

Administration

Linux

PREVIOUSLY ON

GAME OF THRONES

Previously on Games of Codes

- Découverte de Linux
- La ligne de commande
- Des fichiers
- Des utilisateurs et des permissions
- Des processus
- Assembler des commandes
- Écriture de script

Rappels

Utilisez [Tab], ↑ ↓, et Ctrl+A/E !

Soyez attentif à ce que vous tapez et à ce que la machine vous renvoie

Cette semaine

- installer et gérer une distribution
- acquérir des bases de réseau et de sécurité
- administrer un serveur à distance
- configurer et gérer des services
- déployer un serveur web / des apps web

Plan

1. Installer une distribution, gérer les partitions
2. Le gestionnaire de paquet (et les archives)
3. Notions de réseau
4. Notions de cryptographie
5. Se connecter et gérer un serveur avec SSH
6. Services et sécurité basique d'un serveur
7. Déployer un site "basique" avec nginx
8. Déployer une "vrai" app : Nextcloud ?

1. Installer une distribution

et gérer les partitions

1. Installer une distribution

Installons un système Linux nous-même ... et au passage, choisissons un environnement graphique (ou bien si vous ne voulez pas choisir : gardez Cinnamon)

1. Installer une distribution

Fonctionnement de l'environnement graphique : Xorg

C'est le serveur graphique (qui commence à être remplacée par Wayland ?)

Il fonctionne en client/serveur

Et un autre morceau : le window manager

Qui s'occupe de toute la gestion des fenêtres (bordures, décoration, redimensionnement, minimisation, vignette, ...)

1. Installer une distribution

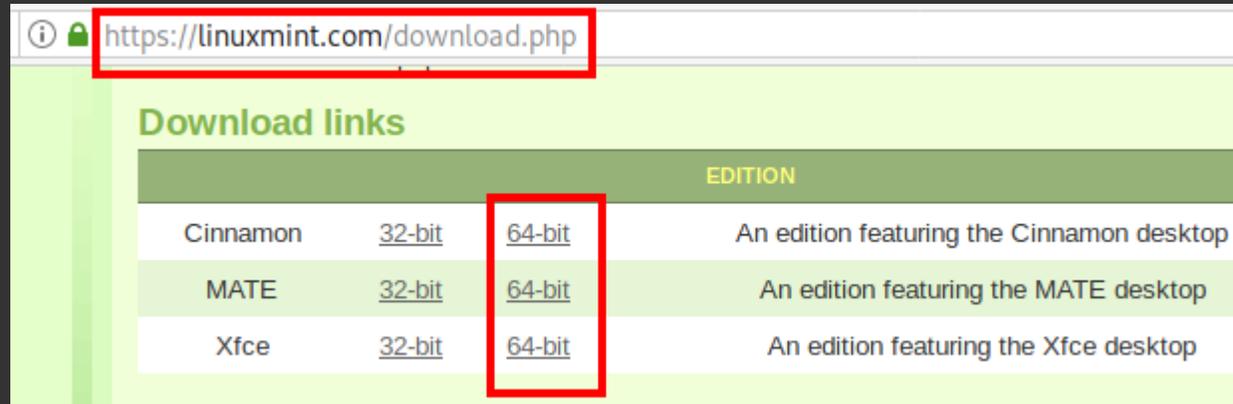
Procédure d'installation générale

(Prerequis : avoir accès au BIOS du système (et avoir de la place))

- Télécharger et flasher une "Live CD/USB"
- Dire au BIOS de booter sur la "Live CD/USB"
- Lancer l'installation
 - (définir un plan de partitionnement)
- Prendre un café
- Rebooter et vérifier que ça a fonctionné

1. Installer une distribution

Telecharger l'ISO



The screenshot shows a web browser window with the URL <https://linuxmint.com/download.php> highlighted in red. Below the URL, the page title "Download links" is visible. A table lists three desktop editions: Cinnamon, MATE, and Xfce. Each edition has two columns for "32-bit" and "64-bit" download links. The "64-bit" links for all three editions are highlighted with a red box.

		EDITION		
Cinnamon	32-bit	64-bit		An edition featuring the Cinnamon desktop
MATE	32-bit	64-bit		An edition featuring the MATE desktop
Xfce	32-bit	64-bit		An edition featuring the Xfce desktop

1. Installer une distribution

Vérifier l'intégrité / authenticité



Sous Linux: `sha256sum <fichier>` directement disponible

Sous Windows: ... il faut trouver un `sha256sum.exe`

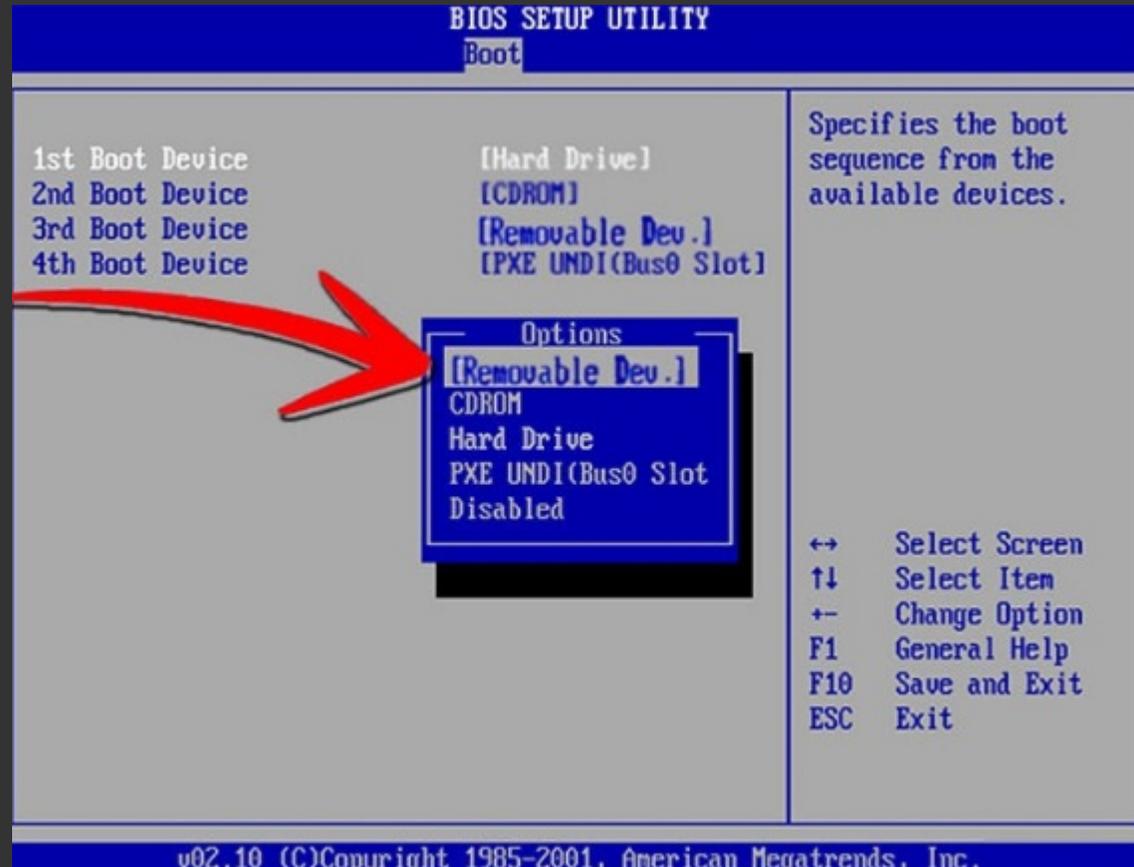
1. Installer une distribution

Le BIOS

- Programme lancé par la machine à son démarrage
- Change entre les modèles de PC ...
- Gère différents aspects "bas-niveau" (e.g. horloge intégrée)
- Gère le lancement du "vrai" système d'exploitation
 - analyse typiquement le lecteur CD
 - ... puis le HDD
 - ... puis le network (PXE)
 - ...
- De nos jours, l'UEFI et Secure boot compliquent beaucoup les choses ...

1. Installer une distribution

Le BIOS



1. Installer une distribution

Live CD/USB

- Un système généralement "éphémère" (données perdues)
- Typiquement sur un CD rom ou une clef USB
- Système entièrement chargé dans la RAM (performances moindres)
- Destiné à tester / faire une démo du système et à l'installer
- Permet aussi d'avoir accès à certains outils
- Généralement sous forme d'un fichier `.iso`



1. Installer une distribution

Lancer l'installation

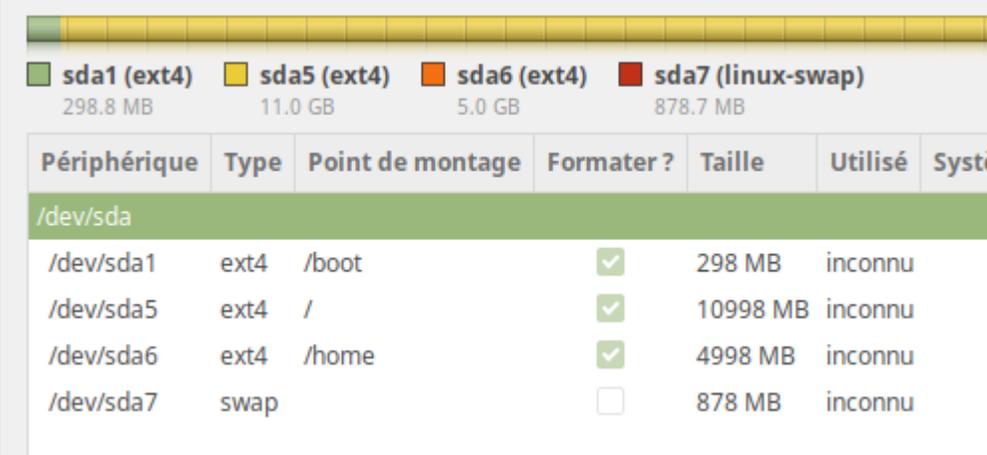
1. Installer une distribution

Lancer l'installation

1. Installer une distribution

Plan de partitionnement (exemple!)

- 300 Mo pour `/boot/` en ext4
- 12 Go pour `/` en ext4
- 3 Go pour `/home/` en ext4
- Le reste en swap (une extension "lente" de la RAM)



The screenshot shows a disk partitioning interface for `/dev/sda`. At the top, a legend identifies the partitions: `sda1 (ext4)` (298.8 MB), `sda5 (ext4)` (11.0 GB), `sda6 (ext4)` (5.0 GB), and `sda7 (linux-swap)` (878.7 MB). Below this is a table with columns for 'Périphérique', 'Type', 'Point de montage', 'Formater?', 'Taille', 'Utilisé', and 'Système'. The table lists the following partitions:

Périphérique	Type	Point de montage	Formater?	Taille	Utilisé	Système
<code>/dev/sda</code>						
<code>/dev/sda1</code>	ext4	<code>/boot</code>	<input checked="" type="checkbox"/>	298 MB	inconnu	
<code>/dev/sda5</code>	ext4	<code>/</code>	<input checked="" type="checkbox"/>	10998 MB	inconnu	
<code>/dev/sda6</code>	ext4	<code>/home</code>	<input checked="" type="checkbox"/>	4998 MB	inconnu	
<code>/dev/sda7</code>	swap		<input type="checkbox"/>	878 MB	inconnu	

1. Installer une distribution

Lancer l'installation "pour de vrai"

- Répondre aux questions pour créer l'utilisateur, etc...
- ... le système s'installe ...

1. Installer une distribution

Finir l'installation

- Redémarrer
- (Enlever le média d'installation)
- (Dire au BIOS de booter de nouveau sur le HDD)

1. Installer une distribution

GRUB

1. Installer une distribution

GRUB

1. Installer une distribution

Résumé du boot complet (du Bios à l'interface de login)

BIOS	Basic Input/Output System executes MBR
MBR	Master Boot Record executes GRUB
GRUB	Grand Unified Bootloader executes Kernel thegeekstuff.com
Kernel	Kernel executes /sbin/init
Init	Init executes runlevel programs
Runlevel	Runlevel programs are executed from /etc/rc.d/rc*.d/

1. Installer une distribution

Log du boot

- Les logs du boot du kernel (contient aussi par ex. le log de la détection de dispositif USB branchés après le boot, etc...) peuvent être trouvés dans `/var/log/dmesg`

1. Installer une distribution

Init levels / Run levels

- 0 = Shutdown
- 1 = Single-user mode : Mode for administrative tasks
- 2 = Multi-user mode, without network interfaces
- 3 = Multi-user mode with networking
- 4 = ... not used ...
- 5 = Multi-user with networking and graphical environment
- 6 = Reboot

**Sous SysVinit, choses à lancées décrites dans /etc/rc.d/rcX.d/...
mais aujourd'hui : c'est différent avec systemd...**

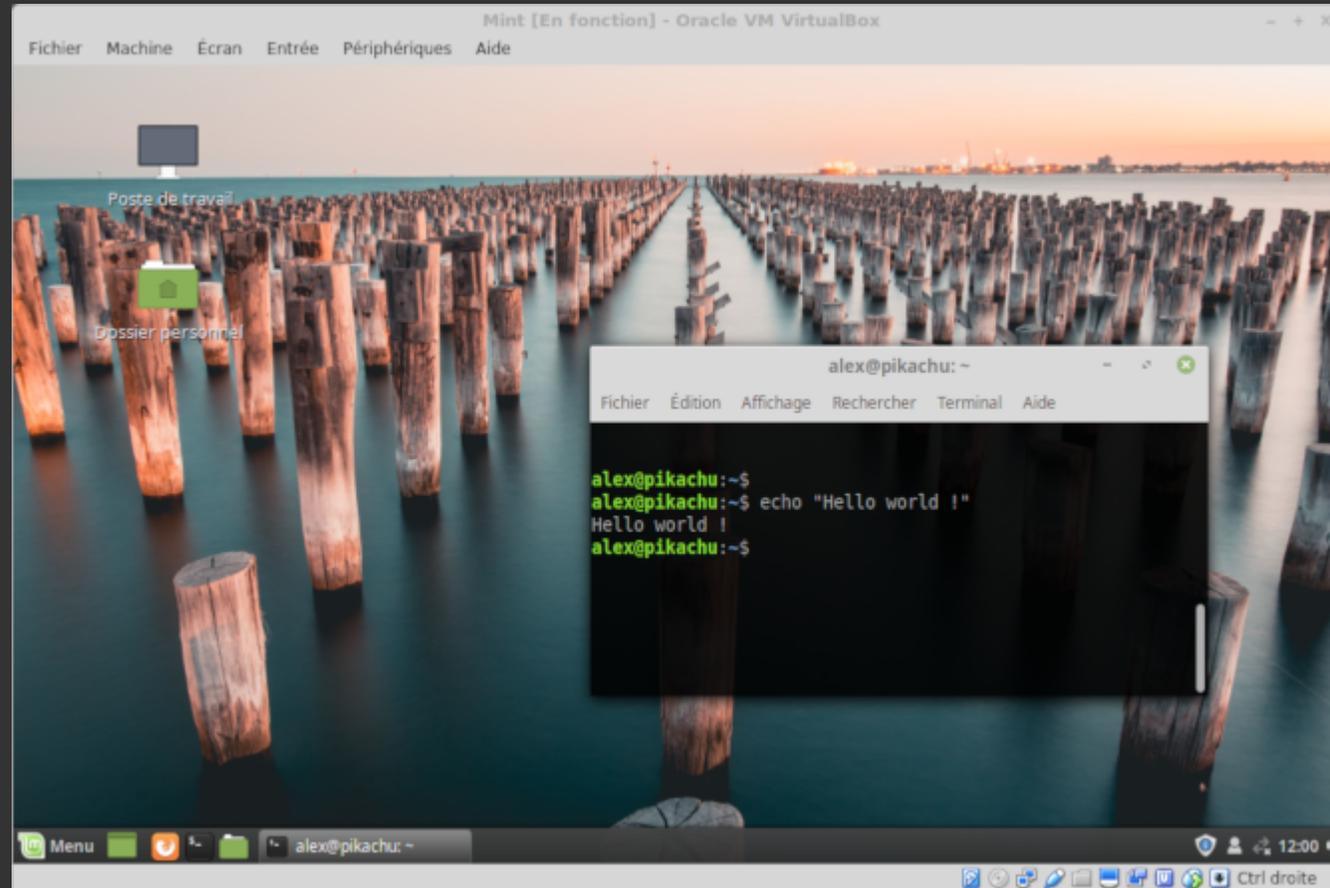
1. Installer une distribution

Login



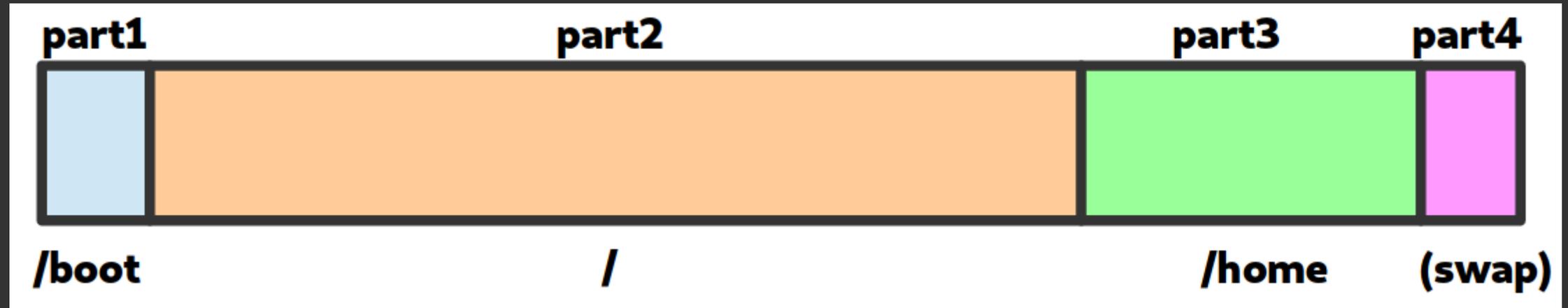
1. Installer une distribution

Le bureau



1. Installer une distribution

Notation des partitions



1. Installer une distribution

Notation des partitions

Les disques partitions sous Linux sont généralement dénommées :

