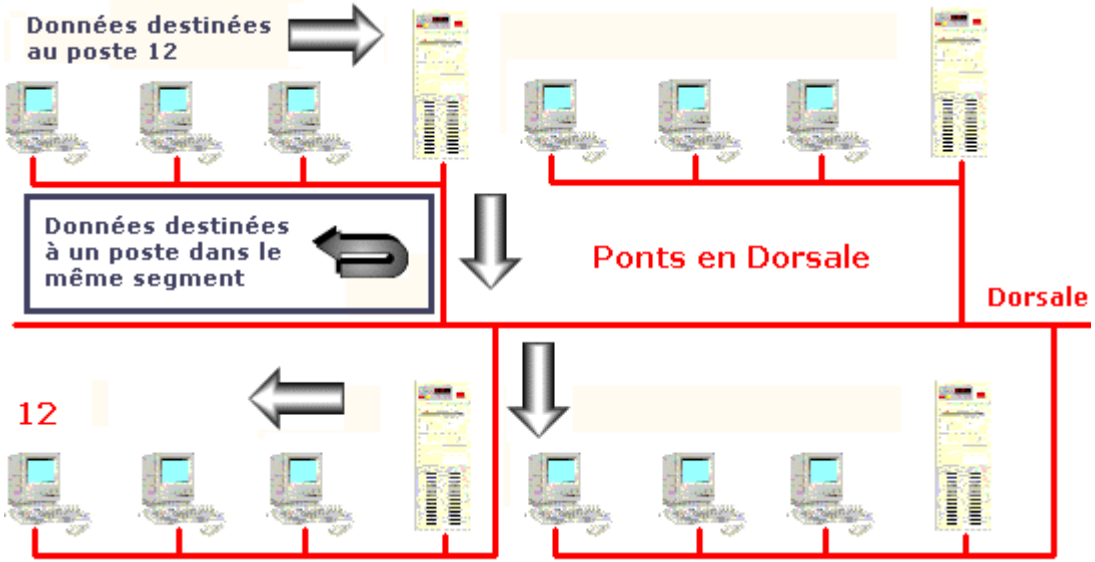
Le Pont en Cascade est à éviter car les données en provenance d'un segment doivent passer par plusieurs Ponts. Ceci a pour conséquence de ralentir la transmission des données, voire même de créer un trafic superflu en cas de rémission par le nœud



;#; ;#;

Le Pont en Dorsale

Le Pont en Dorsale coûte plus chère que la configuration précédente car il faut un nombre de Ponts équivalent au nombre de segments + 1. Par contre elle réduit les problèmes précédemment cités puisque les données ne transitent que par deux Ponts.



;#; ;#;

Les Commutateurs

Un Commutateur peut être considéré comme un Concentrateur intelligent et un Pont. Ils sont gérés souvent par des logiciels. La topologie physique d'un réseau commuté est en étoile. Par contre la topologie logique est spéciale, elle s'appelle une topologie commutée.

Lors de la communication de données entre deux nœuds, le Commutateur ouvre une connexion temporaire virtuelle en fermant les autres ports. De cette façon la bande passante totale est disponible pour cette transmission et les risques de collision sont minimisés.

Certains Commutateurs haut de gamme sont équipés d'un système anti-catastrophe qui leur permet d'isoler une partie d'un réseau en panne afin que les autres parties puissent continuer à fonctionner sans problème.

Les Routeurs

Un Routeur est un Pont sophistiqué capable :

- d'assurer l'interconnexion entre des segments,
- de filtrer le trafic,
- d'isoler une partie du réseau,
- d' explorer les informations d'adressage pour trouver le chemin le plus approprié et le plus rentable pour la transmission des données.

Les Routeurs utilisent une table de routage pour stocker les informations sur :

- les adresses du réseau,
- les solutions de connexion vers d'autres réseaux,
- l'efficacité des différentes routes.

Il existe deux types de Routeur :

- le **Routeur Statique**
  - la table de routage est éditer manuellement,
  - les routes empruntées pour la transmission des données sont toujours les mêmes,
  - il n'y a pas de recherche d'efficacité.
- le **Routeur Dynamique**
  - découvre automatiquement les routes à emprunter dans un réseau.

Les Passerelles

Ce périphérique, souvent un logiciel, sert à faire une conversion de données :

- entre deux technologies différentes (Ethernet - Token-Ring),
- entre deux protocoles différents,
- entre des formats de données différents.

Annexe #2 - Comprendre TCP Version 4

En-tête TCP

L'en-tête TCP est codée sur 4 octets soit 32 bits :

1er octet	2ème octet	3ème octet	4 ème octet
Port source		Port destination	
Numéro de séquence			
Numéro d'acquittement			
Offset	Flags	Fenêtre	
Checksum		Pointeur Urgent	
Options			Padding
Données			

Vous noterez que les numéros de ports sont codés sur 16 bits. Cette information nous permet de calculer le nombres de ports maximum en IPv4, soit 2^{16} ports ou 65 535.

L'**Offset** contient la taille de l'en-tête.

Les **Flags** sont :

- URG - Si la valeur est 1 le pointeur urgent est utilisé. Le numéro de séquence et le pointeur urgent indique un octet spécifique.
- ACK - Si la valeur est 1, le paquet est un accusé de réception
- PSH - Si la valeur est 1, les données sont immédiatement présentées à l'application
- RST - Si la valeur est 1, la communication comporte un problème et la connexion est réinitialisée
- SYN - Si la valeur est 1, le paquet est un paquet de synchronisation
- FIN - Si la valeur est 1, le paquet indique la fin de la connexion

La **Fenêtre** est codée sur 16 bits. La Fenêtre est une donnée liée au fonctionnement d'expédition de données appelé le **sliding window** ou la **fenêtre glissante**. Puisque il serait impossible, pour des raisons de performance, d'attendre l'accusé de réception de chaque paquet envoyé, l'expéditeur envoie des paquets par groupe. La taille de cette groupe s'appelle la Fenêtre. Dans le cas d'un problème de réception d'une partie de la Fenêtre, toute la Fenêtre est ré-expédiée.

Le **Checksum** est une façon de calculer si le paquet est complet.

Le **Padding** est un champ pouvant être rempli de valeurs nulles de façon à ce que la taille de l'en-tête soit un multiple de 32

En-tête UDP

L'en-tête UDP est codée sur 4 octets soit 32 bits :

1er octet	**2ème octet**	**3ème octet**	**4 ème octet**
Port source		Port destination	
longueur		Checksum	
Données			

L'en-tête UDP a une longueur de 8 octets.

Fragmentation et Ré-encapsulation

La taille limite d'un paquet TCP, l'en-tête comprise, ne peut pas dépasser **65 535 octets**. Cependant chaque réseau est qualifié par son MTU (Maximum Tranfer Unit). Cette valeur est la taille maximum d'un paquet autorisée. L'unité est en **octets**. Pour un réseau Ethernet sa valeur est de 1 500. Quand un paquet doit être expédié sur un réseau ayant un MTU inférieur à sa propre taille, le paquet doit être **fractionné**. A la sortie du réseau, le paquet est reconstitué. Cette reconstitution s'appelle **ré-encapsulation**.

Adressage

L'adressage IP requière que chaque périphérique sur le réseau possède une adresse IP unique de 4 octets, soit 32 bits au format XXX.XXX.XXX.XXX De cette façon le nombre total d'adresses est de 2^{32} = 4.3 Milliards.

Les adresses IP sont divisées en 5 classes, de A à E. Les 4 octets des classes A à C sont divisés en deux, une partie qui s'appelle le **Net ID** qui identifie le réseau et une partie qui s'appelle le **Host ID** qui identifie le hôte :

	1er octet	**2ème octet**	**3ème octet**	**4 ème octet**
A	Net ID	Host ID		

	1er octet	2ème octet	3ème octet	4 ème octet
B	Net ID		Host ID	
C	Net ID			Host ID
D	Multicast			
E	Réservé			

L'attribution d'une classe dépend du nombre de hôtes à connecter. Chaque classe est identifié par un **Class ID** composé de 1 à 3 bits :

Classe	Bits ID Classe	Valeur ID Classe	Bits ID Réseau	Nb. de Réseaux	Bits ID hôtes	Nb. d'adresses	Octet de Départ
A	1	0	7	2^7=128	24	2^{24}=16 777 216	1 - 126
B	2	10	14	2^{14}=16 834	16	2^{16}=65 535	128 - 191
C	3	110	21	2^{21}=2 097 152	8	2^8=256	192 - 223

Dans chaque classe, certaines adresses sont réservées pour un usage privé :

Classe	IP de Départ	IP de Fin
A	10.0.0.0	10.255.255.255
B	172.16.0.0	172.31.255.255
C	192.168.0.0	192.168.255.255

Il existe des adresses particulières ne pouvant pas être utilisées pour identifier un hôte :

Adresse Particulière	Description
169.254.0.0 à 169.254.255.255	Automatic Private IP Addressing de Microsoft
Hôte du réseau courant	Tous les bits du Net ID sont à 0
Adresse de réseau	Tous les bits du Host ID sont à 0
Adresse de diffusion	Tous les bits du Host ID sont à 1

L'adresse de réseau identifie le **segment** du réseau entier tandis que l'adresse de diffusion identifie tous les hôtes sur le segment de réseau.

Afin de mieux comprendre l'adresse de réseau et l'adresse de diffusion, prenons le cas de l'adresse 192.168.10.1 en classe C :

	1er octet	2ème octet	3ème octet	4 ème octet

	1er octet	2ème octet	3ème octet	4 ème octet

	Net ID			Host ID
Adresse IP	192	168	10	1
Binaire	11000000	10101000	000001010	00000001
Calcul de l'adresse de réseau				
Binaire	11000000	10101000	000001010	**00000000**
Adresse réseau	192	168	10	0
Calcul de l'adresse de diffusion				
Binaire	11000000	10101000	000001010	**11111111**
Adresse de diffusion	192	168	10	255

Masques de sous-réseaux

Tout comme l'adresse IP, le masque de sous-réseau compte 4 octets ou 32 bits. Les masques de sous-réseaux permettent d'identifer le Net ID et le Host ID :

Classe	Masque	Notation CIDR
A	255.0.0.0	/8
B	255.255.0.0	/16
C	255.255.255.0	/24

Le terme **CIDR** veut dire **Classless InterDomain Routing**. Le terme Notation CIDR correspond au nombre de bits d'une valeur de 1 dans le masque de sous-réseau.

