Administration système

M4101C

2ème année - S4, cours - 2/3
2017-2018

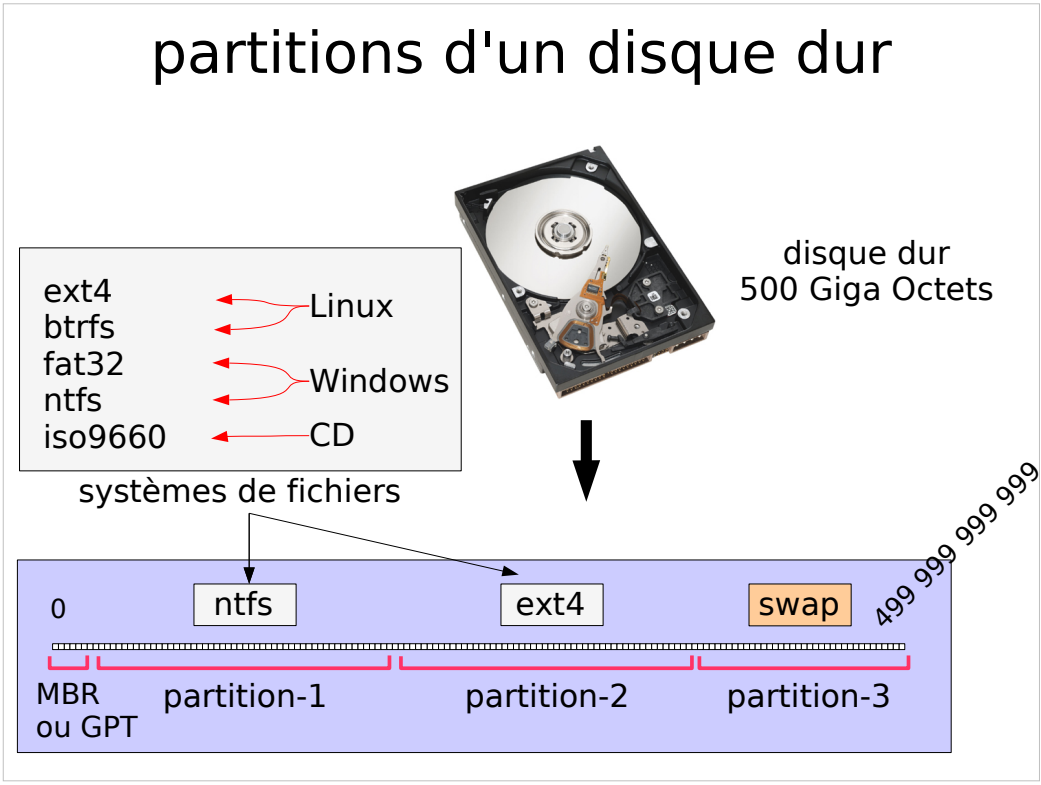
Marcel.Bosc@iutv.univ-paris13.fr

Département informatique IUT de Villetaneuse Université Paris-13

table des matières

1ère partie

gestion des disques



Un disque dur est vu par le système tout simplement comme une série d'octets. Dans cet exemple: de l'octet 0 à 499 999 999 999

Cet espace doit être découpé en plusieurs morceaux : les « partitions ». Chaque partition peut ensuite être "formatée" avec un système de fichiers. Par exemple ntfs pour Windows et ext4 pour Linux.
Au début du disque se trouvent des informations importantes:
- la "table des partions", qui décrit comment le disque est découpé en partitions.
- le boot-loader (amorçage), petit programme qui permet de démarrer le système souhaité.
Le format est soit MBR (Master Boot Record - ancien, mais très utilisé), soit GPT (plus récent)

noms de périphériques Linux

SATA / USB

/dev/sda
/dev/sdb
/dev/sdc
/dev/sdd
...

partitions: 1,2,3 ...
/dev/sda1, /dev/sda2, /dev/sda3 ...