- `/dev/sda` (premier disque)
 - `/dev/sda1` (première partition de `/dev/sda`)
 - `/dev/sda2` (deuxième partition de `/dev/sda`)
- `/dev/sdb` (deuxième disque)
 - `/dev/sdb1` (première partition de `/dev/sdb`)
 - `/dev/sdb2` (deuxième partition de `/dev/sdb`)
 - `/dev/sdb3` (troisième partition de `/dev/sdb`)

1. Installer une distribution

Outil pour lister les disques, gérer les partitions

```
$ fdisk -l
Disk /dev/sda: 29.8 GiB, 32017047552 bytes, 62533296 sectors
[...]
Device          Start      End  Sectors  Size Type
/dev/sda1       2048    2099199 2097152   1G Linux filesystem
/dev/sda2     2099200 62524946 60425747 28.8G Linux filesystem
```

1. Installer une distribution

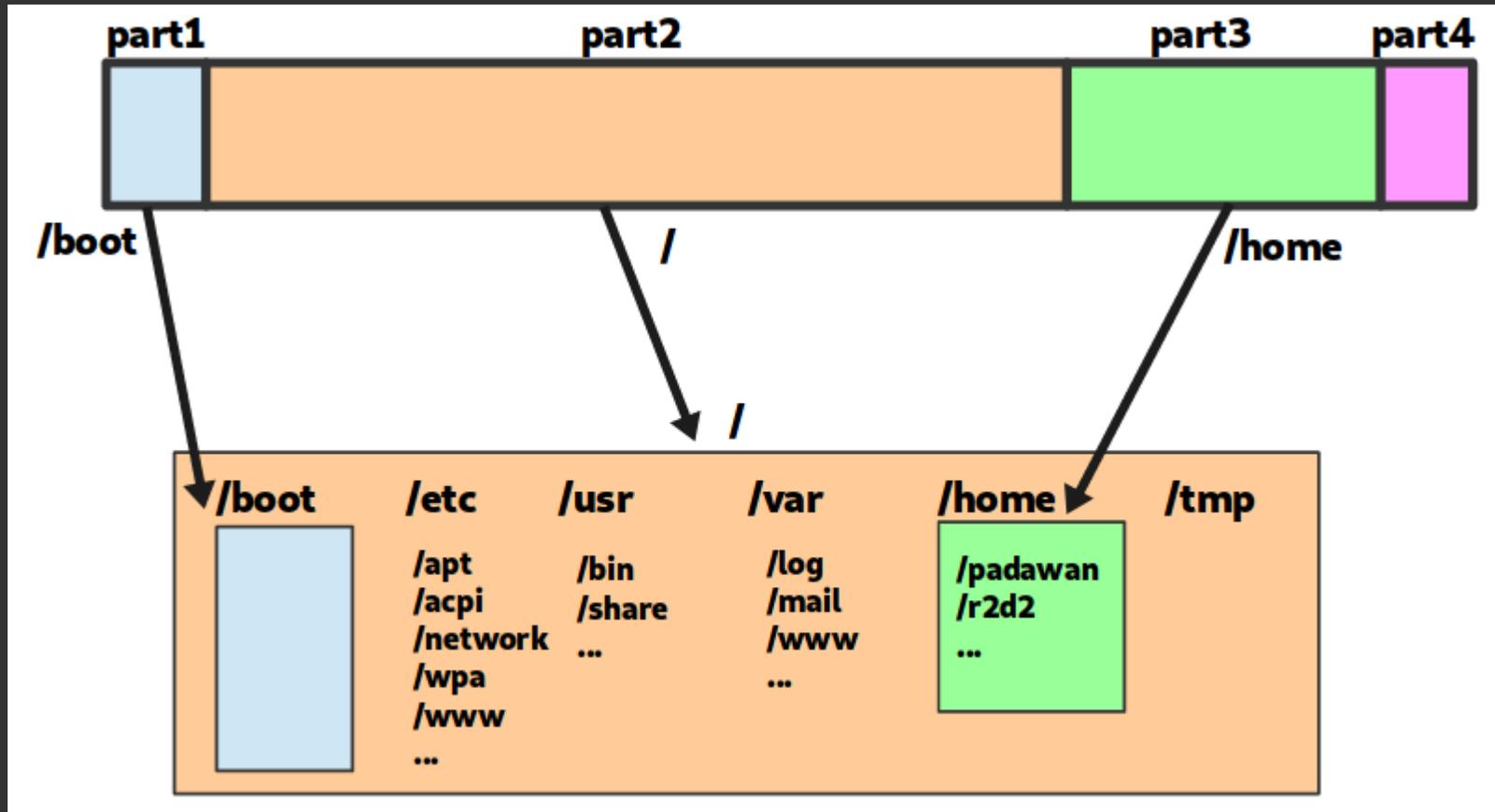
Les points de montage

Une partition ou n'importe quel "bidule de stockage" peut être "monté" dans le système de fichier

- partition
- clef usb
- image iso
- stockage distant
- ...

1. Installer une distribution

Les points de montage



1. Installer une distribution

Les points de montage

Les points de montages sont gérés avec `mount`

```
$ mkdir /media/usbkey  
$ mount /dev/sdb1 /media/usbkey  
$ ls /media/usbkey  
# [le contenu de la clef usb s'affiche]
```

1. Installer une distribution

Les points de montage

On peut "démonter" un element monté avec `umount`

```
$ umount /media/usbkey
```

1. Installer une distribution

Les points de montage : `/etc/fstab`

`/etc/fstab` décrit les systèmes de fichier montés automatiquement au boot

```
# <file system>      <mountpoint> <type>  <options>      <dump>  <pass>
UUID=[id tres long] /                ext4      default         0         1
UUID=[id tres long] /home/          ext4      defaults        0         2
```

(historiquement, la première colonne contenait `/dev/sdxY`, mais les UUID sont plus robustes)

1. Installer une distribution

Les points de montage : outils

Juste `mount` permet aussi de lister les différents points de montage

```
$ mount  
[...]  
/dev/sda1 on /boot type ext4 (rw,noatime,discard,data=ordered)  
/dev/sda2 on / type ext4 (rw,noatime,discard,data=ordered)
```

1. Installer une distribution

Les points de montage : outils

Il existe aussi `df` :

```
$ df -h
Filesystem      Size  Used Avail Use% Mounted on
dev             2.8G   0    2.8G   0% /dev
run             2.8G  1.1M  2.8G   1% /run
/dev/dm-0       29G   22G   5.0G  82% /
tmpfs           2.8G   22M   2.8G   1% /dev/shm
tmpfs           2.8G   0    2.8G   0% /sys/fs/cgroup
tmpfs           2.8G  1.9M  2.8G   1% /tmp
/dev/sda1       976M  105M  804M  12% /boot
tmpfs           567M   16K   567M   1% /run/user/1000
```

1. Installer une distribution

Les points de montage : outils

Et aussi `lsblk` :

```
$ lsblk
NAME                MAJ:MIN RM  SIZE RO TYPE  MOUNTPOINT
sda                  8:0    0 29.8G  0 disk
├─sda1                8:1    0   1G  0 part  /boot
└─sda2                8:2    0 28.8G  0 part  /
```

1. Installer une distribution

Autres configurations du système (avec systemd)

- `hostnamectl`
- `timedatectl`
- `localectl`

2. Le gestionnaire de paquet

(et les archives)

2. Le gestionnaire de paquet

Motivation

Historiquement, c'est très compliqué d'installer un programme :

- le télécharger et le compiler
- la compilation (ou le programme lui-même) requiert des dépendances
- il faut télécharger et compiler les dépendances
- qui requiert elles-mêmes des dépendances ...

Paquet =~ programmes ou librairies

2. Le gestionnaire de paquet

Le travail d'une distribution (entre autre)

- créer et maintenir un ensemble de paquet cohérents
- ... et le gestionnaire de paquet qui va avec
- les (pre)compiler pour fournir des binaires

2. Le gestionnaire de paquet

Le gestionnaire de paquet c'est :

- La "clef de voute" d'une distribution ?
- un **système unifié pour installer** des paquets ...
- ... **et les mettre à jour !**
- le tout en gérant les dépendances et les conflits
- et via une communauté qui s'assure que les logiciels ne font pas n'importe quoi.

2. Le gestionnaire de paquet

Comparaison avec Windows

Sous Windows

- téléchargement d'un .exe par l'utilisateur ...
- ... depuis une source obscure ! (**critical security risk !**)
- procédure d'installation spécifique
- ... qui tente de vous refiler des toolbar bloated, et/ou des CGU obscures
- système de mise à jour spécifique
- nécessité d'installer manuellement des dépendances

2. Le gestionnaire de paquet

One package to rule them all

One package to find them

One package to download them all

and on the system bind them

In the land of GNU/Debian where the penguin lie

2. Le gestionnaire de paquet

Sous Debian

`apt` : couche "haut niveau"

- dépôt,
- authentification,
- ...

`dpkg` : couche "bas niveau"

- gestion des dépendances,
- installation du paquet (`.deb`),
- ...

2. Le gestionnaire de paquet

Parenthèse sur `apt-get`

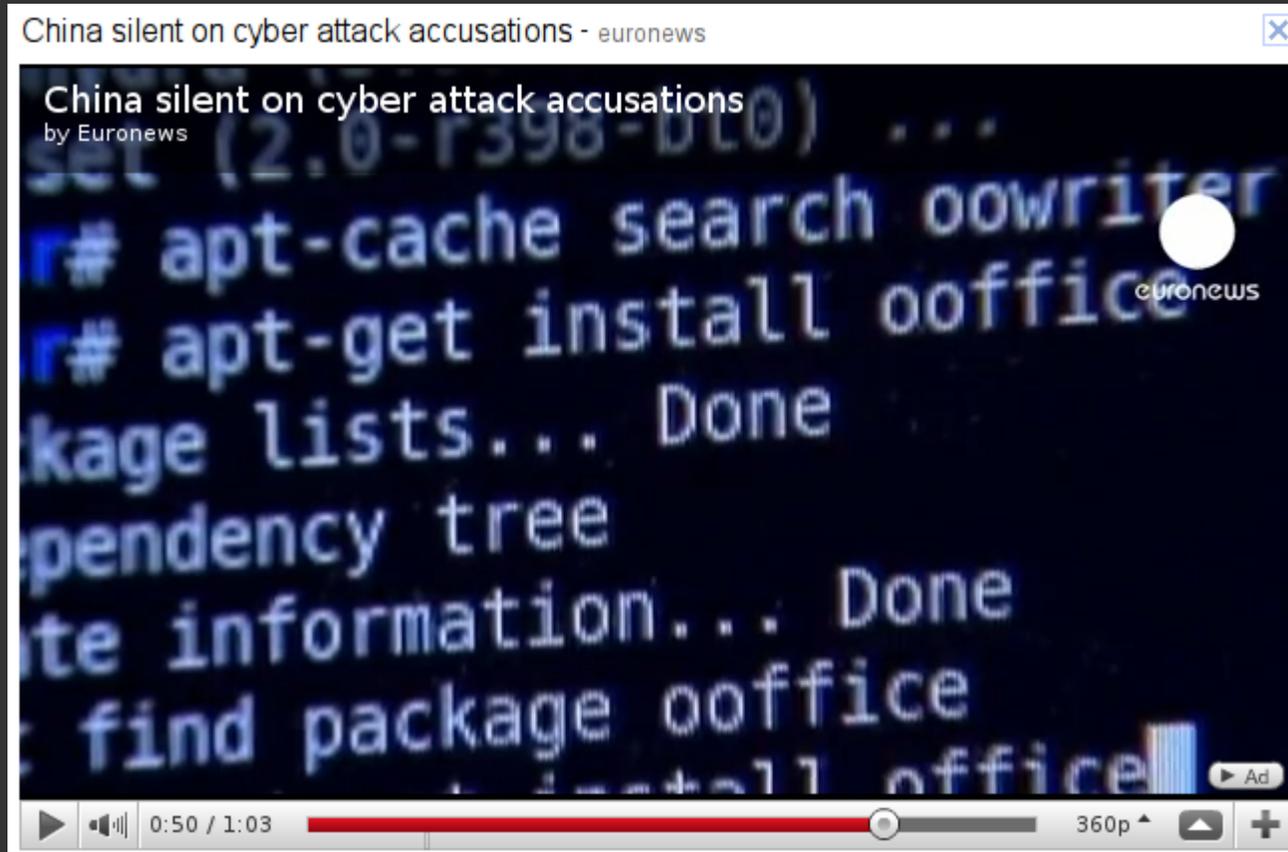
- Historiquement, `apt-get` (et `apt-cache`, `apt-mark`, ..) étaient utilisés
- Syntaxe inutilement complexe ?
- `apt` fourni une meilleur interface (UI et UX)

2. Le gestionnaire de paquet

Utilisation de `apt`

- `apt install <package>`
 - télécharge et installe le paquet et tout son arbre de dépendances
- `apt remove <package>`
 - désinstaller le paquet (et les paquet dont il dépends !)
- `apt autoremove`
 - supprime les paquets qui ne sont plus nécessaires

2. Le gestionnaire de paquet



2. Le gestionnaire de paquet

Mais qu'est-ce que c'est, un paquet ?

Un programme, et des fichiers (dossier `debian/`) qui décrivent le paquet :

- `control` : décrit le paquet et ses dépendances
- `install` : liste des fichiers et leur destination
- `changelog` : un historique de l'évolution du paquet
- `rules` : des trucs techniques pour compiler le paquet
- `postinst`, `prerm`, ... : des scripts lancés quand le paquet est installé, désinstallé, ...

2. Le gestionnaire de paquet

Mettre à jour les paquets

- `apt update`
 - récupère la liste des paquets depuis les dépôts
- `apt full-upgrade`
 - calcule et lance la mise à jour de tous les paquets
 - (anciennement appelé : `apt dist-upgrade`)
- Moins utilisé : `apt upgrade`
 - mise à jour "safe", sans installer/supprimer de nouveaux paquets
 - en général, `full-upgrade` est okay

2. Le gestionnaire de paquet

N.B. : pour les moldus dans la vraie vie, il y a des interfaces graphiques pour gérer tout ça sans ligne de commande, mais ici on présente les détails techniques

2. Le gestionnaire de paquet

Les dépôts

Les dépôts de paquets sont configurés via `/etc/apt/sources.list` et les fichiers du dossier `/etc/apt/sources.list.d/`.