Quand un hôte souhaite émettre il procède d'abord à l'identification de sa propre adresse réseau par un calcul AND (ET) appliqué à sa propre adresse et son masque de sous-réseau qui stipule :

- 1 x 1 = 1
- 0 x 1 = 0
- 1 x 0 = 0
- 0 x 0 = 0

Prenons le cas de l'adresse IP 192.168.10.1 ayant un masque de 255.255.255.0 :

	1er octet	**2ème octet**	**3ème octet**	**4 ème octet**
Adresse IP	192	168	10	1
Binaire	11000000	10101000	00001010	00000001
Masque de sous-réseau				
Binaire	11111111	11111111	11111111	00000000
Calcul AND	11000000	10101000	00001010	00000000
Adresse réseau	192	168	10	0

Cet hôte essaie de communiquer avec un hôte ayant une adresse IP de 192.168.10.10. Il procède donc au même calcul en appliquant **son propre masque de sous-réseau** à l'adresse IP de l'hôte destinataire :

	1er octet	**2ème octet**	**3ème octet**	**4 ème octet**
Adresse IP	192	168	10	10
Binaire	11000000	10101000	00001010	00001010
Masque de sous-réseau				
Binaire	11111111	11111111	11111111	00000000
Calcul AND	11000000	10101000	00001010	00000000
Adresse réseau	192	168	10	0

Puisque l'adresse réseau est identique dans les deux cas, l'hôte émetteur présume que l'hôte de destination se trouve sur son réseau et envoie les paquets directement sur le réseau sans s'adresser à sa passerelle par défaut.

L'hôte émetteur essaie maintenant de communiquer avec avec un hôte ayant une adresse IP de 192.168.2.1. Il procède donc au même calcul en appliquant **son propre masque de sous-réseau** à l'adresse IP de l'hôte destinataire :

	1er octet	**2ème octet**	**3ème octet**	**4 ème octet**
Adresse IP	192	168	2	1
Binaire	11000000	10101000	00000010	00000001
Masque de sous-réseau				
Binaire	11111111	11111111	11111111	00000000
Calcul AND	11000000	10101000	00000010	00000000

	1er octet	2ème octet	3ème octet	4 ème octet
Adresse réseau	192	168	2	0

Dans ce cas, l'hôte émetteur constate que le réseau de destination 192.168.2.0 n'est pas identique à son propre réseau 192.168.10.0. Il adresse donc les paquets à la passerelle par défaut.

VLSM

Puisque le stock de réseaux disponibles sous IPv4 est presque épuisé, une solution a du être trouvée pour créer des sous-réseaux en attendant l'introduction de l'IPv6. Cette solution s'appelle le VLSM ou Variable Length Subnet Masks. Le VLSM exprime les masques de sous-réseaux au format CIDR.

Son principe est simple. Afin de créer des réseaux différents à partir d'une adresse réseau d'une classe donnée, il convient de réduire le nombre d'hôtes. De cette façon les bits 'libérés' du Host ID peuvent être utilisés pour identifier les sous-réseaux.

Pour illustrer ceci, prenons l'exemple d'un réseau 192.168.1.0. Sur ce réseau, nous pouvons mettre 2^8-2 soit 254 hôtes entre 192.168.1.1 au 192.168.1.254.

Supposons que nous souhaiterions diviser notre réseau en 2 sous-réseaux. Pour coder 2 sous-réseaux, il faut que l'on libère 2 bits du Host ID. Les deux bits libérés auront les valeurs binaires suivantes :

- 00
- 01
- 10
- 11

Les valeurs binaires du quatrième octet de nos adresses de sous-réseaux seront donc :

- 192.168.1.00XXXXXX
- 192.168.1.01XXXXXX
- 192.168.1.10XXXXXX
- 192.168.1.11XXXXXX

où les XXXXXX représentent les bits que nous réservons pour décrire les hôtes dans chacun des sous-réseaux.

Nous ne pouvons pas utiliser les deux sous-réseaux suivants :

- 192.168.1.00XXXXXX
- 192.168.1.11XXXXXX

car ceux-ci correspondent aux débuts de l'adresse réseau 192.168.1.0 et de l'adresse de diffusion 192.168.1.255.

Nous pouvons utiliser les deux sous-réseaux suivants :

- 192.168.1.01XXXXXX
- 192.168.1.10XXXXXX

Pour le premier sous-réseau l'adresse réseau et l'adresse de diffusion sont :

Sous-réseau #1	192	168	1	01XXXXXX
Calcul de l'adresse de réseau				
Binaire	11000000	10101000	00000001	01**000000**
Adresse réseau	192	168	1	**64**
Calcul de l'adresse de diffusion				
Binaire	11000000	10101000	00000001	01**111111**
Adresse de diffusion	192	168	1	**127**

- L'adresse CIDR du réseau est donc 192.168.1.64/26 car le Net ID est codé sur 24+2 bits.
- Le masque de sous-réseau est donc le 11111111.11111111.11111111.11000000 ou le 255.255.255.192
- Nous pouvons avoir 2^6-2 soit 62 hôtes.
- La plage valide d'adresses IP est de 192.168.1.65 à 192.168.1.126

Pour le deuxième sous-réseau l'adresse réseau et l'adresse de diffusion sont :

Sous-réseau #2	192	168	1	10XXXXXX
Calcul de l'adresse de réseau				
Binaire	11000000	10101000	00000001	10**000000**
Adresse réseau	192	168	1	**128**

Calcul de l'adresse de diffusion				
Binaire	11000000	10101000	00000001	10**111111**
Adresse de diffusion	192	168	1	**191**

- L'adresse CIDR du réseau est donc 192.168.1.128/26 car le Net ID est codé sur 24+2 bits.
- Le masque de sous-réseau est donc le 11111111.11111111.11111111.11000000 ou le 255.255.255.192
- Nous pouvons avoir 2^6-2 soit 62 hôtes.
- La plage valide d'adresses IP est de 192.168.1.129 à 192.168.1.190

La valeur qui sépare les sous-réseaux est 64. Cette valeur comporte le nom **incrément**.

Ports et sockets

Afin que les données arrivent aux applications que les attendent, TCP utilise des numéros de ports sur la couche transport. Le numéros de ports sont divisés en trois groupes :

- **Well Known Ports**
  - De 1 à 1023
- **Registered Ports**
  - De 1024 à 49151
- **Dynamic** et/ou **Private Ports**
  - De 49152 à 65535

Le couple **numéro IP:numéro de port** s'appelle un **socket**.

/etc/services

Les ports les plus utilisés sont détaillés dans le fichier **/etc/services** :

```
trainee@debian8:~$ su -
```

```
Password:
root@debian8:~# more /etc/services
# Network services, Internet style
#
# Note that it is presently the policy of IANA to assign a single well-known
# port number for both TCP and UDP; hence, officially ports have two entries
# even if the protocol doesn't support UDP operations.
#
# Updated from http://www.iana.org/assignments/port-numbers and other
# sources like http://www.freebsd.org/cgi/cvsweb.cgi/src/etc/services .
# New ports will be added on request if they have been officially assigned
# by IANA and used in the real-world or are needed by a debian package.
# If you need a huge list of used numbers please install the nmap package.

tcpmux      1/tcp                # TCP port service multiplexer
echo        7/tcp
echo        7/udp
discard     9/tcp        sink null
discard     9/udp        sink null
systat      11/tcp        users
daytime     13/tcp
daytime     13/udp
netstat     15/tcp
qotd        17/tcp        quote
msp     18/tcp               # message send protocol
msp     18/udp
chargen     19/tcp        ttytst source
chargen     19/udp        ttytst source
ftp-data    20/tcp
ftp     21/tcp
fsp     21/udp        fspd
ssh     22/tcp               # SSH Remote Login Protocol
ssh     22/udp
telnet      23/tcp
```

```
smtp        25/tcp        mail
time        37/tcp        timserver
time        37/udp        timserver
rlp     39/udp        resource    # resource location
nameserver  42/tcp        name        # IEN 116
--More--(6%)
```

Notez que les ports sont listés par deux :

- le port TCP
- le port UDP

La liste la plus complète peut être consultée à l'adresse suivante **https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml**.

Pour connaitre la liste des sockets ouverts sur l'ordinateur, saisissez la commande suivante :

```
root@debian8:~# netstat -an | more
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 127.0.0.1:25            0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:42370         0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:15023         0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:111             0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:41012           0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:23              0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:7127          0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:33220         127.0.0.1:50656         ESTABLISHED
tcp        0      0 10.0.2.15:22            10.0.2.2:46432          ESTABLISHED
tcp        0      0 127.0.0.1:50656         127.0.0.1:33220         ESTABLISHED
tcp6       0      0 ::1:25                  :::*                    LISTEN
tcp6       0      0 :::33476                :::*                    LISTEN
tcp6       0      0 ::1:15023               :::*                    LISTEN
```

```
tcp6       0      0 :::111                  :::*                    LISTEN
tcp6       0      0 :::22                   :::*                    LISTEN
tcp6       0      0 ::1:22                  ::1:39236               ESTABLISHED
tcp6       0      0 ::1:39236               ::1:22                  ESTABLISHED
udp        0      0 0.0.0.0:32899           0.0.0.0:*
udp        0      0 0.0.0.0:995             0.0.0.0:*
udp        0      0 0.0.0.0:52452           0.0.0.0:*
udp        0      0 0.0.0.0:5353            0.0.0.0:*
udp        0      0 0.0.0.0:68              0.0.0.0:*
udp        0      0 127.0.0.1:613           0.0.0.0:*
udp        0      0 0.0.0.0:111             0.0.0.0:*
udp        0      0 0.0.0.0:53110           0.0.0.0:*
udp6       0      0 :::17599                :::*
udp6       0      0 :::995                  :::*
udp6       0      0 :::5353                 :::*
udp6       0      0 :::33524                :::*
udp6       0      0 :::40492                :::*
udp6       0      0 :::111                  :::*
Active UNIX domain sockets (servers and established)
Proto RefCnt Flags       Type       State         I-Node   Path
unix  2      [ ACC ]     STREAM     LISTENING     17625    /tmp//.java_pid1791
--More--
```