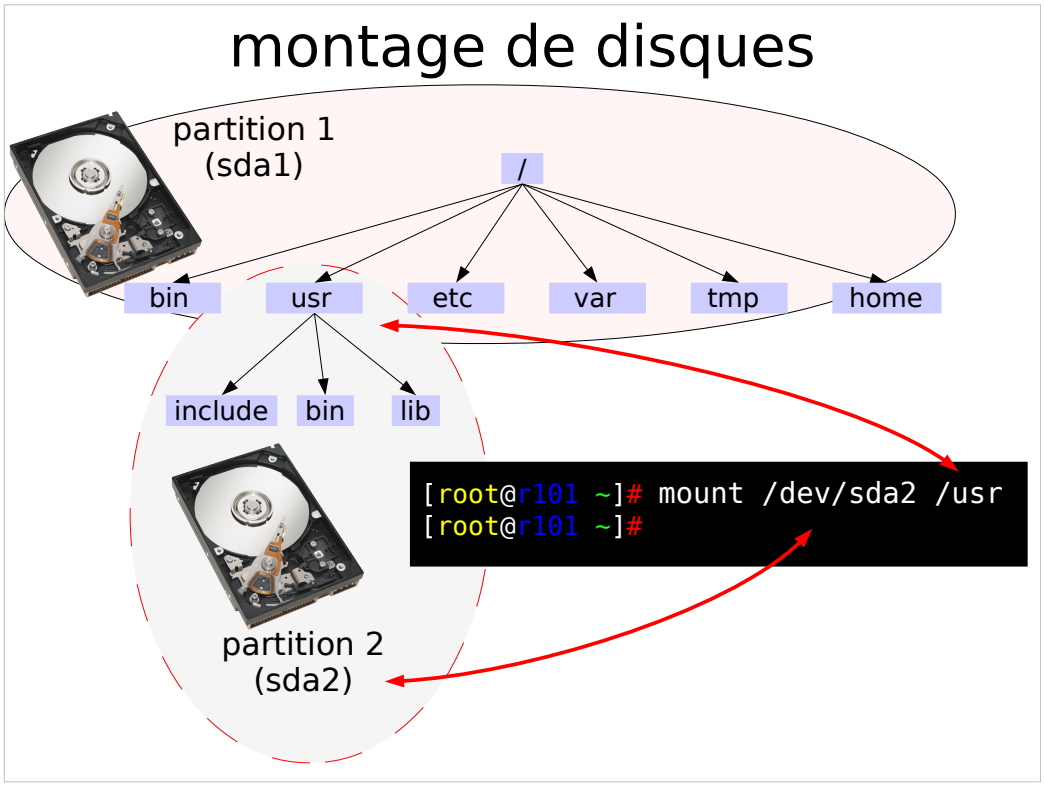
UUID: 31767484-0378-418d-b08e-20e603edab44
identifiant unique

dans le répertoire : /dev

Dans Linux on accède au matériel (périphériques) à travers des fichiers spéciaux qui se trouvent tous dans le répertoire /dev