Exemple :

```
deb http://ftp.debian.fr/debian/ stretch main contrib
```

- `stretch` est le nom actuel de la distribution
- `main` et `contrib` sont des composantes à utiliser
- le protocole est `http` ... l'authenticité des paquets est géré par un autre mécanisme (GPG)

2. Le gestionnaire de paquet

Les versions de Debian

Debian vise un système libre et très stable

- **stable** : paquets éprouvés et très stable (bien que souvent un peu vieux)
- **testing** : paquets en cours de test, comportant encore quelques bugs
- **unstable** (sid) : pour les gens qui aiment vivre dangereusement

Les versions tournent tous les ~2 ans environ

- l'ancienne **testing** devient la nouvelle **stable**
- le passage de version peut être un peu douloureux ... (quoiqu'en vrai c'est de + en + smooth)

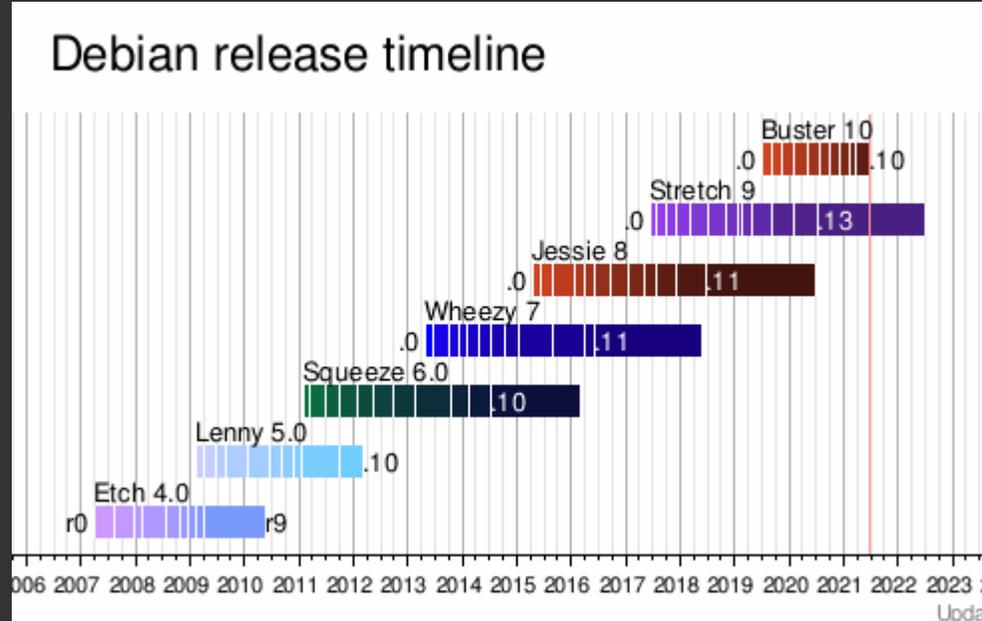
2. Le gestionnaire de paquet

Les versions de Debian

Basé sur les personnages de Toy Story

- 7, wheezy (oldoldstable)
- 8, jessie (oldoldstable)
- 9, stretch (oldstable, depuis juillet 2019)
- 10, buster (**stable, depuis juillet 2019**)
- 11, bullseye (testing, deviendra stable fin juillet 2021)
- 12, bookworm (future testing, stable en été 2023 ?)

2. Le gestionnaire de paquet



2. Le gestionnaire de paquet

Naviguez dans les paquets debian en ligne

<https://packages.debian.org/search>

2. Le gestionnaire de paquet

Les backports

- Un intermédiaire entre stabilité et nouveauté
- Fournissent des paquets venant de `testing` en `stable`
- À utiliser avec prudence

En pratique ...

- Si on a besoin de dépendances récentes, on les installe généralement avec le gestionnaire de paquet correspondant au langage de notre app : `pip`, `npm`, `composer`, `carton`, `gem`, ...

2. Le gestionnaire de paquet

Et les autres distributions ?

- Redhat/Centos : `yum install <pkg>`, `yum search <keyword>`, `yum makecache`, `yum update`, ...
- Archlinux : `pacman -S <pkg>`, `-Ss <keyword>`, `-Syu`, ...

2. Le gestionnaire de paquet

Gérer des archives

`tar` (tape archive) permet de créer des archives (non compressées) qui rassemblent des fichiers.

```
# Créer une archive monarchive.tar  
tar -cvf monarchive.tar file1 file2 folder2/ folder2/  
  
# Désassembler une archive  
tar -xvf monarchive.tar
```

2. Le gestionnaire de paquet

Gérer des archives

`gzip` (gunzip) permet de compresser des fichiers (similaire aux .zip, .rar, ...)

```
# Compresser zblorf.scd  
gzip zblorf.scd
```

```
# [...] le fichier a été compressé et renommé zblorf.scd.gz
```

```
# Décompresser le fichier :  
gzip -d zblorf.scd.gz
```

2. Le gestionnaire de paquet

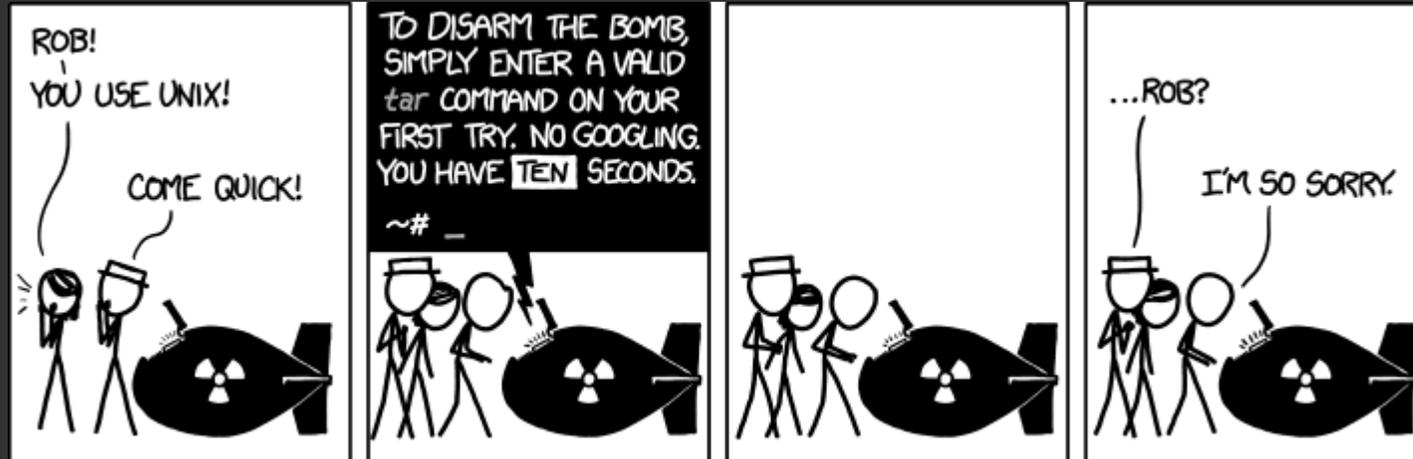
Gérer des archives

`tar` peut en fait être invoqué avec `-z` pour générer une archive compressée

```
# Créer une archive compressée  
tar -cvzf monarchive.tar.gz file1 file2 folder2/ folder2/  
  
# Désassembler une archive  
tar -xvzf monarchive.tar.gz
```

2. Le gestionnaire de paquet

Gérer des archives



3. Notions de réseau

en 60 slides !

3. Notions de réseau

Objectifs

- Comprendre et savoir se représenter les différentes couches
- Savoir faire quelques des tests "de base"
- ... et les commandes associées

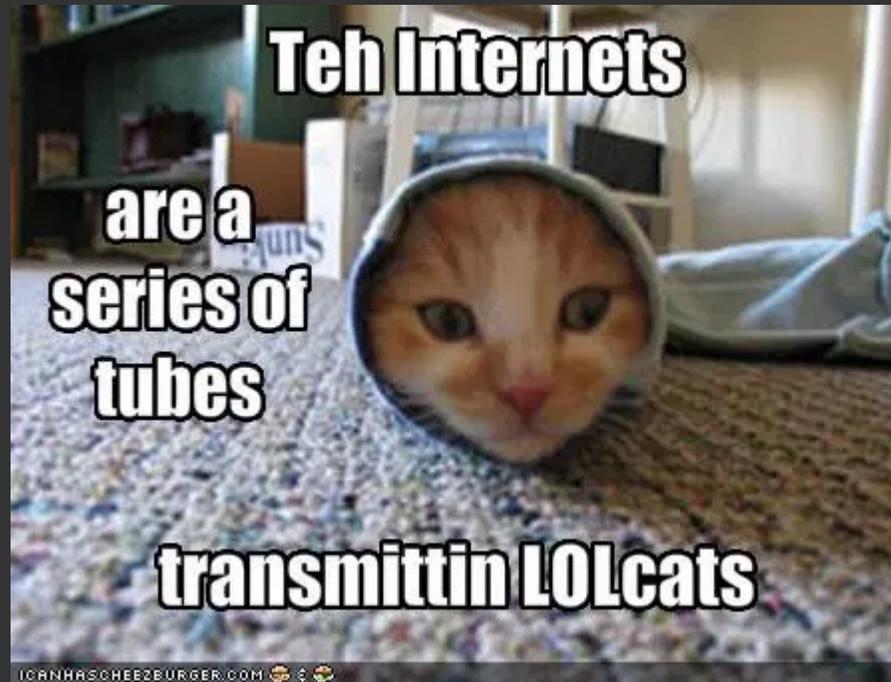
3. Notions de réseau

Notions essentielles à acquérir

- Comprendre ce qu'est une IP
- Comprendre ce qu'est un port
- Comprendre ce qu'est un client et un serveur (au sens logiciel)
- Comprendre ce qu'est un nom de domaine
- Comprendre ce qu'il se passe sous le capot lorsque vous visitez une page web

3. Notions de réseau

Teh interntez



3. Notions de réseau

Modele OSI

- Un empilement de couches
- Est là pour structurer la complexité du réseau
- Similaire au système : créer des abstractions
 - pour ne pas avoir à se soucier de ce qui se passe dans les couches "basses"
 - pour l'interopérabilité
- Chaque parti sur Internet implémente ces couches

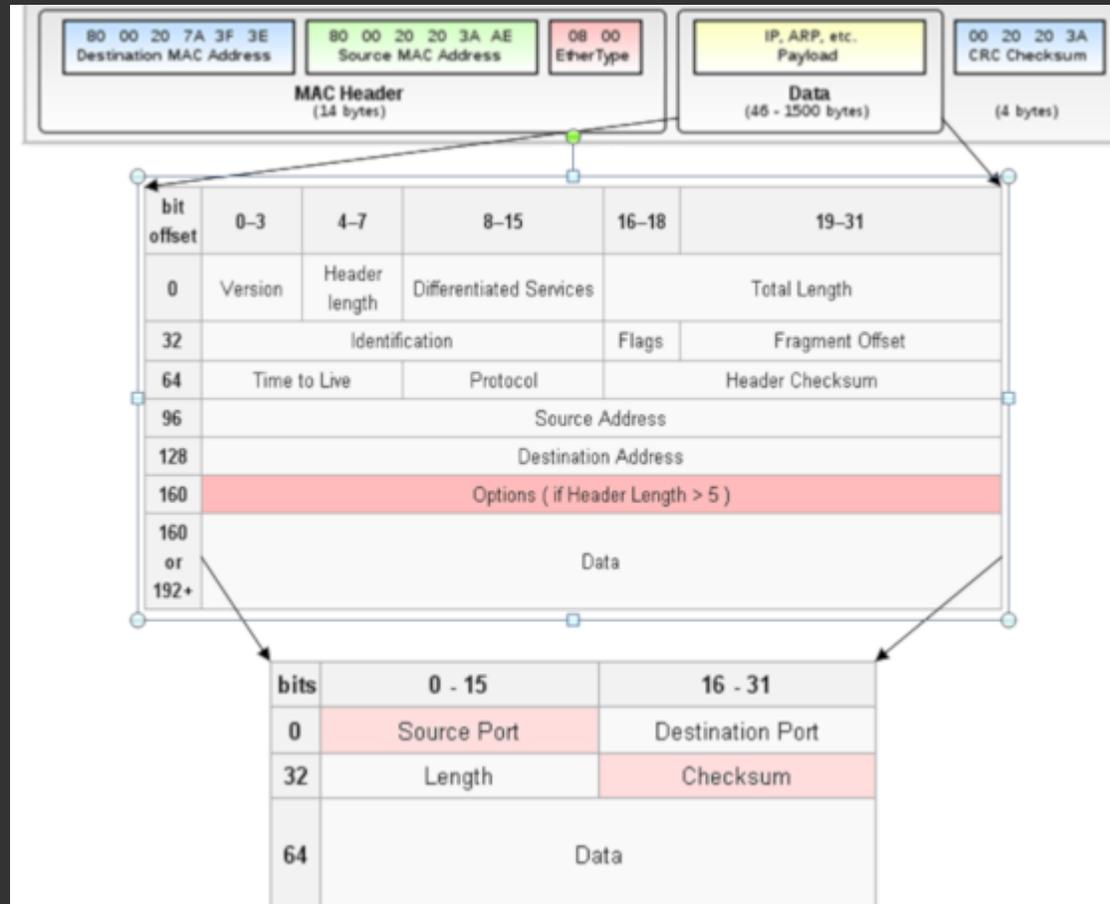
3. Notions de réseau

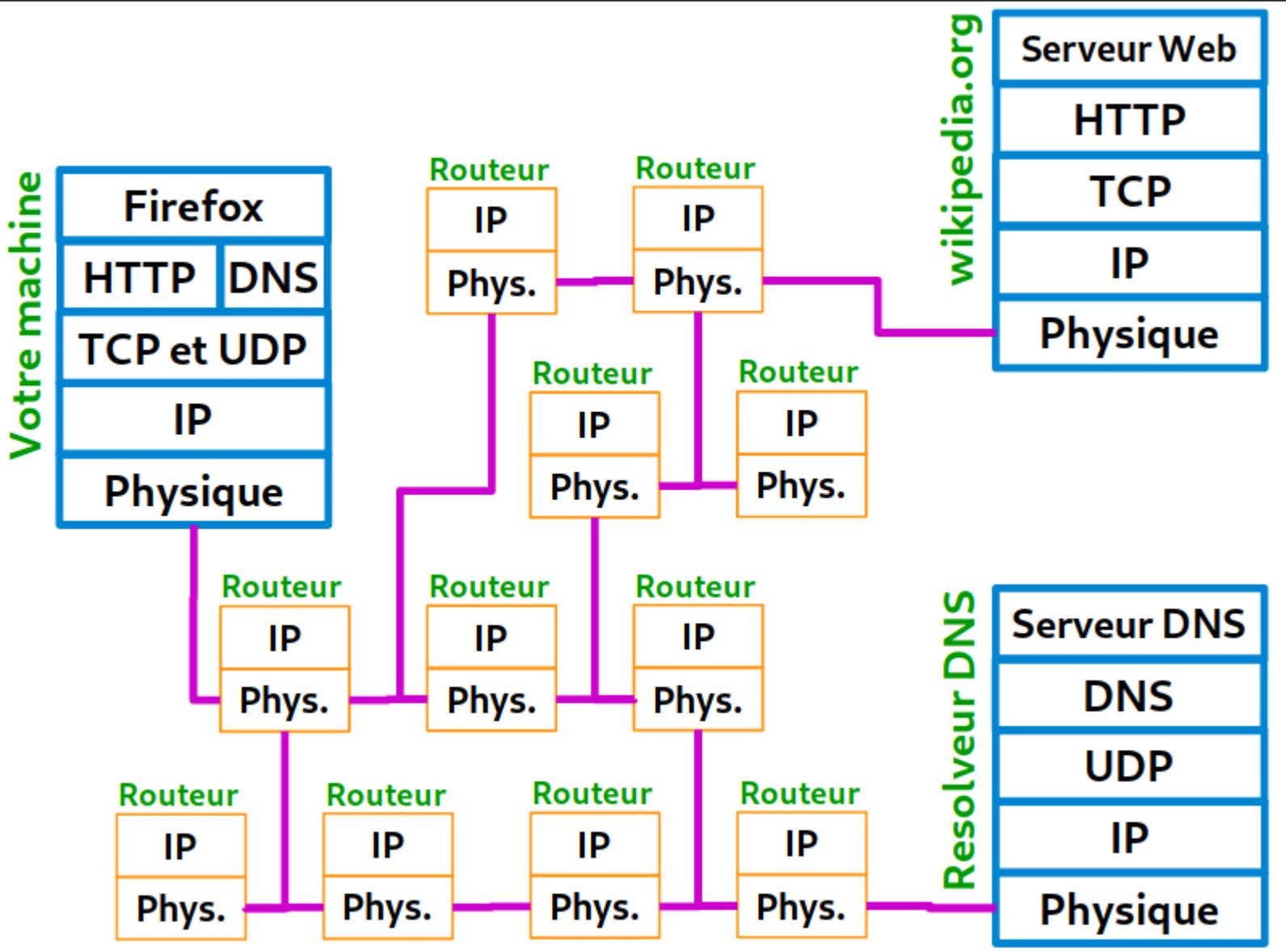
Modele OSI "simplifié": le modèle TCP/IP

- Application
- Transport (TCP)
- Internet (IP)
- Accès réseau (Ethernet, cables, ondes, ...)