Pour connaitre la liste des applications ayant ouvert un port sur l'ordinateur, saisissez la commande suivante :

```
root@debian8:~# netstat -anp | more
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 127.0.0.1:25            0.0.0.0:*               LISTEN      868/exim4
tcp        0      0 127.0.0.1:42370         0.0.0.0:*               LISTEN      1791/Remote Access
tcp        0      0 127.0.0.1:15023         0.0.0.0:*               LISTEN      2449/ssh
tcp        0      0 0.0.0.0:111             0.0.0.0:*               LISTEN      396/rpcbind
tcp        0      0 0.0.0.0:41012           0.0.0.0:*               LISTEN      434/rpc.statd
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      471/sshd
```

```
tcp        0      0 0.0.0.0:23              0.0.0.0:*               LISTEN      4041/inetd
tcp        0      0 127.0.0.1:7127          0.0.0.0:*               LISTEN      1791/Remote Access
tcp        0      0 127.0.0.1:33220         127.0.0.1:50656         ESTABLISHED 1879/Remote Access
tcp        0      0 10.0.2.15:22            10.0.2.2:46432          ESTABLISHED 10584/sshd: trainee
tcp        0      0 127.0.0.1:50656         127.0.0.1:33220         ESTABLISHED 1791/Remote Access
tcp6       0      0 ::1:25                  :::*                    LISTEN      868/exim4
tcp6       0      0 :::33476                :::*                    LISTEN      434/rpc.statd
tcp6       0      0 ::1:15023               :::*                    LISTEN      2449/ssh
tcp6       0      0 :::111                  :::*                    LISTEN      396/rpcbind
tcp6       0      0 :::22                   :::*                    LISTEN      471/sshd
tcp6       0      0 ::1:22                  ::1:39236               ESTABLISHED 2415/sshd: trainee
tcp6       0      0 ::1:39236               ::1:22                  ESTABLISHED 2449/ssh
udp        0      0 0.0.0.0:32899           0.0.0.0:*                           419/dhclient
udp        0      0 0.0.0.0:995             0.0.0.0:*                           396/rpcbind
udp        0      0 0.0.0.0:52452           0.0.0.0:*                           482/avahi-daemon: r
udp        0      0 0.0.0.0:5353            0.0.0.0:*                           482/avahi-daemon: r
udp        0      0 0.0.0.0:68              0.0.0.0:*                           419/dhclient
udp        0      0 127.0.0.1:613           0.0.0.0:*                           434/rpc.statd
udp        0      0 0.0.0.0:111             0.0.0.0:*                           396/rpcbind
udp        0      0 0.0.0.0:53110           0.0.0.0:*                           434/rpc.statd
udp6       0      0 :::17599                :::*                                419/dhclient
udp6       0      0 :::995                  :::*                                396/rpcbind
udp6       0      0 :::5353                 :::*                                482/avahi-daemon: r
udp6       0      0 :::33524                :::*                                482/avahi-daemon: r
udp6       0      0 :::40492                :::*                                434/rpc.statd
udp6       0      0 :::111                  :::*                                396/rpcbind
Active UNIX domain sockets (servers and established)
Proto RefCnt Flags       Type       State         I-Node   PID/Program name    Path
unix  2      [ ACC ]     STREAM     LISTENING     17625    1791/Remote Access  /tmp//.java_pid1791
--More--
```

Résolution d'adresses Ethernet

Chaque protocole peut être encapsulé dans une **trame** Ethernet. Lorsque la trame doit être transportée de l'expéditeur au destinataire, ce premier doit connaitre l'adresse Ethernet du dernier. L'adresse Ethernet est aussi appelée l'adresse **Physique** ou l'adresse **MAC**.

Pour connaître l'adresse Ethernet du destinataire, l'expéditeur fait appel au protocol **ARP**. Les informations reçues sont stockées dans une table. Pour visualiser ces informations, il convient d'utiliser la commande suivante :

```
root@debian8:~# arp -a
? (10.0.2.2) at 52:54:00:12:35:02 [ether] on eth0
? (10.0.2.3) at 52:54:00:12:35:03 [ether] on eth0
```

Options de la commande

Les options de cette commande sont :

```
root@debian8:~# arp --help
Usage:
  arp [-vn]  [<HW>] [-i <if>] [-a] [<hostname>]             <-Display ARP cache
  arp [-v]          [-i <if>] -d  <host> [pub]               <-Delete ARP entry
  arp [-vnD] [<HW>] [-i <if>] -f  [<filename>]            <-Add entry from file
  arp [-v]   [<HW>] [-i <if>] -s  <host> <hwaddr> [temp]             <-Add entry
  arp [-v]   [<HW>] [-i <if>] -Ds <host> <if> [netmask <nm>] pub           <-''-

        -a                      display (all) hosts in alternative (BSD) style
        -s, --set               set a new ARP entry
        -d, --delete            delete a specified entry
        -v, --verbose           be verbose
        -n, --numeric           don't resolve names
        -i, --device            specify network interface (e.g. eth0)
        -D, --use-device        read <hwaddr> from given device
        -A, -p, --protocol      specify protocol family
```

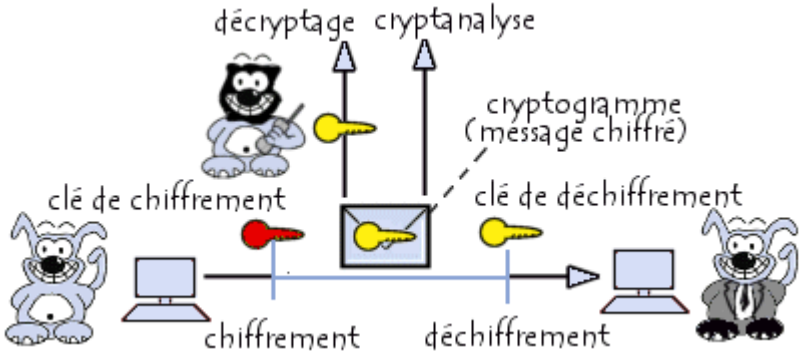
```
       -f, --file              read new entries from file or from /etc/ethers

<HW>=Use '-H <hw>' to specify hardware address type. Default: ether
List of possible hardware types (which support ARP):
  ash (Ash) ether (Ethernet) ax25 (AMPR AX.25)
  netrom (AMPR NET/ROM) rose (AMPR ROSE) arcnet (ARCnet)
  dlci (Frame Relay DLCI) fddi (Fiber Distributed Data Interface) hippi (HIPPI)
  irda (IrLAP) x25 (generic X.25) eui64 (Generic EUI-64)
```

Annexe #3 - Comprendre le Chiffrement

Introduction à la cryptologie

Définitions

- **La Cryptologie**
  - La science qui étudie les aspects scientifiques de ces techniques, c'est-à-dire qu'elle englobe la cryptographie et la cryptanalyse.
- **La Cryptanalyse**
  - Lorsque la clef de déchiffrement n'est pas connue de l'attaquant on parle alors de cryptanalyse ou cryptoanalyse (on entend souvent aussi le terme plus familier de cassage).
- **La Cryptographie**
  - Un terme générique désignant l'ensemble des techniques permettant de chiffrer des messages, c'est-à-dire permettant de les rendre inintelligibles sans une action spécifique. Les verbes crypter et chiffrer sont utilisés.
- **Le Décryptement ou Décryptage**
  - Est le fait d'essayer de déchiffrer illégitimement le message (que la clé de déchiffrement soit connue ou non de l'attaquant).



La Cryptographie

La cryptographie apporte quatre points clefs:

- La confidentialité
  - consiste à rendre l'information inintelligible à d'autres personnes que les acteurs de la transaction.
- L'intégrité
  - consiste à déterminer si les données n'ont pas été altérées durant la communication (de manière fortuite ou intentionnelle).
- L'authentification
  - consiste à assurer l'identité d'un utilisateur.
- La non-répudiation
  - est la garantie qu'aucun des correspondants ne pourra nier la transaction.

La cryptographie est basée sur l'arithmétique. Il s'agit, dans le cas d'un texte, de transformer les lettres qui composent le message en une succession de chiffres (sous forme de bits dans le cas de l'informatique), puis ensuite de faire des calculs sur ces chiffres pour:

- Procéder au chiffrement
  - Le résultat de cette modification (le message chiffré) est appelé cryptogramme (Ciphertext) par opposition au message initial, appelé message en clair (Plaintext)
- Procéder au déchiffrement

Le chiffrement se fait à l'aide d'une clef de chiffrement. Le déchiffrement nécessite une clef de déchiffrement.

On distingue deux types de clefs:

- Les clés symétriques:
  - des clés utilisées pour le chiffrement ainsi que pour le déchiffrement. On parle alors de chiffrement symétrique ou de chiffrement à clé secrète.
- Les clés asymétriques:
  - des clés utilisées dans le cas du chiffrement asymétrique (aussi appelé chiffrement à clé publique). Dans ce cas, une clé différente est utilisée pour le chiffrement et pour le déchiffrement.

Le Chiffrement par Substitution

Le chiffrement par substitution consiste à remplacer dans un message une ou plusieurs entités (généralement des lettres) par une ou plusieurs autres entités. On distingue généralement plusieurs types de cryptosystèmes par substitution :

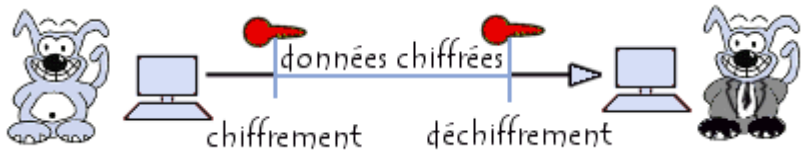
- La substitution **monoalphabétique**
  - consiste à remplacer chaque lettre du message par une autre lettre de l'alphabet
- La substitution **polyalphabétique**
  - consiste à utiliser une suite de chiffres monoalphabétique réutilisée périodiquement
- La substitution **homophonique**
  - permet de faire correspondre à chaque lettre du message en clair un ensemble possible d'autres caractères
- La substitution de **polygrammes**
  - consiste à substituer un groupe de caractères (polygramme) dans le message par un autre groupe de caractères

Algorithmes à clé secrète

Le Chiffrement Symétrique

Ce système est aussi appelé le système à **Clef Secrète** ou à **clef privée**.

Ce système consiste à effectuer une opération de chiffrement par algorithme mais comporte un inconvénient, à savoir qu'il nécessite un canal sécurisé pour la transmission de la clef de chiffrement/déchiffrement.



Le système de Méthode du Masque Jetable (One Time Pad) fût mis au point dans les années 1920. Il utilisait une clef générée aléatoirement à usage unique.

Les algorithmes de chiffrement symétrique couramment utilisés en informatique sont:

- **Data Encryption Standard** (DES),
- **Triple DES** (3DES),
- **RC2**,
- **Blowfish**,
- **International Data Encryption Algorithm** (IDEA),
- **Advanced Encryption Standard** (AES).

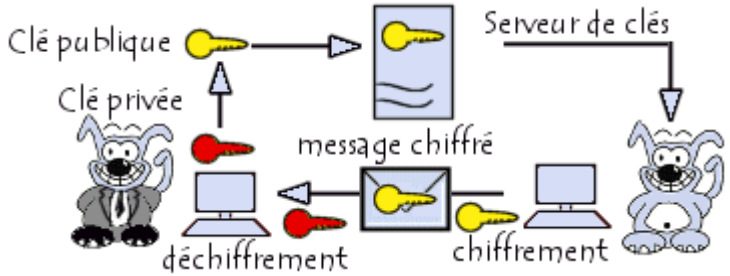
Algorithmes à clef publique

Le Chiffrement Asymétrique

Ce système est aussi appelé **Système à Clef Publique**.