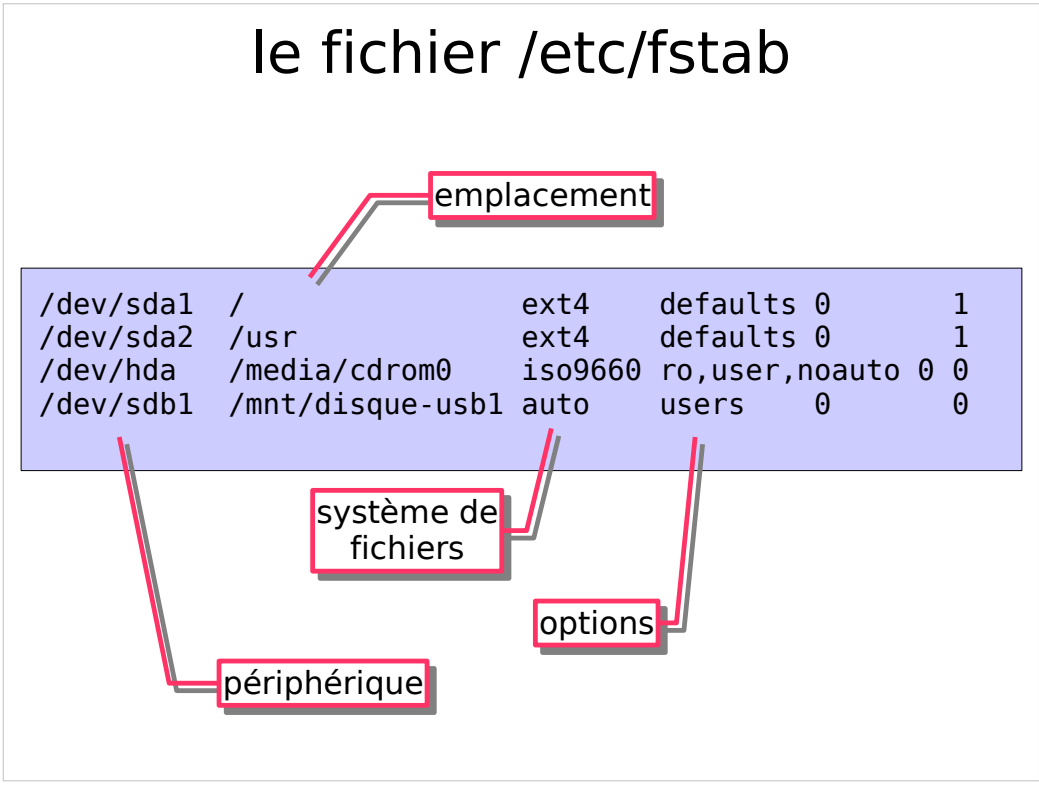
Les disques durs utilisent généralement une interface appelée "SATA".
Les fichiers spéciaux correspondant s'appellent /dev/sda pour le premier disque dur, /dev/sdb pour le deuxième, etc.
Chaque disque dur peut être découpé en partitions: /dev/sda1 /dev/sda2 ... /dev/sdb1 ...

On utilise aussi des identifiants uniques : une long série de chiffres hexadécimaux qui identifie de manière unique chaque disque dur.



Pour pouvoir accéder aux fichiers qui se trouvent dans une partition, il faut associer cette partitions à un répertoire à l'aide de la commande "mount".

Cet exemple associé la 2e partition du premier disque dur au répertoire /usr :
mount /dev/sda2 /usr
C'est à dire : quand on va dans /usr on accède en réalité à la partition /dev/sda2 (et pas réellement au répertoire /usr qui se trouve lui sur /dev/sda1)



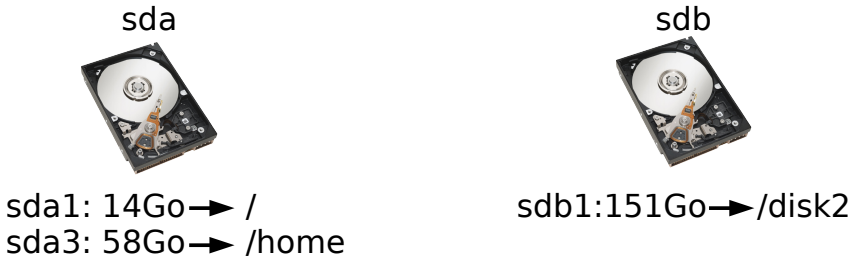
Lors du démarrage, le système a besoin d'accéder aux fichiers. On ne peut pas taper, à la main, les commandes "mount" à chaque démarrage...

Les points de montage sont donc spécifiés dans un fichier : /etc/fstab

Chaque ligne correspond à une partition (ou parfois à un périphérique complet).

la commande « df »

```
[dupond@r10102 ~]# df -h
Sys. de fich.  Tail.  Occ.  Disp.  %Occ.  Monté sur
/dev/sda1       14G   6,7G   6,5G   51%   /
/dev/sda3       58G    46G   9,3G   84%   /home
/dev/sdb1      151G    97G    47G   68%   /disk2
[dupond@r10102 ~]#
```