3. Notions de réseau

Encapsulation des données



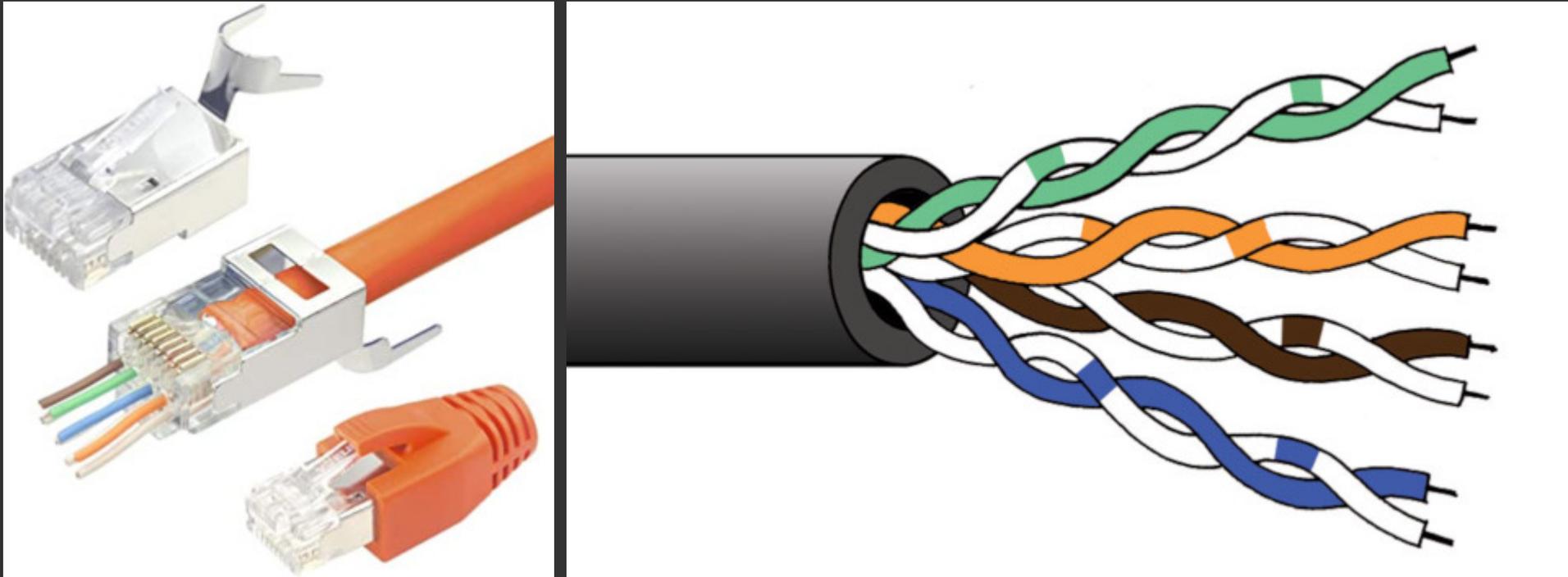


3. Notions de réseau

Exemple de réseau

3. Notions de réseau

Couche 1 : cable RJ45 / paires torsadées



- Différentes catégories de cable : CAT 5, 6, 7, (8)

3. Notions de réseau

Couche 1 : WiFi



- 2.4 GHz vs. 5 GHz
 - 2.4 GHz : meilleure portée, mais moins rapide, peu de canaux
 - 5 GHz : moins bonne portée, mais plus rapide, plus de canaux

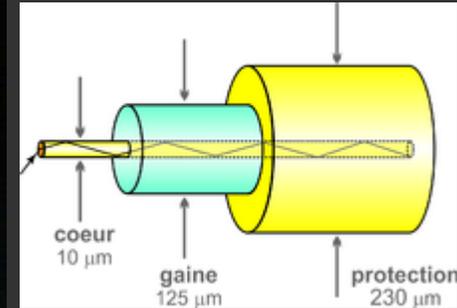
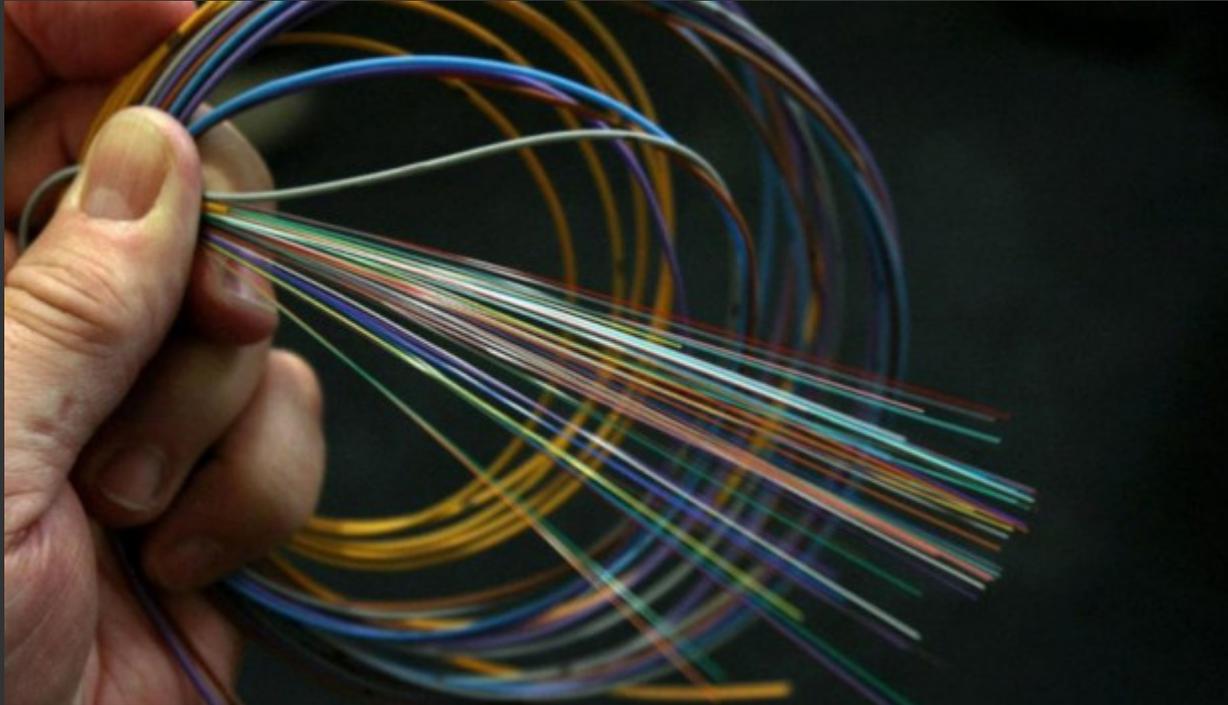
3. Notions de réseau

Couche 1 : 4G/5G



3. Notions de réseau

Couche 1 : fibre optique



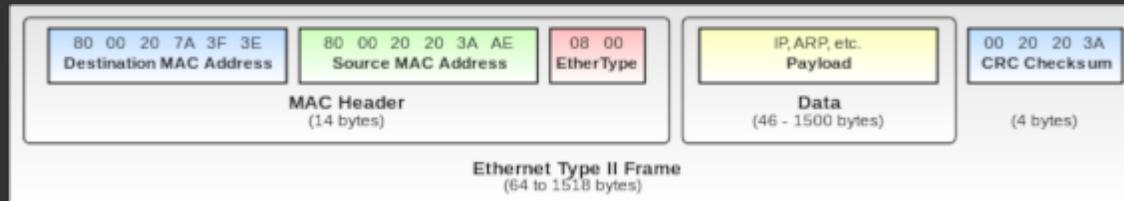
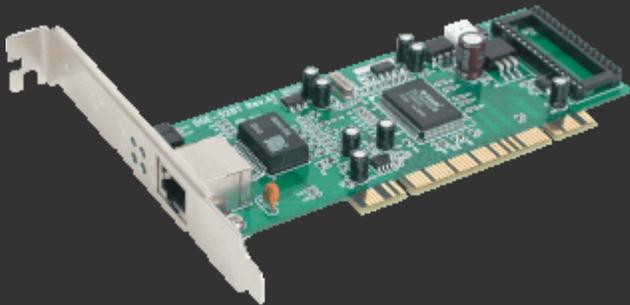
3. Notions de réseau

Couche 1 : liaisons intercontinentales

3. Notions de réseau

Couche 2 : Ethernet

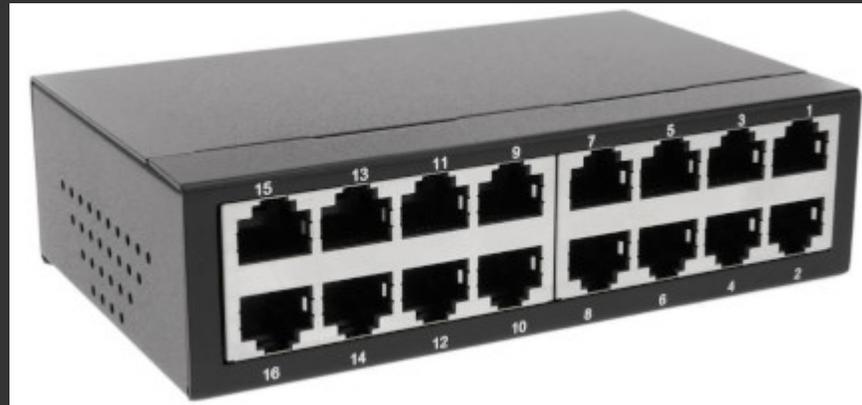
- Protocole pour transmettre l'information sur le médium physique
- Adresse MAC, par ex. `4c:96:0b:7d:d3:1a`
- Les ordinateurs disposent de cartes d'interface ethernet (filaire, wifi)
- (Ethernet s'applique **aussi** au WiFi)



3. Notions de réseau

Couche 2 : Ethernet

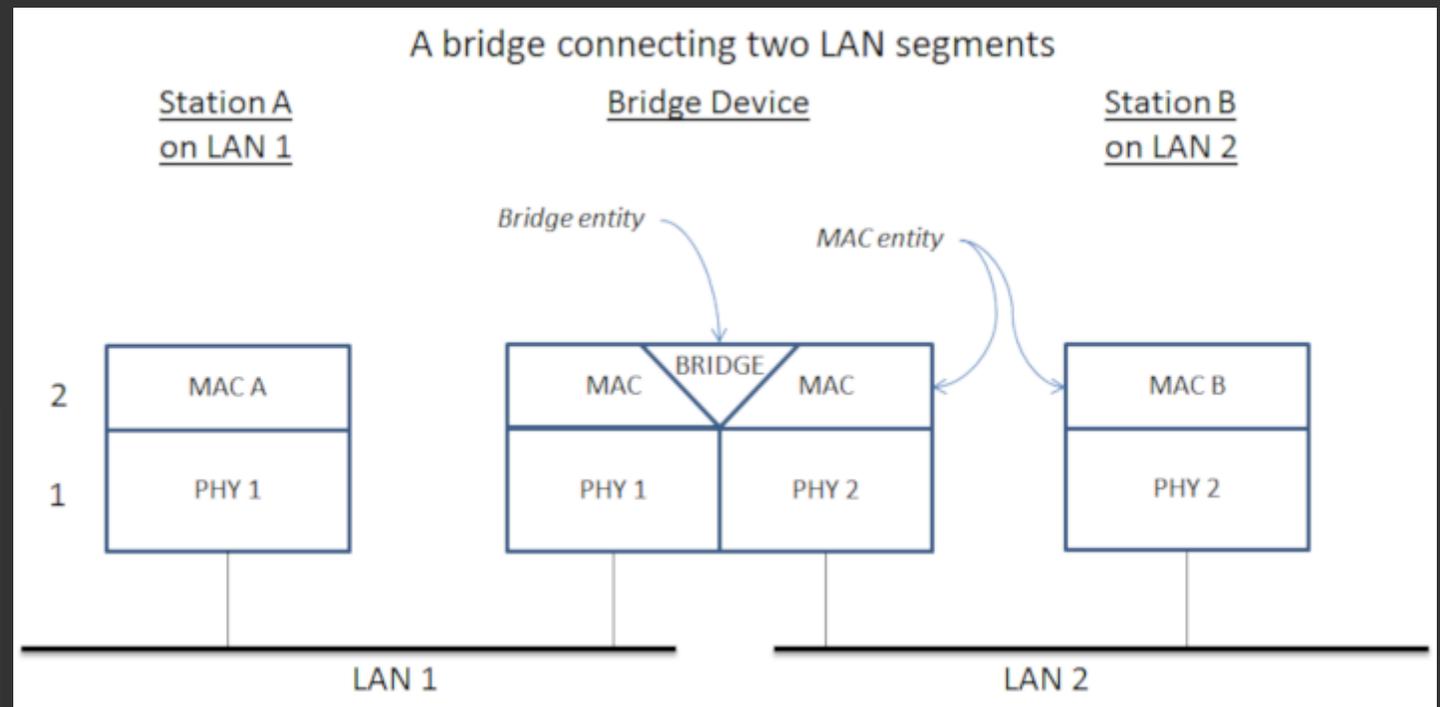
- Les **switch** permettent de connecter plusieurs machines pour créer segment
- Un **switch** est "conscient" de la notion d'adresse ethernet



3. Notions de réseau

Couche 2 : Ethernet

- Un **bridge** permet de "fusionner" plusieurs LAN ensemble



3. Notions de réseau

Couche 2 : Ethernet

Qu'est-ce qu'un VLAN ?

3. Notions de réseau

Couche 2 : les interfaces dans Linux

- Les interfaces sont configurées grâce aux fichiers `/etc/network/interfaces` et `/etc/network/interfaces.d/*`
- `ip a` permet d'obtenir des informations sur les interfaces
 - Historiquement, les noms étaient "simple" : `eth0`, `eth1`, `wlan0`, ...
 - Aujourd'hui les noms sont un peu plus complexes / arbitraires
 - Il existe toujours une interface `lo` (loopback, la boucle locale - 127.0.0.1)
 - Il peut y'avoir d'autres interfaces ou bridges "virtuelles" (contexte de conteneur, etc..)