Ce système consiste à avoir deux clefs appelées des **bi-clefs**:

- Une clef **publique** pour le chiffrement
- Une clef **secrète** ou **privée** pour le déchiffrement



- L'utilisateur A (celui qui déchiffre) choisit une clef privée.
- A partir de cette clef il génère plusieurs clefs publiques grâce à un algorithme.
- L'utilisateur B (celui qui chiffre) choisit une des clefs publiques à travers un canal non-sécurisé pour chiffrer les données à l'attention de l'utilisateur A.

Ce système est basé sur ce que l'on appelle une **fonction à trappe à sens unique** ou **one-way trap door**.

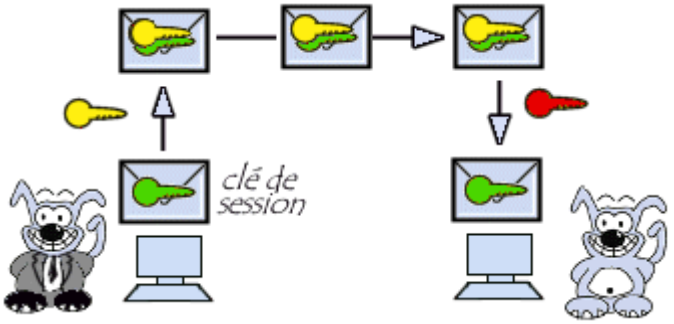
Il existe toutefois un problème – s'assurer que la clef publique récupérée est bien celle qui correspond au destinataire !

Les algorithmes de chiffrement asymétrique couramment utilisés en informatique sont:

- **Digital Signature Algorithm** (DSA)
- **Rivest, Shamir, Adleman** (RSA)

La Clef de Session

Ce système est un compromis entre le système symétrique et le système asymétrique. Il permet l'envoie de données chiffrées à l'aide d'un algorithme de chiffrement symétrique par un canal non-sécurisé et a été mis au point pour palier au problème de lenteur de déchiffrement du système asymétrique.

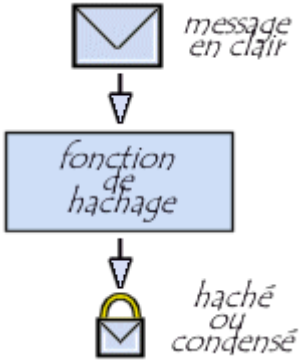


Ce système fonctionne de la façon suivante :

- L'utilisateur A chiffre une clef privée générée aléatoirement, appelée une « clef de session », en utilisant une des clefs publiques de l'utilisateur B.
- L'utilisateur A chiffre les données avec la clef de session.
- L'utilisateur B déchiffre la clef de session en utilisant sa propre clef privée.
- L'utilisateur B déchiffre les données en utilisant la clef de session.

Fonctions de Hachage

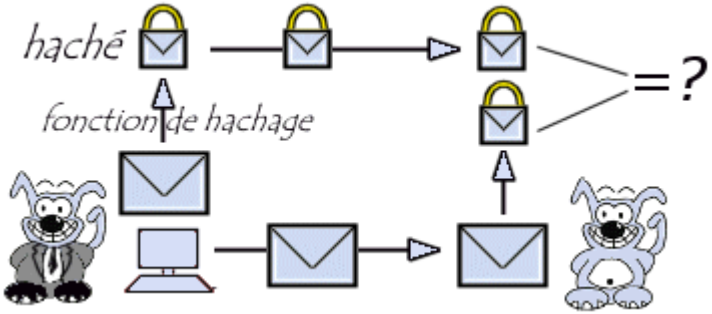
La fonction de **hachage**, aussi appelée une fonction de **condensation**, est à **sens unique** (one way function). Il « condense » un message en clair et produit un haché unique.



Les deux algorithmes de hachage utilisés sont:

- **Message Digest 5** (MD5)
- **Secure Hash Algorithm** (SHA)

Lors de son envoie, le message est accompagné de son haché et il est donc possible de garantir son intégrité:

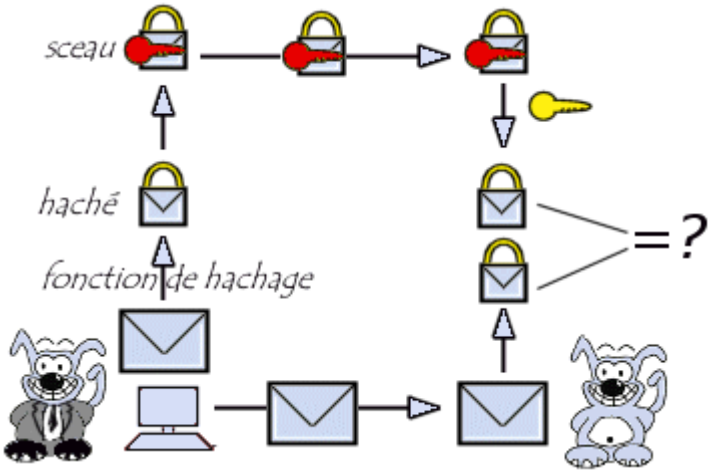


- A la réception du message, le destinataire ou l'utilisateur B calcule le haché du message reçu et le compare avec le haché accompagnant le document.
- Si le message ou le haché a été falsifié durant la communication, les deux empreintes ne correspondront pas.

> Ce système permet de vérifier que l'empreinte correspond bien au message reçu, mais ne permet pas de prouver que le message a bien été envoyé par l'utilisateur A.

Signature Numérique

Pour garantir l'authentification du message l'utilisateur A va chiffrer ou **signer** le haché à l'aide de sa clé privée. Le haché signé est appelé un **sceau**.



- L’utilisateur A envoie le sceau au destinataire.
- A la réception du message L’utilisateur B déchiffre le sceau avec la clé publique de l’utilisateur A.
- Il compare le haché obtenu au haché reçu en pièce jointe.

Ce mécanisme de création de sceau est appelé **scellement**.

Ce mécanisme est identique au procédé utilisé par SSH lors d'une connexion

Utilisation de GnuPG

Présentation

GNU Privacy Guard permet aux utilisateurs de transférer des messages chiffrés et/ou signés.

Installation

Sous RHEL/CentOS 7, le paquet gnupg est installé par défaut :

```
root@debian8:~# whereis gpg
gpg: /usr/bin/gpg /usr/share/man/man1/gpg.1.gz
```

Configuration

Pour initialiser GnuPG, saisissez la commande suivante :

```
root@debian8:~# whereis gpg
gpg: /usr/bin/gpg /usr/share/man/man1/gpg.1.gz
root@debian8:~# gpg
gpg: directory `/root/.gnupg' created
gpg: new configuration file `/root/.gnupg/gpg.conf' created
gpg: WARNING: options in `/root/.gnupg/gpg.conf' are not yet active during this run
gpg: keyring `/root/.gnupg/secring.gpg' created
gpg: keyring `/root/.gnupg/pubring.gpg' created
gpg: Go ahead and type your message ...
^C
gpg: Interrupt caught ... exiting
```

Pour aider gpg dans la génération des clefs, utilisez **rngd** pour fournir suffisament d“Entropy” au noyau :

```
root@debian8:~# apt-get install rng-tools
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  rng-tools
0 upgraded, 1 newly installed, 0 to remove and 95 not upgraded.
Need to get 46.8 kB of archives.
After this operation, 209 kB of additional disk space will be used.
Get:1 http://ftp.fr.debian.org/debian/ jessie/main rng-tools amd64 2-unofficial-mt.14-1 [46.8 kB]
Fetched 46.8 kB in 0s (146 kB/s)
```

```
Selecting previously unselected package rng-tools.
(Reading database ... 82472 files and directories currently installed.)
Preparing to unpack .../rng-tools_2-unofficial-mt.14-1_amd64.deb ...
Unpacking rng-tools (2-unofficial-mt.14-1) ...
Processing triggers for systemd (215-17+deb8u4) ...
Processing triggers for man-db (2.7.0.2-5) ...
Setting up rng-tools (2-unofficial-mt.14-1) ...
Job for rng-tools.service failed. See 'systemctl status rng-tools.service' and 'journalctl -xn' for details.
invoke-rc.d: initscript rng-tools, action "start" failed.
Processing triggers for systemd (215-17+deb8u4) ...

root@debian8:~# rngd -f -r /dev/urandom
rngd 2-unofficial-mt.14 starting up...
entropy feed to the kernel ready
^Cstats: bits received from HRNG source: 60064
stats: bits sent to kernel pool: 4096
stats: entropy added to kernel pool: 4096
stats: FIPS 140-2 successes: 3
stats: FIPS 140-2 failures: 0
stats: FIPS 140-2(2001-10-10) Monobit: 0
stats: FIPS 140-2(2001-10-10) Poker: 0
stats: FIPS 140-2(2001-10-10) Runs: 0
stats: FIPS 140-2(2001-10-10) Long run: 0
stats: FIPS 140-2(2001-10-10) Continuous run: 0
stats: HRNG source speed: (min=64.005; avg=64.949; max=65.998)Mibits/s
stats: FIPS tests speed: (min=99.861; avg=111.541; max=124.663)Mibits/s
stats: Lowest ready-buffers level: 2
stats: Entropy starvations: 0
stats: Time spent starving for entropy: (min=0; avg=0.000; max=0)us
Exiting...

root@debian8:~# cat /proc/sys/kernel/random/entropy_avail
2022
```

Pour générer les clefs, saisissez la commande suivante :

```
root@debian8:~# gpg --gen-key
gpg (GnuPG) 1.4.18; Copyright (C) 2014 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Please select what kind of key you want:
   (1) RSA and RSA (default)
   (2) DSA and Elgamal
   (3) DSA (sign only)
   (4) RSA (sign only)
Your selection? 1
RSA keys may be between 1024 and 4096 bits long.
What keysize do you want? (2048)
Requested keysize is 2048 bits
Please specify how long the key should be valid.
         0 = key does not expire
      <n>  = key expires in n days
      <n>w = key expires in n weeks
      <n>m = key expires in n months
      <n>y = key expires in n years
Key is valid for? (0) 0
Key does not expire at all
Is this correct? (y/N) y

You need a user ID to identify your key; the software constructs the user ID
from the Real Name, Comment and Email Address in this form:
    "Heinrich Heine (Der Dichter) <heinrichh@duesseldorf.de>"

Real name: I2TCH
Email address: infos@i2tch.eu
Comment: Test Key
You selected this USER-ID:
```

```
    "I2TCH (Test Key) <infos@i2tch.eu>"

Change (N)ame, (C)omment, (E)mail or (O)kay/(Q)uit? o
You need a Passphrase to protect your secret key.

We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.

Not enough random bytes available.  Please do some other work to give
the OS a chance to collect more entropy! (Need 158 more bytes)
.......+++++
.+++++
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
+++++
..+++++
gpg: /root/.gnupg/trustdb.gpg: trustdb created
gpg: key 324951F4 marked as ultimately trusted
public and secret key created and signed.

gpg: checking the trustdb
gpg: 3 marginal(s) needed, 1 complete(s) needed, PGP trust model
gpg: depth: 0  valid:   1  signed:   0  trust: 0-, 0q, 0n, 0m, 0f, 1u
pub   2048R/324951F4 2016-08-07
      Key fingerprint = 293A 4AB0 C917 DEAD 1838  FAE6 B112 5F50 3249 51F4
uid                  I2TCH (Test Key) <infos@i2tch.eu>
sub   2048R/315943DA 2016-08-07
```