3. Notions de réseau

Couche 2 : les interfaces dans Linux

```
$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP>
   link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: enp0s25: <NO-CARRIER,BROADCAST,MULTICAST,UP>
   link/ether 33:0e:d8:3f:65:7e
3: wlp3s0: <BROADCAST,MULTICAST,UP,LOWER_UP>
   link/ether 68:a6:2d:9f:ad:07
```

3. Notions de réseau

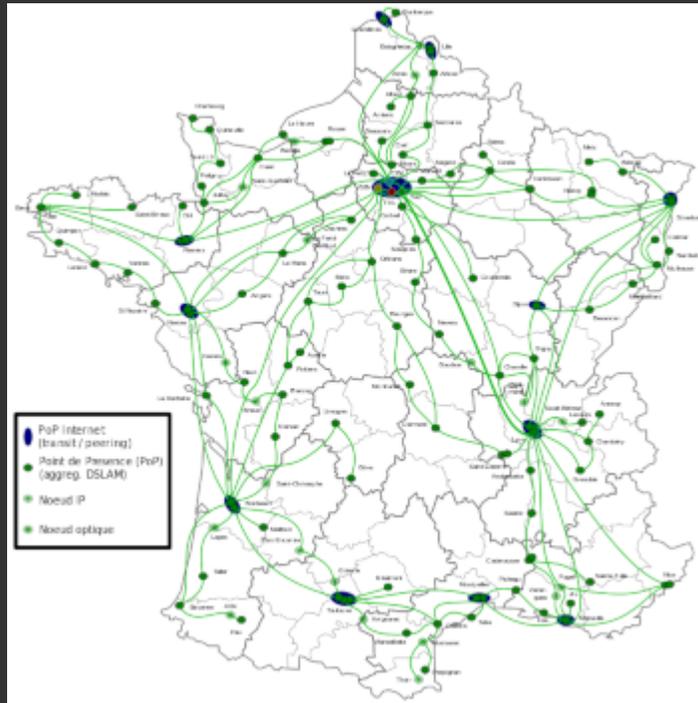
Couche 3 : IP

- IP pour *Internet Protocol*
- IP fait parler **des machines** !
 - .. et permet de relier plusieurs réseaux, qui potentiellement ont des fonctionnements différents
- Protocole de routage des paquets
 - "Best-effort", non fiable !
- Les routeurs, les facteurs d'internet
 - par ex. votre box internet
 - Routeur != Switch, un routeur "comprends" les adresses et protocole IP
 - Capable de discuter entre eux pour optimiser l'acheminement (BGP)

3. Notions de réseau

Couche 3 : IP

- Internet, c'est avant-tout une INTERconnexion d'opérateurs réseaux (NET)
- Ex: le réseau de l'opérateur Proxad



Couche 3 : IP

- Les opérateurs (AS) s'interconnectent (peering) dans des IXP
- Croissance "organique" du réseau

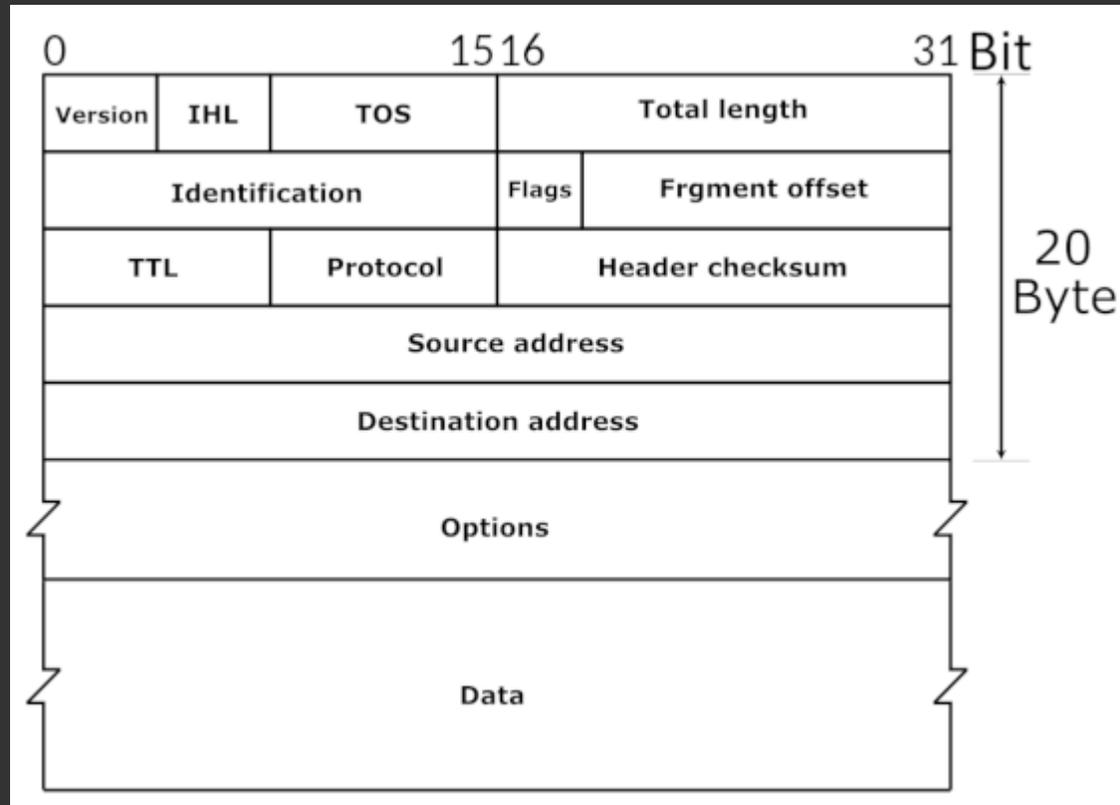
3. Notions de réseau

Couche 3 : IP : système d'adressage (IPv4)

- adresses codées sur 32 bits (4 nombres entre 0 et 255)
- par exemple `92.93.127.10`
- "seulement" 4.3 milliards d'adresses ! (pénurie)

3. Notions de réseau

Couche 3 : IP : IPv4 frame / paquet



3. Notions de réseau

Couche 3 : IP

- Distribution des adresses IP gérées par des ONG (IANA, RIR, LIR, ISP, ...)

3. Notions de réseau

Couche 3 : IP

- Distribution des adresses IP gérées par des ONG (IANA, RIR, LIR, ISP, ...)



3. Notions de réseau

Couche 3 : IP

- Notion de plage d'IP, réseau, masques de sous-réseau, notation CIDR
 - une adresse IP est composée d'une partie "réseau" (préfixe) et d'une partie "hôte"
 - par exemple `192.65.196.0/23` est un bloc de 512 IP attribué au CERN
 - `/23` signifie que les 23 premiers bits constituent la partie réseau
 - Il reste donc $32-23=9$ bits pour la partie hôte, soit $2^9 = 512$ IP
 - Les masques "typiques" sont `/8`, `/16`, `/24` et `/32`

3. Notions de réseau

Couche 3 : IP

- Certains blocs d'IP sont réservés à certains usages
 - Loopback (interne à la machine)
 - 127.0.0.0/8 (c.f. typiquement 127.0.0.1)
 - Réseau locaux (private network)
 - 192.168.0.0/16
 - 10.0.0.0/8
 - 172.16.0.0/12
 - Autres : c.f. https://en.wikipedia.org/wiki/Reserved_IP_addresses

3. Notions de réseau

Couche 3 : Et l'IPv6 ?

- Adresses codées sur 128 bits (soit 2^{94} fois plus d'adresses que IPv4 -> 10^{38} addresses)
 - Par exemple, `2a04:7260:9088:6c00:0044:0000:0000:0001`
 - En IPv6, on peut simplifier les `0` et juste écrire:
`2a04:7260:9088:6c00:44::1`
 - L'équivalent de `127.0.0.1` est `::1`
 - L'équivalent de `192.168.0.0/16` est `fc00::/10`
 - Les masques vont jusqu'à `/128`
- Beaucoup plus commun d'avoir directement un IP "globale" pour chaque machine, "directement" exposée sur le "vrai" internet
 - ... voir même un préfixe, comme par exemple un `/56`

3. Notions de réseau

Couche 3 : Et l'IPv6 ?

- Certains commandes ont un équivalent "v6" (par ex. `ping6`) et/ou une option `-6` (par ex. `ping -6`)
 - pour les URLs, le `:` conflicte avec la notation des ports, il faut alors écrire l'IP entre crochet
 - par ex: `https://[2001:db8:85a3:8d3:1319:8a2e:370:7348]:443/`

3. Notions de réseau

Couche 3 : Et l'IPv6 ?

- Existe depuis 1998 (sigh)
- Incompatible avec IPv4
 - période de transition "dual-stack"
 - problème d'oeuf et la poule / pas d'offre = pas de demande, etc

3. Notions de réseau

Couche 3 : commandes essentielles

`ip a` affiche les interfaces (et IPv4 et v6 associées)

```
$ ip a
enp3s0: <BROADCAST,MULTICAST,UP,LOWER_UP>
  link/ether 40:8d:5c:f3:3e:35
  inet 91.225.41.29/32 scope global enp3s0
  inet6 2a04:7202:8008:60c0::1/56 scope global
```

Voir aussi : `ifconfig` (deprecated) et `ipconfig` (sous windows!)

3. Notions de réseau

Couche 3 : commandes essentielles

`ping` teste la connexion entre deux machines

```
$ ping 91.198.174.192
PING 91.198.174.192 (91.198.174.192) 56(84) bytes of data.
64 bytes from 91.198.174.192: icmp_seq=1 ttl=58 time=51.5 ms
64 bytes from 91.198.174.192: icmp_seq=2 ttl=58 time=65.3 ms
^C
--- 91.198.174.192 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 3ms
rtt min/avg/max/mdev = 51.475/58.394/65.313/6.919 ms
```

Note: `ping` utilise le protocole `ICMP` qui a lieu au niveau de la couche 3 (ou 4 ?)

3. Notions de réseau

Couche 3 : commandes essentielles

`whois` pour obtenir des infos sur le(s) proprio(s) d'une ip

```
$ whois 91.198.174.192
[...]
organisation:      ORG-WFI2-RIPE
org-name:          Wikimedia Foundation, Inc
[...]
mnt-by:            RIPE-NCC-HM-MNT
mnt-by:            WIKIMEDIA-MNT
```

3. Notions de réseau

Couche 3 : commandes essentielles

`traceroute` permet d'étudier la route prise par les paquets

```
$ traceroute 91.198.174.192
 1  _gateway (192.168.0.1)  4.212 ms  6.449 ms  6.482 ms
 2  * 10.13.25.1 (10.13.25.1)  248.615 ms *
 3  211-282-253-24.rev.numericable.fr (211.282.253.24)  251.263 ms  251.9
 4  172.19.132.146 (172.19.132.146)  251.493 ms ip-65.net-80-236-3.stat
 5  prs-b7-link.telia.net (62.115.55.45)  251.692 ms  251.769 ms  251.9
 6  prs-bb4-link.telia.net (62.115.120.30)  252.026 ms prs-bb3-link.tel
 7  adm-bb4-link.telia.net (213.155.136.167)  1070.116 ms  1242.772 ms
 8  adm-b3-link.telia.net (62.115.122.179)  1243.006 ms adm-b3-link.tel
[...]
```

3. Notions de réseau

Couche 4 : TCP

- TCP pour Transmission Control Protocol (1/2)
- TCP est un protocole parmi d'autres qui ont lieu sur la couche 4
 - typiquement, il y a aussi UDP ...
- TCP fait communiquer **des programmes**
 - il y a une mise en place explicite d'un tuyau de communication
- Découpage des messages en petits paquets pour IP
- Fiabilité avec des accusés de réception / renvois

3. Notions de réseau

Couche 4 : Notion de port

- TCP fourni un "tuyau de communication" entre deux programmes
- Notion de 'port' : un nombre entre 1 et 65536 (2^{16})
 - Analogie avec les différents "département" à l'intérieur d'une entreprise
 - plusieurs programmes sur une même machine peuvent vouloir communiquer avec un même programme sur une machine distante, donc l'adresse IP ne suffit pas pour spécifier l'expéditeur / destinataire
- Une connexion entre deux programme est caractérisé par **deux** couples (IP:port)
- Par exemple : votre navigateur web (port 56723) qui discute qui discute avec le serveur web (port 80)
 - côté A : 183.92.18.6:56723 (un navigateur web)
 - côté B : 91.198.174.192:80 (un serveur web)

3. Notions de réseau

Couche 4 : commandes essentielles

`lsof -i` pour lister les connexions active

```
$ lsof -i
ssh          3231 alex IPv4 shadow.local:34658->142.114.82.73.rev.sfr.net
thunderbi   3475 alex IPv4 shadow.local:59424->tic.mailoo.org:imap (ESTA
thunderbi   3475 alex IPv4 shadow.local:57312->tic.mailoo.org:imap (ESTA
waterfox    12193 alex IPv4 shadow.local:54606->cybre.space:https (ESTABL
waterfox    12193 alex IPv4 shadow.local:32580->cybre.space:https (ESTABL
```

3. Notions de réseau

Couche 4 : commandes essentielles

ACHTUNG : ne pas abuser de cela..

```
$ nc -zv 44.112.42.13 22  
Connection to 44.112.42.13 22 port [tcp/ssh] succeeded!
```

3. Notions de réseau

Couche 4 : commandes essentielles

`tcpdump` pour regarder l'activité sur le réseau

3. Notions de réseau

Couche 4 : commandes essentielles

`wireshark`, similaire à `tcpdump`, mais beaucoup plus puissant, et en interface graphique

3. Notions de réseau

Couche 5+ : Modèle client/serveur

Un **serveur** (au sens logiciel) est un programme. Comme un serveur dans un bar (!)
:

- il **écoute** et attends qu'on lui demande un **service** en suivant **un protocole**
- par exemple : fournir la page d'accueil d'un site
- le serveur écoute sur *un port* : par exemple : 80

Le **client** est celui qui demande le service selon **le protocole**

- il toque à la bonne porte
- explique sa demande
- le serveur lui réponds (on espère)

3. Notions de réseau

Couche 5+ : netstat

`netstat -tulpn` permet de lister les programmes qui écoutent et attendent

```
> netstat -tulpn | grep LISTEN | grep "80\|25"
tcp        0.0.0.0:80      LISTEN    28634/nginx: master
tcp        0.0.0.0:25      LISTEN    1331/master # <- postfix, un serveur mail
tcp6       :::80      LISTEN    28634/nginx: master
tcp6       :::25      LISTEN    1331/master # <- postfix, un serveur mail
```

3. Notions de réseau

Couche 5+ : notion de protocole

- Un protocole = une façon de discuter entre programmes
- Conçus pour une finalité particulière
- Ont généralement un port "par défaut" / conventionnel (c.f. `/etc/services`)
 - 80/http : le web (des "vitrines" pour montrer et naviguer dans du contenu)
 - 443/https : le web (mais en chiffré)
 - 25/smtp : le mail (pour relayer les courriers électroniques)
 - 993/imap : le mail (synchroniser des boites de receptions)
 - 587/smtps : le mail (soumettre un courrier à envoyer)
 - 22/ssh : lancer des commandes à distance
 - 53/dns : transformer des noms en ip

3. Notions de réseau

Couche 5+ : HTTP

- On ouvre un socket TCP avec le serveur distant
- On envoie `GET /` et on reçoit 200 + la page d'accueil
- On envoie `GET /chaton.jpg` et on reçoit 200 + une image (si elle existe)
- On envoie `GET /meaningoflife.txt` et on reçoit 404 (si la page n'existe pas)
- On peut ajouter des Headers aux requetes et réponses (c.f. debugger firefox)
- Il existe d'autres requetes : POST, PUT, DELETE, ...

3. Notions de réseau

Couche 5+ : Le web

- Le web, ce n'est pas Internet
- Le web est construit grâce au langage HTML, généralement transporté par HTTP
- "Web" désigne la "toile" créée par les liens hypertextes, une fonctionnalité introduite par HTML

3. Notions de réseau

Le web

- Dans le modèle OSI:
 - 7 Application: votre onglet dans le navigateur, une application web
 - 6 Présentation: HTML, CSS, JS, PNG, ...
 - 5 Session: HTTP / HTTPS
 - 4 TCP
 - 3 IP
 - 2 (liaison)
 - 1 (physique)

3. Notions de réseau

DNS : Domain name server (1/5)

- Retenir cinquante numéros de téléphone (ou coordonnées GPS) par cœur, c'est pas facile
- On invente l'annuaire et les adresses postales
- `wikipedia.org -> 91.198.174.192`
- On peut acheter des noms chez des *registrars* (OVH, Gandi, ...)
- Composant critique d'Internet (en terme fonctionnel)
- Fonctionne en UDP et (et pas en TCP)

3. Notions de réseau

DNS : Domain name server (2/5)

- Il existe des résolveurs DNS à qui on peut demander de résoudre un nom via le protocole DNS (port 53)
- Par exemple :
 - 8.8.8.8, le resolveur de Google
 - 9.9.9.9, un nouveau service qui "respecte la vie privée"
 - 89.234.141.66, le resolveur de ARN
 - 208.67.222.222, OpenDNS
- **Choix critique pour la vie privée !!**
- Généralement, vous utilisez (malgré vous) le resolveur de votre FAI, ou bien celui de Google

3. Notions de réseau

DNS : Domain name server (3/5)

- Sous Linux, le résolveur DNS se configure via un fichier `/etc/resolv.conf`

```
$ cat /etc/resolv.conf  
nameserver 89.234.141.66
```

3. Notions de réseau

DNS : Domain name server (4/5)

`ping` fonctionne aussi avec noms de domaine

`host` permet sinon de connaître l'ip associée

```
$ host wikipedia.org
wikipedia.org has address 91.198.174.192
wikipedia.org has IPv6 address 2620:0:862:ed1a::1
wikipedia.org mail is handled by 50 mx2001.wikimedia.org.
wikipedia.org mail is handled by 10 mx1001.wikimedia.org.
```

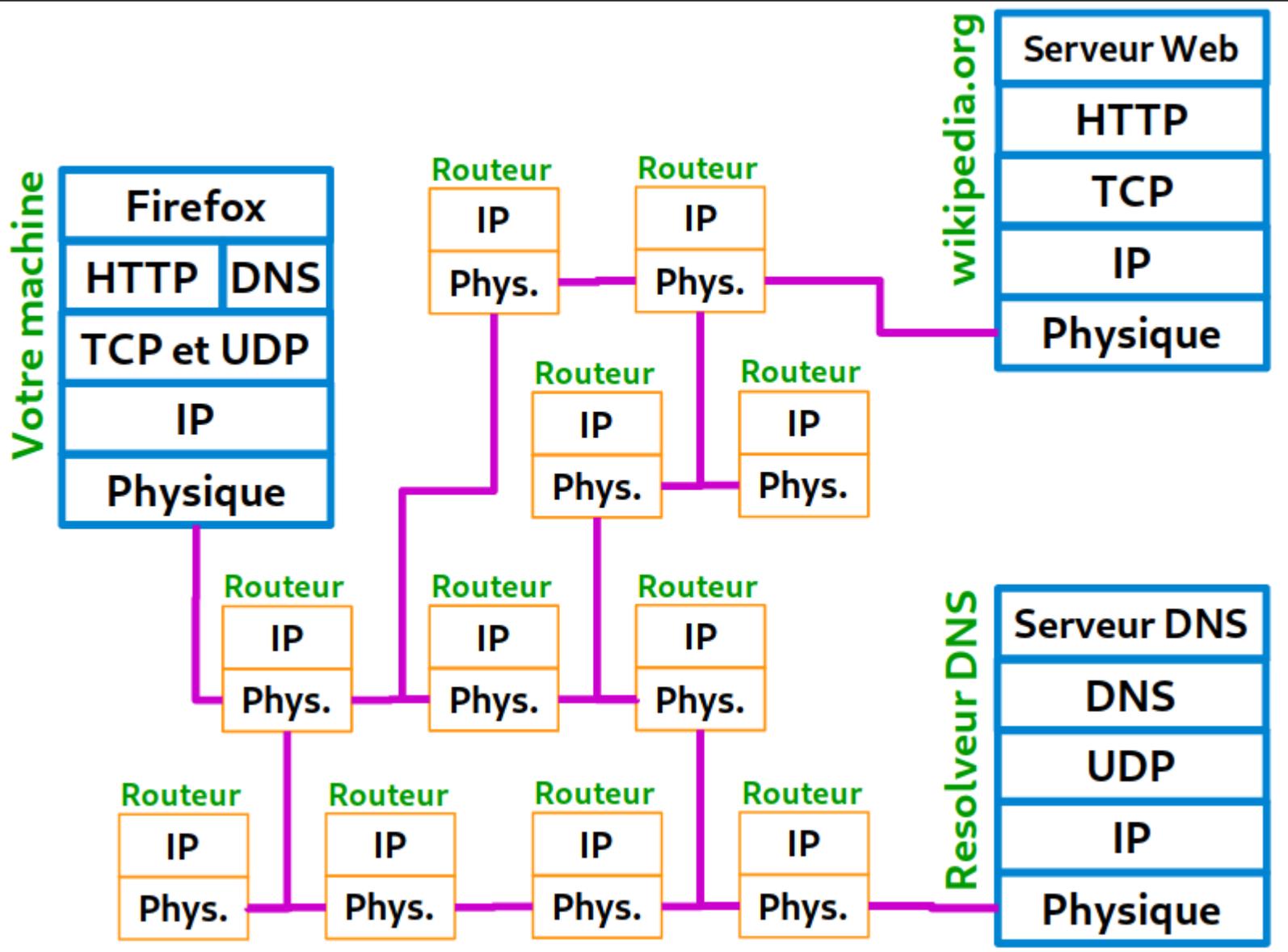
3. Notions de réseau

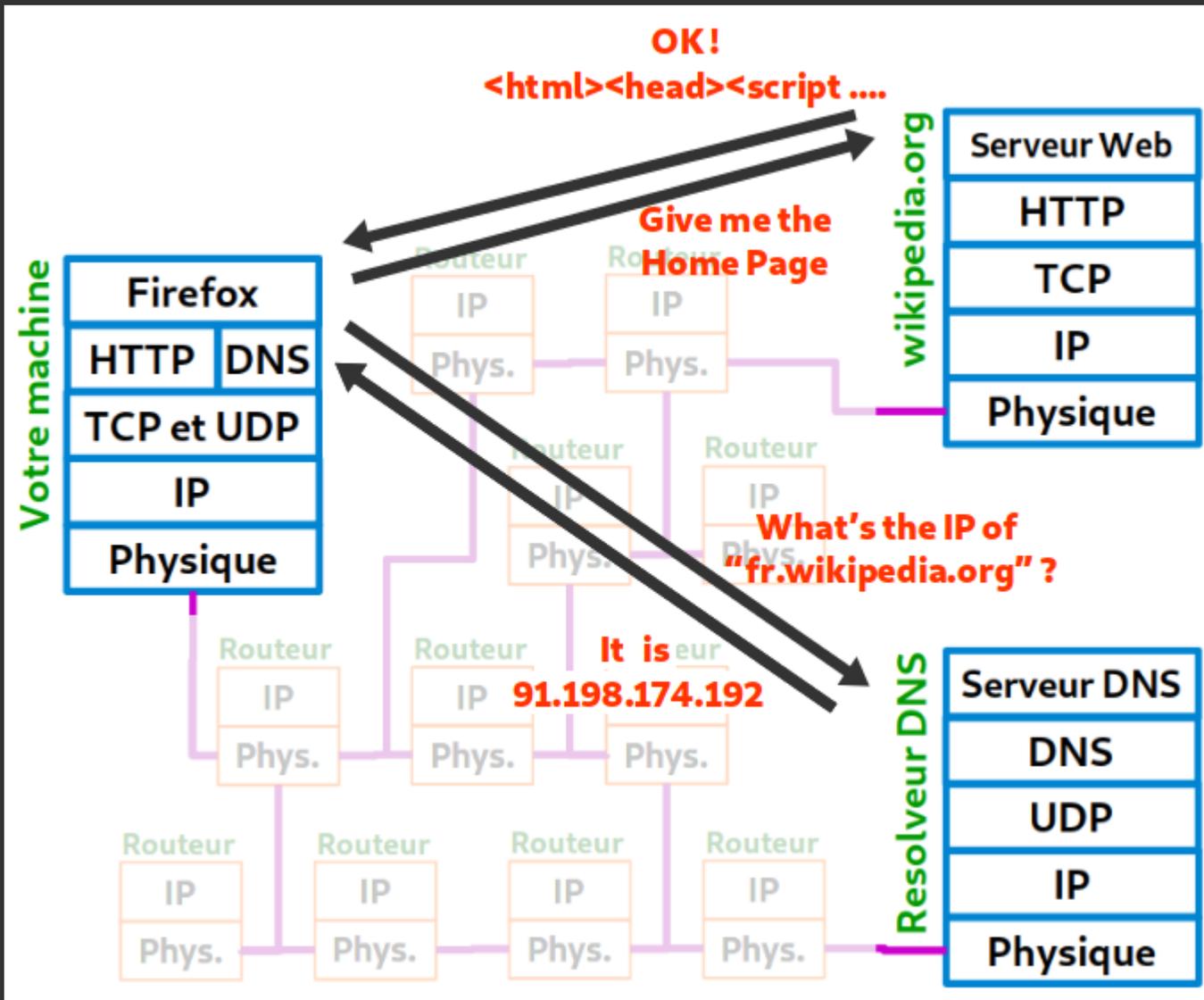
DNS : Domain name server (5/5)

- On peut outrepasser / forcer la résolution DNS de certains domaine avec le fichier `/etc/hosts`

```
> cat /etc/hosts
127.0.0.1    localhost
127.0.1.1    shadow
::1         localhost ip6-localhost ip6-loopback
ff02::1     ip6-allnodes
ff02::2     ip6-allrouters

127.0.0.1 google.com
127.0.0.1 google.fr
127.0.0.1 www.google.com
127.0.0.1 www.google.fr
127.0.0.1 facebook.com
```







- Give me the Home Page
- OK! `<html><head><script`



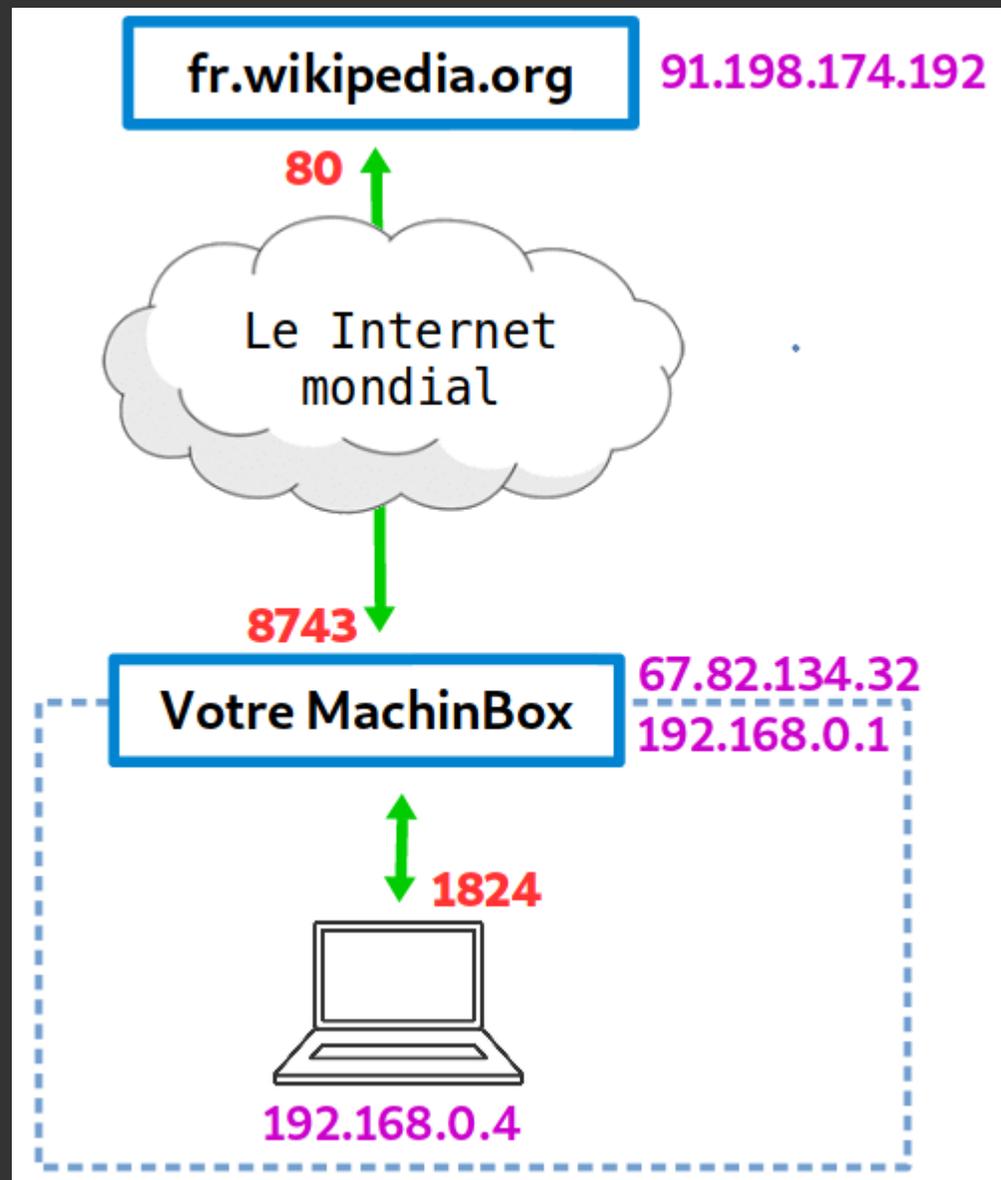
- What's the IP of "fr.wikipedia.org"?
- It is 91.198.174.192

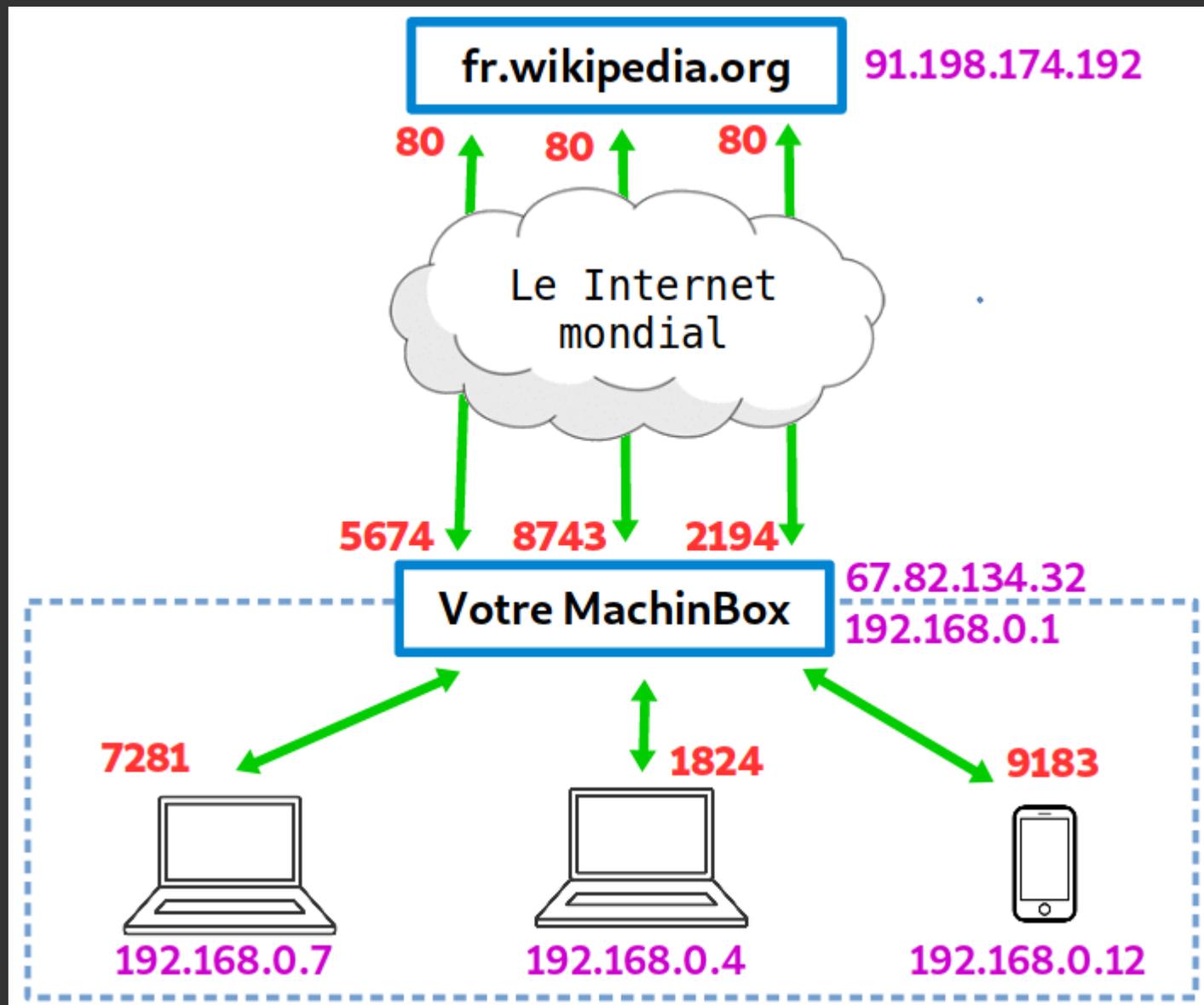


3. Notions de réseau

Réseau local, DHCP, NAT (1/6)