La liste de clefs peut être visualisée avec la commande suivante :

```
root@debian8:~# gpg --list-keys
/root/.gnupg/pubring.gpg
------------------------
pub   2048R/324951F4 2016-08-07
uid                  I2TCH (Test Key) <infos@i2tch.eu>
sub   2048R/315943DA 2016-08-07
```

Pour importer la clef d'un correspondant dans sa trousse de clefs il convient d'utiliser la commande suivante :

```
# gpg --import la-clef.asc
```

Pour exporter sa clef publique, il convient d'utiliser la commande suivante :

```
root@debian8:~# gpg --export --armor I2TCH > ~/I2TCH.asc
root@debian8:~# cat I2TCH.asc
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v1

mQENBFem/AUBCADD1ExMqUny634CUdpqY8zvkaaPdg3JdV3dSmUEMOhhqyaftp++
RLqhi7F7+/uSTNv1I611xsVQ1zgPO8YTIwM2c/glIKa6pRTfPFO7qzPczFMhwNrO
a2jnwJjlqg4/BB8N0xWry2TTOjn6lTcwtWcFzjy0jmjFDQLut0/85mnGNB2aml9r
iaSMF2XwvjSIz6nPv0EVvfzzLUdebKBDBrqiZebCbCfeTzR3yyMr9rA8QWdfCHuW
yQaI/FO9BvHdv6CTmBXlKcfu+nTuBfhUfWlWIYn2549Fy47KIdLS+rB7b91FlUw+
8kidI4AVIUm2WlmpLzTJN/0m1PuBoHgi8TLzABEBAAG0IUkyVENIIChUZXN0IEtl
eSkgPGluZm9zQGkydGNoLmV1PokBOAQTAQIAIgUCV6b8BQIbAwYLCQgHAwIGFQgC
CQoLBBYCAwECHgECF4AACgkQsRJfUDJJUfSz3ggAwsH18xuOjidLNST2OM+8Sf/6
954Ajp3x7zBSQCihhHnuVL/vwlrIJ8ScHudQDPq42+zghfH+37O1PXU7cv43hsXT
+cLtSRjGtVE3uvScIUodJONGJJM6oOf3doyLsYHfA1511IWViryTGylS5sBBfjcN
/ljiSjE9K/IRqJ6g1iKU8reC5DU+FmjpOJsnTTef2Je91iHfuR+DPZUzgimulI5G
0Nrq4pqOD/o1j55N+AKM3fqlqRlyUuozsT+CfBGC/xN9gj1FTeeof3bwNoWuiD75
8nYA97eX0jQooxyrHq+9HHU6kvFtOVVpUEbgHyZzenQzdbAuYabJTETE7vgMfbkB
```

```
DQRXpvwFAQgArXPkbmoLS/B7eBUm1cTVvkcJET/RG4AcVs4aZNZ24ve8/qNva7Ec
d4A3kG1t3rHKlFlHnsGm8tHw3Jjg+/6WFFAzG4mzm8QrwA+vnmcHmSrhVgCaA0NS
vqOIWCys96bKcwLIJuYDK9kLuUDRRniMcaA2sl44BaVDl9V8HEm3PS3jYbewwCZa
Z7vtiiK39cyn6AatZHBw7ubeYupmtTUc34dfSym5K7jKgg3VOhaVGsDF1OPogBit
w1tLdNHRyxYkIkhV9xtBIaUMSDsfulmMVsQXXCwY27m3EQMc25C7xuQ5K/+TRLsv
DKzcYcPcHd0cT8B7Ym7+Xetq+CdQD1jOSQARAQABiQEfBBgBAgAJBQJXpvwFAhsM
AAoJELESX1AySVH0/PsIAKwbeV1fwcucq3X+afa/DtEmFWlRS50XwqVGb/ADu10R
7XsftBUSRBPMTvKvFNLSOklggKKP2OFz3ZmBttjdJXL+24U48LtLplG2cfXpAzjD
7rMVuKICgJGHlLWr+sT3t2uH/Mw7Rn3aw0a1MpV5GoqrBvKfdcYbTcbp9HSzCaJL
eVYbXTwXRLlyhcPVy2BZl8OVQlizqLkuzonAmzOMUpYAXU983MZlzEKhMp8R/otC
vx/Y+7kCHiWBqZzpD1MktjeSxUWt+jC7z4pb13t7DOFUzysiBdAYhWtCWAIbJdO2
Pi6GtIZkpKnV1Q7Ct6x3K7HJKfpis3YjIZSFq+Gr09o=
=bhYQ
-----END PGP PUBLIC KEY BLOCK-----
```

Cette clef peut ensuite être jointe à des messages électroniques ou bien déposée sur un serveur de clefs tel http://www.keyserver.net.

Signer un message

Créez maintenant un message à signer :

```
root@debian8:~# vi ~/message.txt
root@debian8:~# cat ~/message.txt
This is a test message for gpg
```

Pour signer ce message en format binaire, il convient d'utiliser la commande suivante :

```
root@debian8:~# gpg --default-key I2TCH --detach-sign message.txt

You need a passphrase to unlock the secret key for
user: "I2TCH (Test Key) <infos@i2tch.eu>"
2048-bit RSA key, ID 324951F4, created 2016-08-07
```

```
root@debian8:~# ls -l | grep message
-rw-r--r-- 1 root root       31 Aug  7 10:18 message.txt
-rw-r--r-- 1 root root      287 Aug  7 10:18 message.txt.sig

root@debian8:~# cat message.txt.sig
□□□□□□□□W□□□
    □□_P2IQ□E□□□>□□)7□Lt□□□□C□+□vC□0□□cX□□□□□1N□Z□6□;□□□□□R□□□□Y□□>□gЧ□□□c□□□□T□iO|&□□TF□□□h□□□□u□□□□□V□
                              _ă□□Tf□□9ct□□□ě□□Q□i>□□ж□□□"K□c□□)□□□~□□□□*□u□f%□□DykheG/u□□_□K□'□□}G□H
                        □R^i□tЛь□y□`□[]a□_1□дP)□a□□\□□□□ЭFO$m□□l
                                                                    3z□□□я□{□Ҫe□*H□□w□H□N   □□□□x□ڳ□□
```

Pour signer ce message en format ascii, il convient d'utiliser la commande suivante :

```
root@debian8:~# gpg --default-key I2TCH --armor --detach-sign message.txt

You need a passphrase to unlock the secret key for
user: "I2TCH (Test Key) <infos@i2tch.eu>"
2048-bit RSA key, ID 324951F4, created 2016-08-07

root@debian8:~# ls -l | grep message
-rw-r--r-- 1 root root       31 Aug  7 10:18 message.txt
-rw-r--r-- 1 root root      473 Aug 10 11:29 message.txt.asc
-rw-r--r-- 1 root root      287 Aug  7 10:18 message.txt.sig

root@debian8:~# cat message.txt.asc
-----BEGIN PGP SIGNATURE-----
Version: GnuPG v1

iQEcBAABAgAGBQJXqwICAAoJELESX1AySVH0tqUH/igOKaMwaEIgPCwUpv8rJLIy
2cCj2stliojf437f7eZDyHTNiB/ghmLc6GCogV3uA+jJg2O5vidI5HTD7M2qnnix
CTFRMwH0v61ssZTU/nB+KyO2S3NWRGV1T/dGXJGjXf2QBcSowWoTeQBjbVGDnB3O
twDpWH8NFdW7yGNoKdnPkCdrKEMuUCthLz05W1yCngFDpPwkd8MM9wFq7UdQ+EN9
G2iiyIQbArOI18v67BH571z+4UlOfJlUB+O2C9D8tobJBU7KNSPLWnYAJck6YrrC
```

```
3RVW5M74LcHtfQypSeCJAqxiZf2SMtNU99zPnDqMzwX8tNCpYrlL3IX6DdduOJ4=
=/wL9
-----END PGP SIGNATURE-----
```

Pour vérifier la signature d'un message signé en mode ascii, il convient d'utiliser la commande :

```
root@debian8:~# gpg --verify message.txt.asc
gpg: assuming signed data in `message.txt'
gpg: Signature made Wed 10 Aug 2016 11:34:35 BST using RSA key ID 324951F4
gpg: Good signature from "I2TCH (Test Key) <infos@i2tch.eu>"
```

> Pour vérifier la signature d'un message signé en mode ascii et produit en dehors du message lui-même, il convient d'utiliser la commande :
>
> ```
> # gpg --verify message.txt.asc message.txt
> ```

Pour signer ce message **dans le message lui-même** en format ascii, il convient d'utiliser la commande suivante :

```
root@debian8:~# gpg --default-key I2TCH --clearsign message.txt

You need a passphrase to unlock the secret key for
user: "I2TCH (Test Key) <infos@i2tch.eu>"
2048-bit RSA key, ID 324951F4, created 2016-08-07

File `message.txt.asc' exists. Overwrite? (y/N) y
root@debian8:~# ls -l | grep message
-rw-r--r-- 1 root root       31 Aug  7 10:18 message.txt
-rw-r--r-- 1 root root      551 Aug 10 11:30 message.txt.asc
-rw-r--r-- 1 root root      287 Aug  7 10:18 message.txt.sig
root@debian8:~# cat message.txt.asc
-----BEGIN PGP SIGNED MESSAGE-----
```

```
Hash: SHA1

This is a test message for gpg
-----BEGIN PGP SIGNATURE-----
Version: GnuPG v1

iQEcBAEBAgAGBQJXqwJHAAoJELESX1AySVH0VEMIAJc7NP2v8s/mmllpRi3Kj9W1
oTS721z/XSbM4iQaFM7QnoJmlfCaAbOB01WNqiAVL4A1LGFntttknsoMF7lERU6k
hPCeMtdcWTF3/KwlLHBZ3jjNJeVS4BfJjiW0qStgbMRuzaVHV0iONACA80mnYEOx
TkVnk/IMkO0fsCBocbFeJ/DdR/P9o4u4yqhkwin2+cKPoEWUYBODhIOtHzLMuq1m
582UmrGUQq5z6CC7kiZzifbOtm54pT5MioVfHpYwt6+zlfvYhgVn8VQ62eKAg0zs
IaaRTYVmlD1XUbWxswqvBA9RwIRck6A5Oi5YAoH8jUHaZjvVK9KaEXDQ7Ga/Nk4=
=i5f6
-----END PGP SIGNATURE-----
```

Chiffrer un message

Pour chiffrer un message, il faut disposer de la clef publique du destinataire du message. Ce dernier utilisera ensuite sa clef privée pour déchiffrer le message. Il convient de préciser le destinataire du message, ou plus précisément la clef publique à utiliser, lors d'un chiffrement :

```
  gpg --recipient <destinataire> --encrypt <message>
```

- <*destinataire*> représente toute information permettant de distinguer sans ambigüité une clef publique dans votre trousseau. Cette information peut-être le nom ou l'adresse email associé à la clef publique que vous voulez utiliser,
- <*message*> représente le message à chiffrer.