- En pratique, on est peu souvent "directement" connecté à internet
 - MachinBox
 - Routeur de l'entreprise
- Pas assez d'IPv4 pour tout le monde
 - nécessité de sous-réseaux "domestique" / des réseau "local"
 - basé sur les NAT (network address translation)
- Quand je me connecte au réseau:
 - mon appareil demande au routeur une IP, suivant le protocole DHCP (dynamic host configuration protocol)
 - le routeur a un range d'IP qu'il peut attribuer, typiquement quelque chose comme **192.168.0.0/24**





3. Notions de réseau

Réseau local, DHCP, NAT (4/6)

- Le routeur agit comme "gateway" (la "passerelle" vers les internets)
 - (c.f. `ip route`, et la route par défaut)
- Depuis l'extérieur du réseau local, il n'est pas possible de parler "simplement" à une machine
- Exemple : Je ne peux a priori pas parler à la machine 192.168.0.12 de mon réseau local chez moi depuis le centre de formation...
- Egalement : Difficulté de connaître sa vraie IP "globale" ! Il faut forcément demander à une autre machine ... c.f whatsmyip.com

3. Notions de réseau

Réseau local, DHCP, NAT (5/6)

La situation se complexifie avec Virtualbox :

- Typiquement Virtualbox créé un NAT à l'intérieur de votre machine
- Les différentes VM ont alors des adresses en 10.0.x.y

Le Internet mondial

Votre MachinBox

192.168.0.1

192.168.0.5



192.168.0.2



192.168.0.3



Routeur virtuel

10.0.0.1



10.0.0.2

Routeur virtuel

10.0.0.1



10.0.0.2

Routeur virtuel

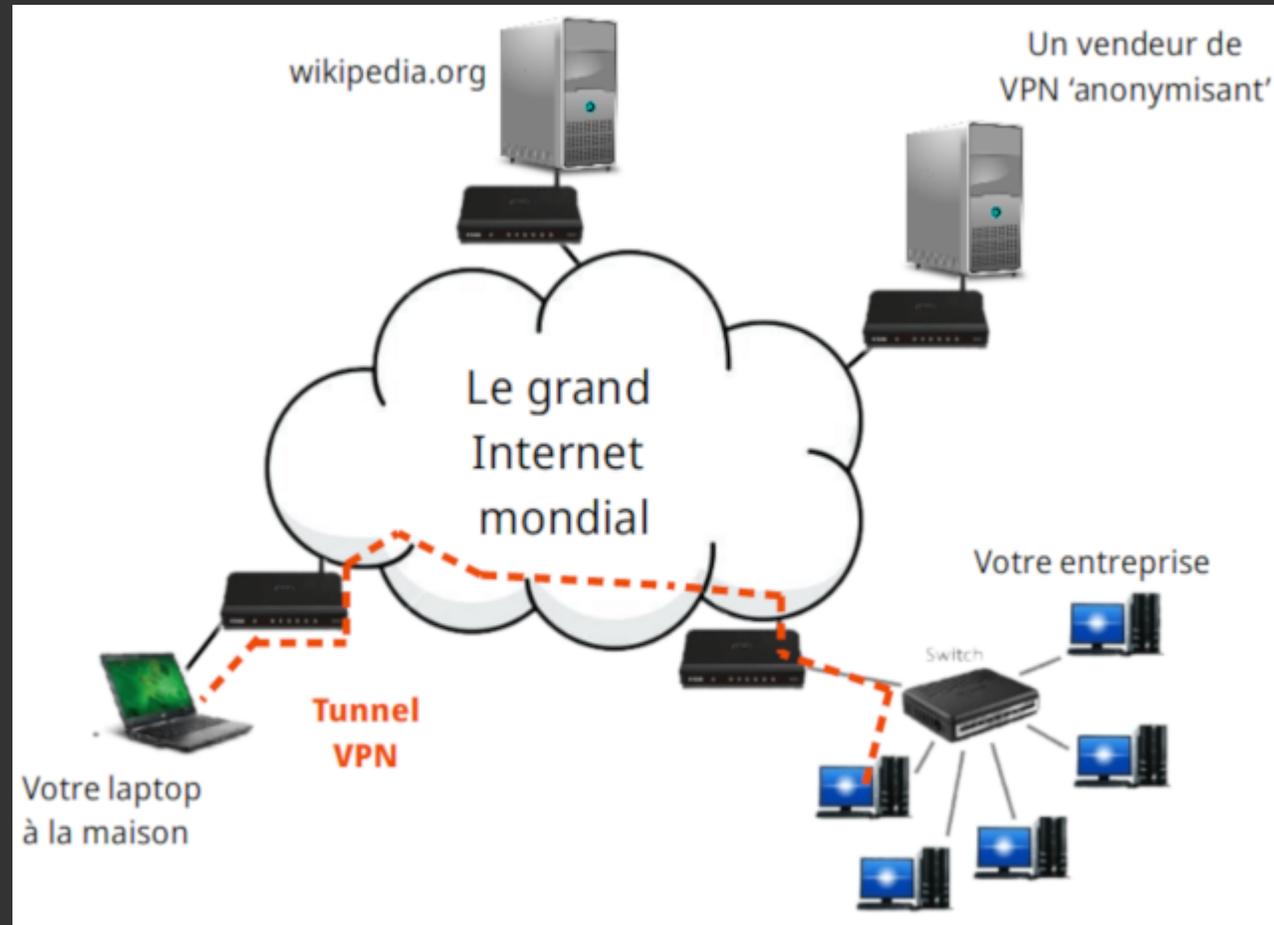
10.0.0.1



10.0.0.2

3. Notions de réseau

Et les VPNs, késaco ?



3. Notions de réseau

Et les VPNs, késaco ?

- Virtual Private Network
- Il s'agit de faire "comme si" on était connecté depuis un autre endroit

Plusieurs utilités possibles:

- accéder à des services accessibles seulement au sein d'un réseau privé (par ex. entreprise)
- forcer une communication à être chiffrée
- "anonymiser" ses requêtes (partager une IP commune avec pleins de gens)
- contourner des géo-restrictions
- ...

3. Notions de réseau

Autres notions : proxys, firewall

4. Notions de cryptographie

4. Notions de cryptographie

Principe, vocabulaire

Protéger des messages (confidentialité, authenticité, intégrité) en s'aidant souvent de secrets ou clés.

- Confidentialité : seul l'expéditeur et le destinataire ont accès au message
- Authenticité : le message reçu par le destinataire provient bien de l'expéditeur
- Intégrité : le message reçu est complet et n'a pas été déformé

4. Notions de cryptographie

Exemple de chiffrement symétrique

Historique : le nombre de César

- un algorithme : décalage des lettres dans l'alphabet
- un secret / une clef (par exemple : 3)
- pour déchiffrer : opération inverse triviale

```
Linux c'est sympatoche  
0lqxa f'hvw vbpsdwrfkh
```

4. Notions de cryptographie

Chiffrement asymétrique

Pas d'équivalent classique ...

- imaginer un sorte de nombre de César où l'on chiffre en décalant de 3 ...
- ... mais pour déchiffrer, il faut faire -12 !

4. Notions de cryptographie

Chiffrement asymétrique

Les mathématiques permettent de générer un couple de clef (A, B) :

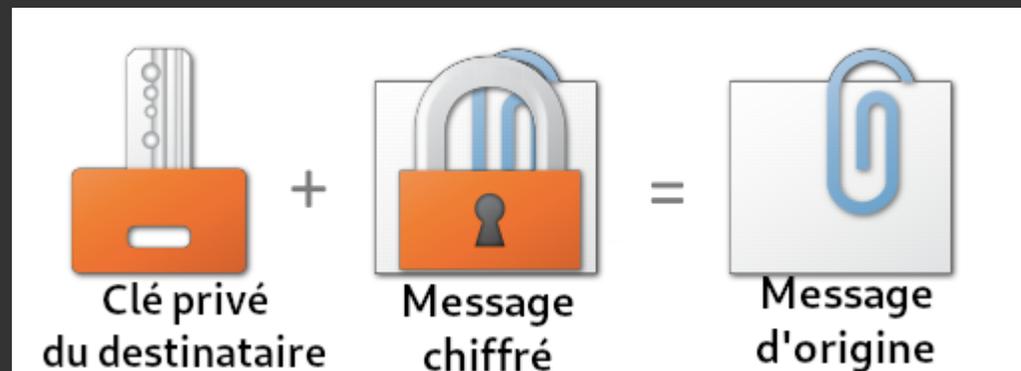
- `chiffrer(message, A)` peut être déchiffré uniquement avec `B`
- `chiffrer(message, B)` peut être déchiffré uniquement avec `A`

4. Notions de cryptographie

Chiffrement asymétrique

- On nomme une clef la clef **privée** : on la garde secrètement rien que pour nous
- On nomme l'autre la clef **publique** : on la donne à tout le monde
- Si quelqu'un cherche à vous envoyer un message, ils chiffrent en utilisant votre clef publique
- Vous seul avez la clef privée et pouvez déchiffrer.





4. Notions de cryptographie

Chiffrement asymétrique

- Le chiffrement asymétrique assure la confidentialité et l'intégrité
- Mais pas l'authenticité !
- Besoin d'un mécanisme de "signature"



Message de l'expéditeur

+



Clé publique du destinataire

=



Message chiffré pour le destinataire

Hash ↓

c48a7a1f076

Hash du document

+



Clef privée de l'expéditeur

=



Signature



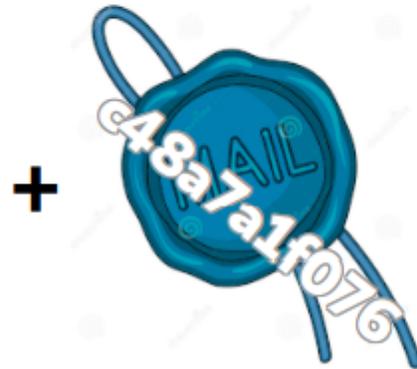
Clé privée
du destinataire

Message chiffré

Message d'origine



Clef publique
de l'expediteur



Signature

= **c48a7a1f076**



Le hash correspond ?
= bonne signature !

4. Notions de cryptographie

Echange de clef

- Vous recevez un mail de Edward Snowden avec sa clef publique en copie
- Comment s'assurer que c'est la vraie bonne clef ?
- (Spoiler alert : vous ne pouvez a priori pas...)

Problème général de sécurité : il est difficile de s'assurer de l'authenticité initiale de la clef publique

4. Notions de cryptographie

Solution 1 : la vraie vie

Voir Edward Snowden en chair et en os, et récupérer la clef avec lui

4. Notions de cryptographie

Solution 2 : web of trust

La clef de Edward Snowden a été signée par pleins de journalistes et activistes indépendant à travers le monde, ce qui diminue le risque d'une falsification

4. Notions de cryptographie

Solution 3 : autorités de certification

Vous faites confiance à Microsoft et Google (!?), qui certifient avoir vérifié que E. Snowden possède cette clef.

- C'est le principe des autorités de certification utilisé par HTTPS
- Votre navigateur fait confiance à des clefs prédéfinies correspondant à des tiers de "confiance" (e.g. Google, ...)
- Le certificat HTTPS contient une signature qui a été produite avec l'une des clefs de ces tiers de confiance
- Vous pouvez ainsi faire confiance "par délégation"

4. Notions de cryptographie

Applications

- HTTPS (SSL/TLS, x509)
- SSH
- Emails chiffrés
- Signature des paquets dans APT
- ...

5. Se connecter et gérer un serveur avec SSH

5. SSH et les serveurs

À propos des serveurs

Serveur (au sens matériel)

- machine destinée à fournir des services (e.g. un site web)
- allumée et connectée 24/7
- typiquement sans interface graphique
- ... et donc administrée à distance

5. SSH et les serveurs

À propos des serveurs

Serveur (au sens logiciel)

- aussi appelé "daemon", ou service
- programme qui écoute en permanence et attends qu'un autre programme le contacte
 - par ex. : un serveur web attends des clients
- écoute typiquement sur un ou plusieurs port
 - par ex. : 80 pour HTTP

5. SSH et les serveurs

Serveurs : quel support matériel ?



5. SSH et les serveurs

Serveurs : quel support matériel ?



5. SSH et les serveurs



... Plot twist !



5. SSH et les serveurs

"Virtual" Private Server (VPS)

VPS = une VM dans un datacenter



5. SSH et les serveurs

"Virtual" Private Server (VPS)

... qui tourne quelque part sur une vraie machine

5. SSH et les serveurs



DigitalOcean

MEMORY	vCPUs	SSD DISK	TRANSFER	PRICE
1 GB	1 vCPU	25 GB	1 TB	\$5/mo \$0.007/hr
2 GB	1 vCPU	50 GB	2 TB	\$10/mo \$0.015/hr
3 GB	1 vCPU	60 GB	3 TB	\$15/mo \$0.022/hr
DEVELOPMENT				
2 GB	2 vCPUs	60 GB	3 TB	\$15/mo \$0.022/hr
1 GB	3 vCPUs	60 GB	3 TB	\$15/mo \$0.022/hr

5. SSH et les serveurs



The image shows the Scaleway logo at the top, which consists of a stylized 'S' icon followed by the word 'scaleway' in a lowercase, sans-serif font.

START1-XS	€1.99/mo €0.004/hour
1 X86 64bit Cores 1GB memory 100Mbit/s Unmetered	
START1-S	€3.99/mo €0.008/hour
2 X86 64bit Cores 2GB memory 200Mbit/s Unmetered	
START1-M	€7.99/mo €0.016/hour
4 X86 64bit Cores 4GB memory 300Mbit/s Unmetered	
START1-L	€15.99/mo €0.032/hour
8 X86 64bit Cores 8GB memory 400Mbit/s Unmetered	

5. SSH et les serveurs

SSH : Secure Shell

- Un protocole **client-serveur**, par défaut sur le port 22
- Prendre le contrôle d'une machine à distance via un shell
- Sécurisé grâce à du chiffrement asymétrique
 - le serveur a un jeu de clef publique/privé
 - le client peut aussi en avoir un (sinon : mot de passe)
- Outil "de base" pour administrer des serveurs

5. SSH et les serveurs

Syntaxe : `ssh utilisateur@machine`

```
$ ssh admin@ynh-forge.netlib.re
The authenticity of host 'ynh-forge.netlib.re (46.101.221.117)' can't be
RSA key fingerprint is SHA256:CuPd7AtmqS0UE6DwDDG68hQ+qIT2tQqZqm8pfo2oB
Are you sure you want to continue connecting (yes/no)? █
```

5. SSH et les serveurs

Syntaxe : `ssh utilisateur@machine`

```
$ ssh admin@ynh-forge.netlib.re
The authenticity of host 'ynh-forge.netlib.re (46.101.221.117)' can't be
RSA key fingerprint is SHA256:CuPd7AtmqS0UE6DwDDG68hQ+qIT2tQqZqm8pfo2oB
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'ynh-forge.netlib.re' (RSA) to the list of k
Debian GNU/Linux 9
admin@ynh-forge.netlib.re's password: █
```

5. SSH et les serveurs

Syntaxe : `ssh utilisateur@machine`

```
$ ssh admin@ynh-forge.netlib.re
The authenticity of host 'ynh-forge.netlib.re (46.101.221.117)' can't be
RSA key fingerprint is SHA256:CuPd7AtmqS0UE6DwDDG68hQ+qIT2tQqZqm8pfo2oB
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'ynh-forge.netlib.re' (RSA) to the list of k
Debian GNU/Linux 9
admin@ynh-forge.netlib.re's password:

Last login: Thu Oct  4 08:52:07 2018 from 90.63.229.46
admin@ynh-forge:~$ █
```

5. SSH et les serveurs

SSH : se logger

- ACHTUNG : Soyez attentif à dans quel terminal vous tapez !!!
- En se connectant la première fois, on vérifie la clef publique du serveur
- On a besoin du mot de passe pour se connecter
- ... mais la bonne pratique est d'utiliser nous-aussi une clef

5. SSH et les serveurs

SSH : avec une clef

... mais pourquoi ?

- Pas de mot de passe qui se balade sur le réseau
- Pas nécessaire de retaper le mot de passe à chaque fois
- Possibilité d'automatiser des tâches (clef sans mot de passe)
- (Plusieurs personnes peuvent avoir accès à un meme utilisateur sans devoir se mettre d'accord sur un mot de passe commun)

5. SSH et les serveurs

SSH : avec une clef

1 - Générer avec `ssh-keygen -t rsa -b 4096 -C "commentaire ou description"`

```
$ ssh-keygen -t rsa -b 4096 -C "Clef pour la formation"
```

5. SSH et les serveurs

SSH : avec une clef

1 - Générer avec `ssh-keygen -t rsa -b 4096 -C "commentaire ou description"`

```
$ ssh-keygen -t rsa -b 4096 -C "Clef pour la formation"
Generating public/private rsa key pair.
Enter file in which to save the key (/home/alex/.ssh/id_rsa):
Enter passphrase (empty for no passphrase): # Mot de passe
Enter same passphrase again: # (again)
Your identification has been saved in /home/alex/.ssh/id_rsa.
Your public key has been saved in /home/alex/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:ZcAKHVtTXUPz3ipqia4i+soRHZQ4tYsDGfc5ieEGWcY "Clef pour la format
```

5. SSH et les serveurs

SSH : avec une clef

2 - Configurer la clef sur le serveur

- soit *depuis le client* avec

```
ssh-copy-id -i chemin/vers/la/clef user@machine
```

- soit *depuis le serveur* en rajoutant la clef dans `~/.ssh/authorized_keys`
 - (généralement, l'admin vous demande votre clef)

5. SSH et les serveurs

SSH : avec une clef

3 - Utiliser la clef pour se connecter

```
$ ssh -i ~/.ssh/ma_clef alex@jaimелеcafe.com  
Enter passphrase for key '/home/alex/.ssh/ma_clef': █
```

5. SSH et les serveurs

SSH : avec une clef

3 - Utiliser la clef pour se connecter

```
$ ssh -i ~/.ssh/ma_clef alex@jaimelecafe.com
Enter passphrase for key '/home/alex/.ssh/ma_clef':

Last login: Mon Oct  8 19:46:32 2018 from 11.22.33.44
user@jaimelecafe.com:~$ █
```

- Le système peut potentiellement se souvenir du mot de passe pour les prochaines minutes, comme avec sudo
- Il peut ne pas y avoir de mot de passe (utilisation dans des scripts)

5. SSH et les serveurs

SSH : configuration côté client

- Le fichier `~/.ssh/config` peut être édité pour définir des machines et les options associées

```
Host jaimlecafe
  User alex
  Hostname jaimlecafe.com
  IdentityFile ~/.ssh/ma_clef
```

- On peut ensuite écrire simplement : `ssh jaimlecafe`

5. SSH et les serveurs

5. SSH et les serveurs

SCP : copier des fichiers

`scp <source> <destination>` permet de copier des fichiers entre le client et le serveur

- Le chemin d'un fichier distant s'écrit `machine:/chemin/vers/fichier`
- ou (avec un user) : `utilisateur@une.machine.com:/chemin/vers/fichier`

Exemples :

```
$ scp slides.html bob@dismorphia.info:/home/alex/  
$ scp bob@dismorphia.info:/home/alex/.bashrc ./
```

5. SSH et les serveurs

Divers

- Client SSH sous Windows : MobaXterm
- `sshfs` pour monter des dossiers distants
- `ssh -D` pour créer des tunnels chiffrés (similaires à des VPNs)

6 - Services et sécurité basique d'un serveur

6 - Services et sécurité

Objectifs

- Parler de la gestion des services
- Tout en appliquant ça à certaines pratiques "de base" de sécurité d'un serveur

6 - Services et sécurité

sshd

- Un service ou "daemon" qui écoute sur le port 22
- Il gère les connexions SSH ...
- comme d'autres services : il passe sa vie toujours éveillé et prêt à répondre
- Comme beaucoup d'autre programmes : sa configuration est dans `/etc/` et ses logs dans `/var/log/`

En particulier :

- `/etc/ssh/sshd_config` : configuration du daemon
- `/var/log/daemon.log` : un fichier de log utilisé par plusieurs daemons
- `/var/log/auth.log` : logs d'authentification

6 - Services et sécurité

/etc/ssh/sshd_config

Port 22

HostKey /etc/ssh/ssh_host_ecdsa_key

PermitRootLogin yes

AllowGroups root ssh

6 - Services et sécurité

Bonnes pratiques en terme de ssh

- (plus ou moins subjectif !..)
- Changer le port 22 en quelque chose d'autre (2222, 2323, 2200, ...)
- Désactiver le login root en ssh
- Utiliser exclusivement des clefs

6 - Services et sécurité

Gérer un service avec **systemd**

```
$ systemctl status <nom_du_service> # Obtenir des informations sur le service
```

```
$ systemctl start <nom_du_service> # Démarrer le service  
$ systemctl reload <nom_du_service> # Recharger la configuration  
$ systemctl restart <nom_du_service> # Redémarrer le service  
$ systemctl stop <nom_du_service> # Stopper le service
```

```
$ systemctl enable <nom_du_service> # Lancer le service au démarrage du système  
$ systemctl disable <nom_du_service> # Ne pas lancer le service au démarrage du système
```

```
systemctl status ssh
```

```
● ssh.service - OpenBSD Secure Shell server
```

```
Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor pre
```

```
Active: active (running) since Wed 2018-10-10 17:43:11 UTC; 3h 17min
```

```
Main PID: 788 (sshd)
```

```
CGroup: /system.slice/ssh.service
```

```
└─788 /usr/sbin/sshd -D
```

```
Oct 10 20:39:34 scw-5e2fca sshd[5063]: input_userauth_request: invalid
Oct 10 20:39:34 scw-5e2fca sshd[5063]: pam_unix(sshd:auth): check pass;
Oct 10 20:39:34 scw-5e2fca sshd[5063]: pam_unix(sshd:auth): authenticat
Oct 10 20:39:37 scw-5e2fca sshd[5063]: Failed password for invalid user
Oct 10 20:39:37 scw-5e2fca sshd[5063]: Connection closed by 5.101.40.10
```

6 - Services et sécurité

Investiguer des logs

- Fouiller `/var/log` ... par exemple : `/var/log/auth.log`

```
Oct 10 20:50:35 scw-5e2fca sshd[5157]: Invalid user user from 5.101.40.1
Oct 10 20:50:35 scw-5e2fca sshd[5157]: input_userauth_request: invalid u
Oct 10 20:50:35 scw-5e2fca sshd[5157]: pam_unix(sshd:auth): check pass;
Oct 10 20:50:35 scw-5e2fca sshd[5157]: pam_unix(sshd:auth): authenticati
Oct 10 20:50:38 scw-5e2fca sshd[5157]: Failed password for invalid user
Oct 10 20:50:38 scw-5e2fca sshd[5157]: Connection closed by 5.101.40.101
Oct 10 21:01:37 scw-5e2fca sshd[5174]: Invalid user user from 5.101.40.1
Oct 10 21:01:37 scw-5e2fca sshd[5174]: input_userauth_request: invalid u
Oct 10 21:01:37 scw-5e2fca sshd[5174]: pam_unix(sshd:auth): check pass;
Oct 10 21:01:37 scw-5e2fca sshd[5174]: pam_unix(sshd:auth): authenticati
Oct 10 21:01:39 scw-5e2fca sshd[5174]: Failed password for invalid user
Oct 10 21:01:39 scw-5e2fca sshd[5174]: Connection closed by 5.101.40.101
```

6 - Services et sécurité

Investiguer des logs

The systemd way : `journalctl -u <nom_du_service>`

Par exemple : `journalctl -u ssh`

6 - Services et sécurité

Protéger contre le brute-force : fail2ban

- Fail2ban analyse automatiquement les logs
- Cherche / détecte des activités suspectes connues
 - Par exemple : une IP qui essaye des mots de passe
- Déclenche une action ... comme bannir l'IP pour un certain temps
 - (Basé sur iptables qui permet de définir des règles réseau)
- Les "jails" sont configurées via /etc/fail2ban/jail.conf
- Fail2ban loggue ses actions dans /var/log/fail2ban.log

6 - Services et sécurité

fail2ban : exemple de la jail SSH

- Analyse `/var/log/auth.log`
- Recherche des lignes comme `Failed password for user from W.X.Y.Z`

```
# Global settings
bantime = 600
findtime = 600
maxretry = 5

[sshd]
port = ssh
logpath = /var/log/auth.log
```

6 - Services et sécurité

fail2ban : le log de fail2ban

```
2018-10-10 20:50:35 INFO      [sshd] Found 5.101.40.101
2018-10-10 20:50:35 INFO      [sshd] Found 5.101.40.101
2018-10-10 20:50:38 INFO      [sshd] Found 5.101.40.101
2018-10-10 20:50:39 NOTICE   [sshd] Ban 5.101.40.101
2018-10-10 21:00:40 NOTICE   [sshd] Unban 5.101.40.101
2018-10-10 21:01:37 INFO      [sshd] Found 5.101.40.101
2018-10-10 21:01:37 INFO      [sshd] Found 5.101.40.101
2018-10-10 21:01:39 INFO      [sshd] Found 5.101.40.101
2018-10-10 21:01:40 NOTICE   [sshd] Ban 5.101.40.101
2018-10-10 21:11:41 NOTICE   [sshd] Unban 5.101.40.101
```

6 - Services et sécurité

fail2ban : exemple de la jail récursive

- Analyse `/var/log/fail2ban.log` (!!)
- Recherche des lignes comme `Ban W.X.Y.Z`

```
# Global settings
bantime = 600
findtime = 600
maxretry = 5

[recidive]
logpath = /var/log/fail2ban.log
banaction = %(banaction_allports)s
bantime = 604800 ; 1 week
findtime = 86400 ; 1 day
```

6 - Services et sécurité

Sécurité : modèle de menace

- De qui cherche-t-on à se protéger ?
 - Des acteurs gouvernementaux ? (NSA, Russie, Chine, ...)
 - Des attaques ciblées ? (DDOS, ransomware, espionnage économique)
 - Des attaques automatiques ? (bots)
 - De pannes systèmes ? (c.f. backups, résilience)
 - Des utilisateurs d'un site ? (injections, abus, ...)
 - Des collègues ?
 - ...

6 - Services et sécurité

Sécurité : modèle de menace

- Que cherche-t-on à protéger ?
 - Le front-end ?
 - L'accès aux serveurs ?
 - Des informations sur la vie de l'entreprise ?
 - Les infos personnelles des utilisateurs ?
 - L'intégrité et la résilience d'un système ?
 - Sa vie privée ? (historique de navigation, geolocalisation)
 - ...

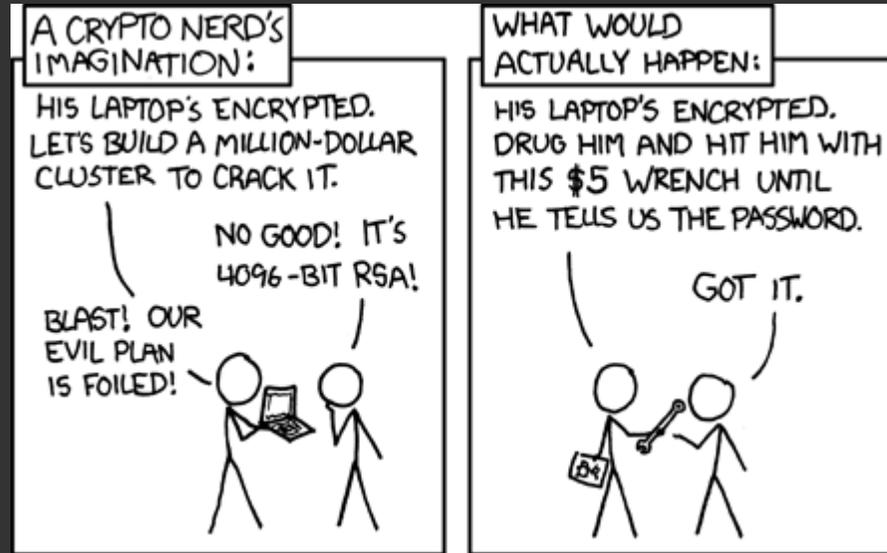
6 - Services et sécurité

Sécurité basique d'une machine (bureau, serveur)

1. Maintenir son système à jour
2. Minimiser la surface d'attaque
 - logiciels / apps installées
 - ports ouverts
 - permissions des utilisateurs et fichiers
 - accès physique
 - ...
3. Utiliser des mots de passe robustes (ou idéalement des clefs)
4. Utiliser des protocoles sécurisés

6 - Services et sécurité

6 - Services et sécurité



6 - Services et sécurité

Exemple de risque de sécurité subtil

Si on lance cette commande :

```
commande_complexe --argument --password "super_secret"
```

Le mot de passe `super_secret` sera visible par d'autres utilisateurs dans `ps -ef` ...!

7. Déployer un site "basique" avec nginx

7. Nginx

Généralités

- Un serveur web/HTTP "léger"
- Écoute sur le port 80 (et généralement 443 aussi si configuré pour HTTPS)
- Sert des pages web

Intérêt dans cette formation :

- manipuler un autre service
- rendre + utile/concret le fait d'avoir un serveur

7. Nginx

Configuration, logs

- `/etc/nginx/nginx.conf` : conf principale
- `/etc/nginx/sites-enabled/default` : conf du site par défaut
- `/var/log/nginx/access.log` : le log d'accès aux pages
- `/var/log/nginx/error.log` : les erreurs (s'il y'en a)

7. Nginx

/etc/nginx/sites-enabled/default

```
server {  
    listen 80 default_server;  
    listen [::]:80 default_server;  
  
    # [...]  
}
```

7. Nginx

Location blocks

```
location / {  
    alias /var/www/html/;  
}  
  
location /blog {  
    alias /var/www/blog/;  
}
```

En allant sur `monsite.web/blog`, on accédera aux fichiers dans `/var/www/blog/` (par défaut, `index.html` généralement)

7. Nginx

Location blocks

```
location / {  
    alias /var/www/html/;  
}  
  
location /blog {  
    alias /var/www/blog/;  
}  
  
location /app {  
    proxy_pass http://127.0.0.1:1234/;  
}
```

En allant sur `monsite.web/app`, nginx deleguera la requête à un autre programme sur la machine qui écoute sur le port 1234.

7. Nginx

nginx -t : verifier que la conf semble correcte

```
$ nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
```

(on peut ensuite faire **systemctl reload nginx** en toute sérénité)

7. Nginx

Fichier de log (access.log)

```
88.66.22.66 - - [10/Oct/2018:20:13:23 +0000] "GET / HTTP/1.1" 403 140 "-"
88.66.22.66 - - [10/Oct/2018:20:15:11 +0000] "GET / HTTP/1.1" 200 57 "-"
88.66.22.66 - - [10/Oct/2018:20:15:14 +0000] "GET /test HTTP/1.1" 301 18
88.66.22.66 - - [10/Oct/2018:20:15:15 +0000] "GET /test/ HTTP/1.1" 200 5
```

7. Nginx

Fichier d'erreurs (`error.log`)

(Exemple)

```
2018/10/10 09:06:44 [error] 28638#28638: *851331 open() "/usr/share/nginx/
```

(ACHTUNG : quand on débogue, toujours comparer l'heure actuelle du serveur à l'heure des erreurs pour vérifier quand elles ont eu lieu !)