Par exemple pour chiffrer un message en mode binaire, il convient de saisir la commande suivante :

```
root@debian8:~# gpg --recipient I2TCH --encrypt message.txt

root@debian8:~# ls -l | grep message
-rw-r--r-- 1 root root       31 Aug  7 10:18 message.txt
-rw-r--r-- 1 root root      473 Aug 10 11:34 message.txt.asc
```

```
-rw-r--r-- 1 root root      367 Aug 10 11:37 message.txt.gpg
-rw-r--r-- 1 root root      287 Aug  7 10:18 message.txt.sig

root@debian8:~# cat message.txt.sig
□□□□□□□□W□□□
    □□_P2IQ□E□□□>□□)7□Lt□□□□C□+□vC□0□□cX□□□□□1N□Z□6□;□□□□□R□□□□Y□□>□gЧ□□□c□□□□T□iO|&□□TF□□□h□□□□u□□□□□V□
                              _ă□□Tf□□9ct□□□ě□□Q□i>□□ж□□□"K□c□□)□□□~□□□□*□u□f%□□DykheG/u□□_□K□'□□}G□H
                       □R^i□tЈь□y□`□[]a□_1□ժP)□a□□\□□□□ɘFO$m□□l
                                                                  3z□□□я□{□ҿe□*H□□w□H□N  □□□□x□ڳ□□
```

Et pour chiffrer un message en mode ascii, il convient de saisir la commande suivante :

```
root@debian8:~# gpg --recipient I2TCH --armor --encrypt message.txt
File `message.txt.asc' exists. Overwrite? (y/N) y

root@debian8:~# ls -l | grep message
-rw-r--r-- 1 root root       31 Aug  7 10:18 message.txt
-rw-r--r-- 1 root root      579 Aug 10 11:38 message.txt.asc
-rw-r--r-- 1 root root      367 Aug 10 11:37 message.txt.gpg
-rw-r--r-- 1 root root      287 Aug  7 10:18 message.txt.sig

root@debian8:~# cat message.txt.asc
-----BEGIN PGP MESSAGE-----
Version: GnuPG v1

hQEMA9hdNoMxWUPaAQgAk8S4GyQNVeB36uoLY0icuC/WXoxL5P7cJHXVnQuPQiBU
XJX3rHW0RCavc6mLTJvUGYUysubcVisQFWm3LTZ0ZD796S671EdLNNGgsfHNdR90
ext6f6UihR2ep5Y++8eTSat7YQc8nLiF5yexbnOCBuyxGI7gP2lLiZAX6sifmY61
pMegrrhA4BN2Wupbwm2A7WX4NcU4mdZtNxW+zC1Rtp+NOr6ad5JftEes2yPdUcrD
WW2gZaL1DJx50OmBc/tmUt1ea2VEtVDr1WDosD6dEWUbffDBA6wzr1DzUW488D0s
mYoVwu1bYSSSzmNGGvlFDa6EE/nHwVMhvgld1SB63tJeAbYgXKgEyKTfgIe/Byss
EyofLf5p+DVtJs1MK3OdJ87GV5n2XamtD3Qp3O2EvF/YW9e9aRmt6rF9jLSbIBiS
m4Ka3BM8p2yK9PQQbAvQIIyUg9TjP31c1bzdRa/VNQ==
```

```
=laaP
-----END PGP MESSAGE-----
```

Pour décrypter un message il convient d'utiliser la commande suivante :

```
root@debian8:~# gpg --decrypt message.txt.asc

You need a passphrase to unlock the secret key for
user: "I2TCH (Test Key) <infos@i2tch.eu>"
2048-bit RSA key, ID 315943DA, created 2016-08-07 (main key ID 324951F4)

gpg: encrypted with 2048-bit RSA key, ID 315943DA, created 2016-08-07
      "I2TCH (Test Key) <infos@i2tch.eu>"
This is a test message for gpg
```

PKI

On appelle **PKI** (Public Key Infrastucture, ou en français **infrastructure à clé publique (ICP)**, parfois **infrastructure de gestion de clés (IGC)**) l'ensemble des solutions techniques basées sur la cryptographie à clé publique.

Les cryptosystèmes à clés publiques permettent de s'affranchir de la nécessité d'avoir recours systématiquement à un canal sécurisé pour s'échanger les clés. En revanche, la publication de la clé publique à grande échelle doit se faire en toute confiance pour assurer que :

- La clé publique est bien celle de son propriétaire ;
- Le propriétaire de la clé est digne de confiance ;
- La clé est toujours valide.

Ainsi, il est nécessaire d'associer au bi-clé (ensemble clé publique / clé privée) un certificat délivré par un **tiers de confiance** : l'infrastructure de gestion de clés.

Le tiers de confiance est une entité appelée communément autorité de certification (ou en anglais Certification authority, abrégé CA) chargée d'assurer la véracité des informations contenues dans le certificat de clé publique et de sa validité.

Pour ce faire, l'autorité signe le certificat de clé publique à l'aide de sa propre clé en utilisant le principe de signature numérique.

Le rôle de l'infrastructure de clés publiques est multiple et couvre notamment les champs suivants :

- enregistrer des demandes de clés en vérifiant l'identité des demandeurs ;
- générer les paires de clés (clé privée / clé publique) ;
- garantir la confidentialité des clés privées correspondant aux clés publiques ;
- certifier l'association entre chaque utilisateurs et sa clé publique ;
- révoquer des clés (en cas de perte par son propriétaire, d'expiration de sa date de validité ou de compromission).

Une infrastructure à clé publique est en règle générale composée de trois entités distinctes :

- L'autorité d'enregistrement (AE ou RA pour Recording authority), chargée des formalité administratives telles que la vérification de l'identité des demandeurs, le suivi et la gestion des demandes, etc.) ;
- L'autorité de certification (AC ou CA pour Certification Authority), chargée des tâches techniques de création de certificats. L'autorité de certification est ainsi chargée de la signature des demandes de certificat (CSR pour Certificate Signing Request, parfois appelées PKCS#10, nom du format correspondant). L'autorité de certification a également pour mission la signature des listes de révocations (CRL pour Certificate Revocation List) ;
- L'Autorité de dépôt (Repository) dont la mission est de conserver en sécurité les certificats.

Certificats X509

Pour palier aux problèmes liés à des clefs publiques piratées, un système de certificats a été mis en place.

Le certificat permet d’associer la clef publique à une entité ou une personne. Les certificats sont délivrés par des Organismes de Certification.

Les certificats sont des fichiers divisés en deux parties :

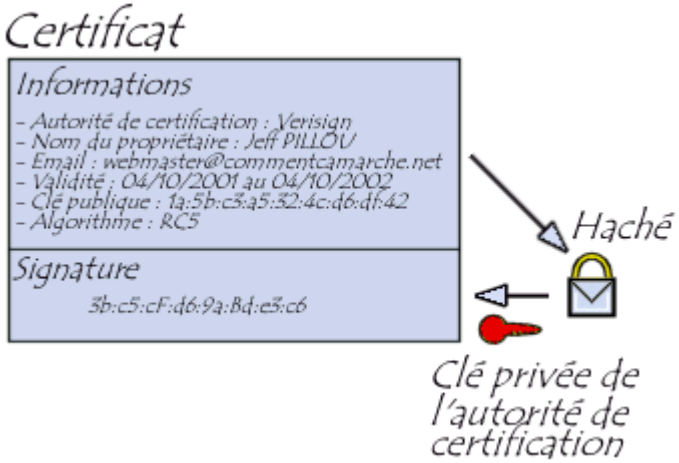
- La partie contenant les informations
- La partie contenant la signature de l'autorité de certification

La structure des certificats est normalisée par le standard **X.509** de l’**Union internationale des télécommunications**.

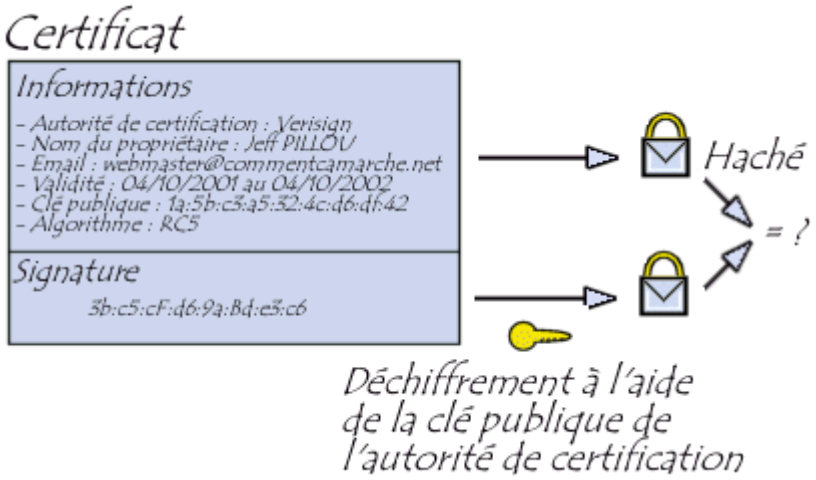
Elle contient :

- Le nom de l'autorité de certification
- Le nom du propriétaire du certificat
- La date de validité du certificat
- L'algorithme de chiffrement utilisé
- La clé publique du propriétaire

Le Certificat est signé par l'autorité de certification:



La vérification se passe ainsi:



Annexe 4 - La Commande iw

La commande **iw** permet de gérer le wifi :

```
root@debian11:~# iw dev wlo1 link
Connected to 00:5f:67:53:be:f4 (on wlo1)
    SSID: TP-Link_BEF5_5G
    freq: 5220
    RX: 3323698885 bytes (7423914 packets)
    TX: 1238020520 bytes (5751871 packets)
    signal: -30 dBm
    rx bitrate: 433.3 MBit/s VHT-MCS 9 80MHz short GI VHT-NSS 1
    tx bitrate: 433.3 MBit/s VHT-MCS 9 80MHz short GI VHT-NSS 1

    bss flags:    short-slot-time
    dtim period:    1
    beacon int:    100
root@debian11:~# iwconfig wlo1
```

```
wlo1      IEEE 802.11  ESSID:"TP-Link_BEF5_5G"
          Mode:Managed  Frequency:5.22 GHz  Access Point: 00:5F:67:53:BE:F4
          Bit Rate=433.3 Mb/s   Tx-Power=22 dBm
          Retry short limit:7   RTS thr:off   Fragment thr:off
          Encryption key:off
          Power Management:on
          Link Quality=70/70  Signal level=-30 dBm
          Rx invalid nwid:0  Rx invalid crypt:0  Rx invalid frag:0
          Tx excessive retries:0  Invalid misc:2   Missed beacon:0
```

Les options de cette commande sont :

```
root@debian11:~# iw --help
Usage:  iw [options] command
Options:
        --debug         enable netlink debugging
        --version       show version (5.9)
Commands:
        dev <devname> ap stop
                Stop AP functionality


        dev <devname> ap start
                <SSID> <control freq> [5|10|20|40|80|80+80|160] [<center1_freq> [<center2_freq>]] <beacon
interval in TU> <DTIM period> [hidden-ssid|zeroed-ssid] head <beacon head in hexadecimal> [tail <beacon tail in
hexadecimal>] [inactivity-time <inactivity time in seconds>] [key0:abcde d:1:6162636465]


        phy <phyname> coalesce show
                Show coalesce status.

        phy <phyname> coalesce disable
                Disable coalesce.
```

```
        phy <phyname> coalesce enable <config-file>
                Enable coalesce with given configuration.
                The configuration file contains coalesce rules:
                  delay=<delay>
                  condition=<condition>
                  patterns=<[offset1+]<pattern1>,<[offset2+]<pattern2>,...>
                  delay=<delay>
                  condition=<condition>
                  patterns=<[offset1+]<pattern1>,<[offset2+]<pattern2>,...>
                  ...
                delay: maximum coalescing delay in msec.
                condition: 1/0 i.e. 'not match'/'match' the patterns
                patterns: each pattern is given as a bytestring with '-' in
                places where any byte may be present, e.g. 00:11:22:-:44 will
                match 00:11:22:33:44 and 00:11:22:33:ff:44 etc. Offset and
                pattern should be separated by '+', e.g. 18+43:34:00:12 will
                match '43:34:00:12' after 18 bytes of offset in Rx packet.


        dev <devname> auth <SSID> <bssid> <type:open|shared> <freq in MHz> [key 0:abcde d:1:6162636465]
                Authenticate with the given network.


        dev <devname> connect [-w] <SSID> [<freq in MHz>] [<bssid>] [key 0:abcde d:1:6162636465] [mfp:req/opt/no]
                Join the network with the given SSID (and frequency, BSSID).
                With -w, wait for the connect to finish or fail.

        dev <devname> disconnect
                Disconnect from the current network.

        dev <devname> cqm rssi <threshold|off> [<hysteresis>]
                Set connection quality monitor RSSI threshold.
```

```
        event [-t|-r] [-f]
                Monitor events from the kernel.
                -t - print timestamp
                -r - print relative timestamp
                -f - print full frame for auth/assoc etc.

        dev <devname> ftm start_responder [lci=<lci buffer in hex>] [civic=<civic buffer in hex>]
                Start an FTM responder. Needs a running ap interface


        dev <devname> ftm get_stats
                Get FTM responder statistics.


        phy <phyname> hwsim wakequeues


        phy <phyname> hwsim stopqueues


        phy <phyname> hwsim setps <value>


        phy <phyname> hwsim getps


        dev <devname> ibss join <SSID> <freq in MHz> [NOHT|HT20|HT40+|HT40-|5MHz|10MHz|80MHz] [fixed-freq]
[<fixed bssid>] [beacon-interval <TU>] [basic-rates <rate in Mbps,rate2,...>] [mcast-rate <rate in Mbps>] [key
d:0:abcde]
                Join the IBSS cell with the given SSID, if it doesn't exist create
                it on the given frequency. When fixed frequency is requested, don't
                join/create a cell on a different frequency. When a fixed BSSID is
                requested use that BSSID and do not adopt another cell's BSSID even
                if it has higher TSF and the same SSID. If an IBSS is created, create
```

```
                it with the specified basic-rates, multicast-rate and beacon-interval.

        dev <devname> ibss leave
                Leave the current IBSS cell.

        features


        commands
                list all known commands and their decimal & hex value

        phy
        list
                List all wireless devices and their capabilities.

        phy <phyname> info
                Show capabilities for the specified wireless device.

        dev <devname> switch channel <channel> [NOHT|HT20|HT40+|HT40-|5MHz|10MHz|80MHz] [beacons <count>] [block-
tx]
        dev <devname> switch freq <freq> [NOHT|HT20|HT40+|HT40-|5MHz|10MHz|80MHz] [beacons <count>] [block-tx]
        dev <devname> switch freq <control freq> [5|10|20|40|80|80+80|160] [<center1_freq> [<center2_freq>]]
[beacons <count>] [block-tx]
                Switch the operating channel by sending a channel switch announcement (CSA).

        dev
                List all network interfaces for wireless hardware.

        dev <devname> info
                Show information for this interface.

        dev <devname> del
                Remove this virtual interface
```

```
        dev <devname> interface add <name> type <type> [mesh_id <meshid>] [4addr on|off] [flags <flag>*] [addr
<mac-addr>]
        phy <phyname> interface add <name> type <type> [mesh_id <meshid>] [4addr on|off] [flags <flag>*] [addr
<mac-addr>]
                Add a new virtual interface with the given configuration.
                Valid interface types are: managed, ibss, monitor, mesh, wds.

                The flags are only used for monitor interfaces, valid flags are:
                none:     no special flags
                fcsfail:  show frames with FCS errors
                control:  show control frames
                otherbss: show frames from other BSSes
                cook:     use cooked mode
                active:   use active mode (ACK incoming unicast packets)
                mumimo-groupid <GROUP_ID>: use MUMIMO according to a group id
                mumimo-follow-mac <MAC_ADDRESS>: use MUMIMO according to a MAC address

                The mesh_id is used only for mesh mode.

        help [command]
                Print usage for all or a specific command, e.g.
                "help wowlan" or "help wowlan enable".

        dev <devname> link
                Print information about the current link, if any.

        dev <devname> measurement ftm_request <config-file> [timeout=<seconds>] [randomise[=<addr>/<mask>]]
                Send an FTM request to the targets supplied in the config file.
                Each line in the file represents a target, with the following format:
                <addr> bw=<[20|40|80|80+80|160]> cf=<center_freq> [cf1=<center_freq1>] [cf2=<center_freq2>]
[ftms_per_burst=<samples per burst>] [ap-tsf] [asap] [bursts_exp=<num of bursts exponent>] [burst_period=<burst
period>] [retries=<num of retries>] [burst_duration=<burst duration>] [preamble=<legacy,ht,vht,dmg>] [lci]
[civic] [tb] [non_tb]
```

```
        dev <devname> mesh leave
                Leave a mesh.

        dev <devname> mesh join <mesh ID> [[freq <freq in MHz> <NOHT|HT20|HT40+|HT40-|80MHz>] [basic-rates <rate
in Mbps,rate2,...>]], [mcast-rate <rate in Mbps>] [beacon-interval <time in TUs>] [dtim-period <value>]
[vendor_sync on|off] [<param>=<value>]*
                Join a mesh with the given mesh ID with frequency, basic-rates,
                mcast-rate and mesh parameters. Basic-rates are applied only if
                frequency is provided.

        dev <devname> mgmt dump frame <type as hex ab> <pattern as hex ab:cd:..> [frame <type> <pattern>]* [count
<frames>]
                Register for receiving certain mgmt frames and print them.
                Frames are selected by their type and pattern containing
                the first several bytes of the frame that should match.

                Example: iw dev wlan0 mgmt dump frame 40 00 frame 40 01:02 count 10


        dev <devname> mpath dump
                List known mesh paths.

        dev <devname> mpath set <destination MAC address> next_hop <next hop MAC address>
                Set an existing mesh path's next hop.

        dev <devname> mpath new <destination MAC address> next_hop <next hop MAC address>
                Create a new mesh path (instead of relying on automatic discovery).

        dev <devname> mpath del <MAC address>
                Remove the mesh path to the given node.

        dev <devname> mpath get <MAC address>
                Get information on mesh path to the given node.
```

```
        dev <devname> mpath probe <destination MAC address> frame <frame>
                Inject ethernet frame to given peer overriding the next hop
                lookup from mpath table.
                .Example: iw dev wlan0 mpath probe xx:xx:xx:xx:xx:xx frame 01:xx:xx:00


        dev <devname> mpp dump
                List known mesh proxy paths.

        dev <devname> mpp get <MAC address>
                Get information on mesh proxy path to the given node.

        wdev <idx> nan add_func type <publish|subscribe|followup> [active] [solicited] [unsolicited] [bcast]
[close_range] name <name> [info <info>] [flw_up_id <id> flw_up_req_id <id> flw_up_dest <mac>] [ttl <ttl>] [srf
<include|exclude> <bf|list> [bf_idx] [bf_len] <mac1;mac2...>] [rx_filter <str1:str2...>] [tx_filter
<str1:str2...>]


        wdev <idx> nan rm_func cookie <cookie>


        wdev <idx> nan config [pref <pref>] [bands [2GHz] [5GHz]]


        wdev <idx> nan stop


        wdev <idx> nan start pref <pref> [bands [2GHz] [5GHz]]


        dev <devname> ocb leave
                Leave the OCB mode network.

        dev <devname> ocb join <freq in MHz> <5MHz|10MHz>
```

```
                Join the OCB mode network.

        dev <devname> offchannel <freq> <duration>
                Leave operating channel and go to the given channel for a while.

        wdev <idx> p2p stop


        wdev <idx> p2p start


        dev <devname> cac channel <channel> [NOHT|HT20|HT40+|HT40-|5MHz|10MHz|80MHz]
        dev <devname> cac freq <freq> [NOHT|HT20|HT40+|HT40-|5MHz|10MHz|80MHz]
        dev <devname> cac freq <control freq> [5|10|20|40|80|80+80|160] [<center1_freq> [<center2_freq>]]
        dev <devname> cac trigger channel <channel> [NOHT|HT20|HT40+|HT40-|5MHz|10MHz|80MHz]
        dev <devname> cac trigger freq <frequency> [NOHT|HT20|HT40+|HT40-|5MHz|10MHz|80MHz]
        dev <devname> cac trigger freq <frequency> [5|10|20|40|80|80+80|160] [<center1_freq> [<center2_freq>]]
                Start or trigger a channel availability check (CAC) looking to look for
                radars on the given channel.

        phy <phyname> channels
                Show available channels.

        reg reload
                Reload the kernel's regulatory database.

        phy <phyname> reg get
                Print out the devices' current regulatory domain information.

        reg get
                Print out the kernel's current regulatory domain information.

        reg set <ISO/IEC 3166-1 alpha2>
                Notify the kernel about the current regulatory domain.
```

```
        dev <devname> roc start <freq> <time in ms>


        dev <devname> scan [-u] [freq <freq>*] [duration <dur>] [ies <hex as 00:11:..>] [meshid <meshid>]
[lowpri,flush,ap-force,duration-mandatory] [randomise[=<addr>/<mask>]] [ssid <ssid>*|passive]
                Scan on the given frequencies and probe for the given SSIDs
                (or wildcard if not given) unless passive scanning is requested.
                If -u is specified print unknown data in the scan results.
                Specified (vendor) IEs must be well-formed.

        dev <devname> scan sched_stop
                Stop an ongoing scheduled scan.

        dev <devname> scan sched_start [interval <in_msecs> | scan_plans [<interval_secs:iterations>*]
<interval_secs>] [delay <in_secs>] [freqs <freq>+] [matches [ssid <ssid>]+]] [active [ssid <ssid>]+|passive]
[randomise[=<addr>/<mask>]]
                Start a scheduled scan at the specified interval on the given frequencies
                with probing for the given SSIDs (or wildcard if not given) unless passive
                scanning is requested.  If matches are specified, only matching results
                will be returned.

        dev <devname> scan abort
                Abort ongoing scan

        dev <devname> scan trigger [freq <freq>*] [duration <dur>] [ies <hex as 00:11:..>] [meshid <meshid>]
[lowpri,flush,ap-force,duration-mandatory] [randomise[=<addr>/<mask>]] [ssid <ssid>*|passive]
                Trigger a scan on the given frequencies with probing for the given
                SSIDs (or wildcard if not given) unless passive scanning is requested.
                Duration(in TUs), if specified, will be used to set dwell times.


        dev <devname> scan dump [-u]
                Dump the current scan results. If -u is specified, print unknown
                data in scan results.
```

```
        dev <devname> set bitrates [legacy-<2.4|5> <legacy rate in Mbps>*] [ht-mcs-<2.4|5> <MCS index>*] [vht-
mcs-<2.4|5> [he-mcs-<2.4|5|6> <NSS:MCSx,MCSy... | NSS:MCSx-MCSy>*] [sgi-2.4|lgi-2.4] [sgi-5|lgi-5] [he-gi-
<2.4|5|6> <0.8|1.6|3.2>] [he-ltf-<2.4|5|6> <1|2|4>]
                Sets up the specified rate masks.
                Not passing any arguments would clear the existing mask (if any).

        dev <devname> set tidconf [peer <MAC address>] tids <mask> [override] [sretry <num>] [lretry <num>]
[ampdu [on|off]] [amsdu [on|off]] [noack [on|off]] [rtscts [on|off]][bitrates <type [auto|fixed|limit]> [legacy-
<2.4|5> <legacy rate in Mbps>*] [ht-mcs-<2.4|5> <MCS index>*] [vht-mcs-<2.4|5> <NSS:MCSx,MCSy... | NSS:MCSx-
MCSy>*] [sgi-2.4|lgi-2.4] [sgi-5|lgi-5]]
                Setup per-node TID specific configuration for TIDs selected by bitmask.
                If MAC address is not specified, then supplied TID configuration
                applied to all the peers.
                Examples:
                  $ iw dev wlan0 set tidconf tids 0x1 ampdu off
                  $ iw dev wlan0 set tidconf tids 0x5 ampdu off amsdu off rtscts on
                  $ iw dev wlan0 set tidconf tids 0x3 override ampdu on noack on rtscts on
                  $ iw dev wlan0 set tidconf peer xx:xx:xx:xx:xx:xx tids 0x1 ampdu off tids 0x3 amsdu off rtscts
on
                  $ iw dev wlan0 set tidconf peer xx:xx:xx:xx:xx:xx tids 0x2 bitrates auto
                  $ iw dev wlan0 set tidconf peer xx:xx:xx:xx:xx:xx tids 0x2 bitrates limit vht-mcs-5 4:9


        dev <devname> set mcast_rate <rate in Mbps>
                Set the multicast bitrate.

        dev <devname> set peer <MAC address>
                Set interface WDS peer.

        dev <devname> set noack_map <map>
                Set the NoAck map for the TIDs. (0x0009 = BE, 0x0006 = BK, 0x0030 = VI, 0x00C0 = VO)

        dev <devname> set 4addr <on|off>
                Set interface 4addr (WDS) mode.
```

```
        dev <devname> set type <type>
                Set interface type/mode.
                Valid interface types are: managed, ibss, monitor, mesh, wds.

        dev <devname> set meshid <meshid>
        dev <devname> set monitor <flag>*
                Set monitor flags. Valid flags are:
                none:     no special flags
                fcsfail:  show frames with FCS errors
                control:  show control frames
                otherbss: show frames from other BSSes
                cook:     use cooked mode
                active:   use active mode (ACK incoming unicast packets)
                mumimo-groupid <GROUP_ID>: use MUMIMO according to a group id
                mumimo-follow-mac <MAC_ADDRESS>: use MUMIMO according to a MAC address

        dev <devname> set mesh_param <param>=<value> [<param>=<value>]*
                Set mesh parameter (run command without any to see available ones).

        phy <phyname> set txq limit <packets> | memory_limit <bytes> | quantum <bytes>
                Set TXQ parameters. The limit and memory_limit are global queue limits
                for the whole phy. The quantum is the DRR scheduler quantum setting.
                Valid values: 1 - 2**32

        phy <phyname> set antenna <bitmap> | all | <tx bitmap> <rx bitmap>
                Set a bitmap of allowed antennas to use for TX and RX.
                The driver may reject antenna configurations it cannot support.

        dev <devname> set txpower <auto|fixed|limit> [<tx power in mBm>]
                Specify transmit power level and setting type.

        phy <phyname> set txpower <auto|fixed|limit> [<tx power in mBm>]
                Specify transmit power level and setting type.
```

```
        phy <phyname> set distance <auto|distance>
                Enable ACK timeout estimation algorithm (dynack) or set appropriate
                coverage class for given link distance in meters.
                To disable dynack set valid value for coverage class.
                Valid values: 0 - 114750

        phy <phyname> set coverage <coverage class>
                Set coverage class (1 for every 3 usec of air propagation time).
                Valid values: 0 - 255.

        phy <phyname> set netns { <pid> | name <nsname> }
                Put this wireless device into a different network namespace:
                    <pid>    - change network namespace by process id
                    <nsname> - change network namespace by name from /run/netns
                               or by absolute path (man ip-netns)


        phy <phyname> set retry [short <limit>] [long <limit>]
                Set retry limit.

        phy <phyname> set rts <rts threshold|off>
                Set rts threshold.

        phy <phyname> set frag <fragmentation threshold|off>
                Set fragmentation threshold.

        dev <devname> set channel <channel> [NOHT|HT20|HT40+|HT40-|5MHz|10MHz|80MHz]
        phy <phyname> set channel <channel> [NOHT|HT20|HT40+|HT40-|5MHz|10MHz|80MHz]
        dev <devname> set freq <freq> [NOHT|HT20|HT40+|HT40-|5MHz|10MHz|80MHz]
        dev <devname> set freq <control freq> [5|10|20|40|80|80+80|160] [<center1_freq> [<center2_freq>]]
        phy <phyname> set freq <freq> [NOHT|HT20|HT40+|HT40-|5MHz|10MHz|80MHz]
        phy <phyname> set freq <control freq> [5|10|20|40|80|80+80|160] [<center1_freq> [<center2_freq>]]
                Set frequency/channel the hardware is using, including HT
                configuration.
```

```
        phy <phyname> set name <new name>
                Rename this wireless device.

        dev <devname> set power_save <on|off>
                Set power save state to on or off.

        dev <devname> get mesh_param [<param>]
                Retrieve mesh parameter (run command without any to see available ones).

        phy <phyname> get txq
                Get TXQ parameters.

        dev <devname> get power_save <param>
                Retrieve power save state.

        dev <devname> station dump [-v]
                List all stations known, e.g. the AP on managed interfaces

        dev <devname> station set <MAC address> txpwr <auto|limit> [<tx power dBm>]
                Set Tx power for this station.

        dev <devname> station set <MAC address> airtime_weight <weight>
                Set airtime weight for this station.

        dev <devname> station set <MAC address> mesh_power_mode <active|light|deep>
                Set link-specific mesh power mode for this station

        dev <devname> station set <MAC address> vlan <ifindex>
                Set an AP VLAN for this station.

        dev <devname> station set <MAC address> plink_action <open|block>
                Set mesh peer link action for this station (peer).

        dev <devname> station del <MAC address> [subtype <subtype>] [reason-code <code>]
```

```
                Remove the given station entry (use with caution!)
                Example subtype values: 0xA (disassociation), 0xC (deauthentication)

        dev <devname> station get <MAC address>
                Get information for a specific station.

        dev <devname> survey dump
                List all gathered channel survey data

        dev <devname> vendor recvbin <oui> <subcmd> <filename|-|hex data>


        dev <devname> vendor recv <oui> <subcmd> <filename|-|hex data>


        dev <devname> vendor send <oui> <subcmd> <filename|-|hex data>


        phy <phyname> wowlan show
                Show WoWLAN status.

        phy <phyname> wowlan disable
                Disable WoWLAN.

        phy <phyname> wowlan enable [any] [disconnect] [magic-packet] [gtk-rekey-failure] [eap-identity-request]
[4way-handshake] [rfkill-release] [net-detect [interval <in_msecs> | scan_plans [<interval_secs:iterations>*]
<interval_secs>] [delay <in_secs>] [freqs <freq>+] [matches [ssid <ssid>]+]] [active [ssid <ssid>]+|passive]
[randomise[=<addr>/<mask>]]] [tcp <config-file>] [patterns [offset1+]<pattern1> ...]
                Enable WoWLAN with the given triggers.
                Each pattern is given as a bytestring with '-' in places where any byte
                may be present, e.g. 00:11:22:-:44 will match 00:11:22:33:44 and
                00:11:22:33:ff:44 etc.
                Offset and pattern should be separated by '+', e.g. 18+43:34:00:12 will match '43:34:00:12' after
18 bytes of offset in Rx packet.
```

```
                The TCP configuration file contains:
                  source=ip[:port]
                  dest=ip:port@mac
                  data=<hex data packet>
                  data.interval=seconds
                  [wake=<hex packet with masked out bytes indicated by '-'>]
                  [data.seq=len,offset[,start]]
                  [data.tok=len,offset,<token stream>]

                Net-detect configuration example:
                 iw phy0 wowlan enable net-detect interval 5000 delay 30 freqs 2412 2422 matches ssid foo ssid
bar


Commands that use the netdev ('dev') can also be given the
'wdev' instead to identify the device.

You can omit the 'phy' or 'dev' if the identification is unique,
e.g. "iw wlan0 info" or "iw phy0 info". (Don't when scripting.)

Do NOT screenscrape this tool, we don't consider its output stable.
```