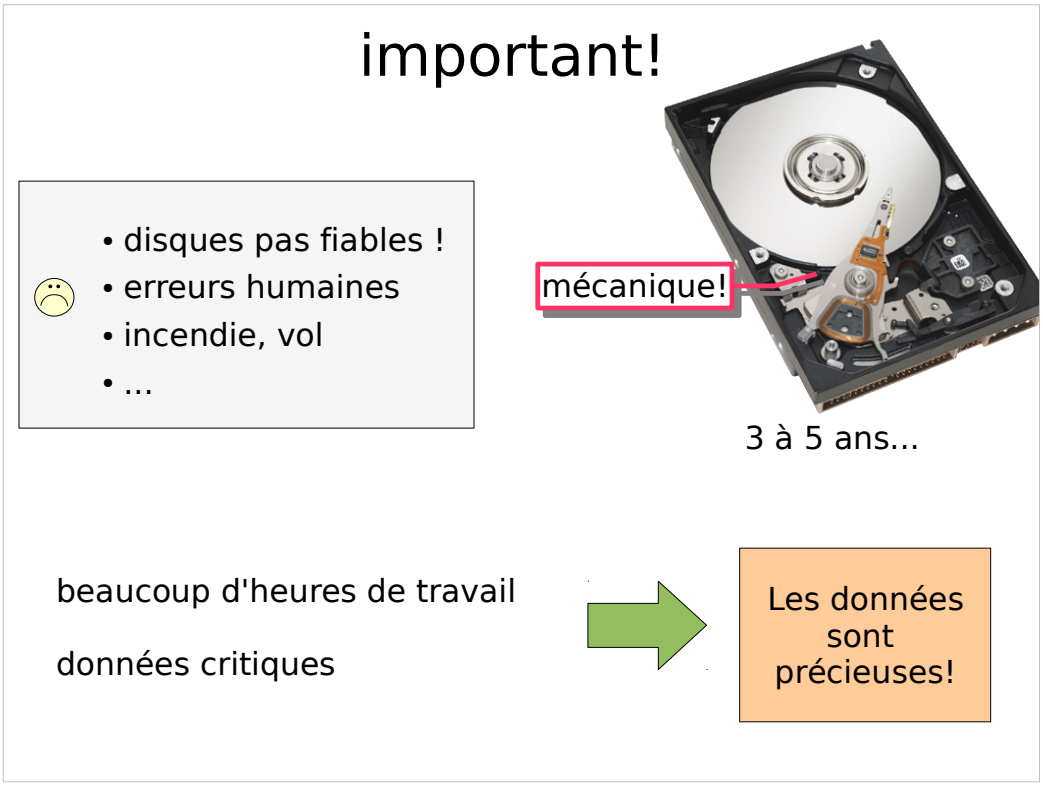


La commande "df", très utilisée, permet de regarder l'état de chacun des systèmes de fichiers montés sur le système.
La commande permet aussi de voir l'espace disponible sur chacun de ces systèmes de fichier.

Les périphériques qui ne sont pas montés ne sont pas visibles (utiliser sudo fdisk -l)

3ème partie

données et sauvegardes

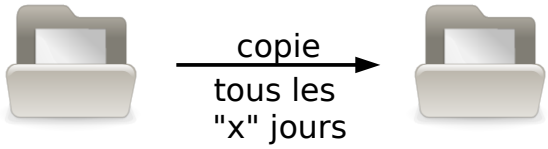


Chaque disque contient une grande quantité de données. Dans presque tous les cas, leur perte représente presque une très grande quantité de travail. Parfois elles sont critiques ou irremplaçables.
Les disques durs sont fragiles (consommables). Les erreurs humaines, les virus, les vols, etc... peuvent faire perdre ces données.
Il est indispensable de prévoir systématiquement un système de sauvegarde.
Toute donnée non sauvegardée doit être considérée comme jetable.
C'est vrai dans un cadre professionnel. C'est aussi vrai pour vos données personnelles.

Il existe de nombreuses approches pour faire des sauvegardes. Voici quelques critères:

facile à faire : si chaque sauvegarde demande du travail, l'expérience montre qu'au bout d'un temps, elles ne seront plus faites.
semi-automatique: il est donc indispensable de les automatiser, tout en gardant une vérification humaine.
régulière: plus on attend entre chaque sauvegarde, plus on perd potentiellement de données.
lieu séparé : si le système de sauvegarde se trouve dans le même lieu que les données d'origine, on risque de perdre les deux en même temps (exemple : incendie, vol)

sauvegarde par duplication



pas d'historique

pertes de donnée pas remarquée => définitive !!

procédure très simple

équipement simple

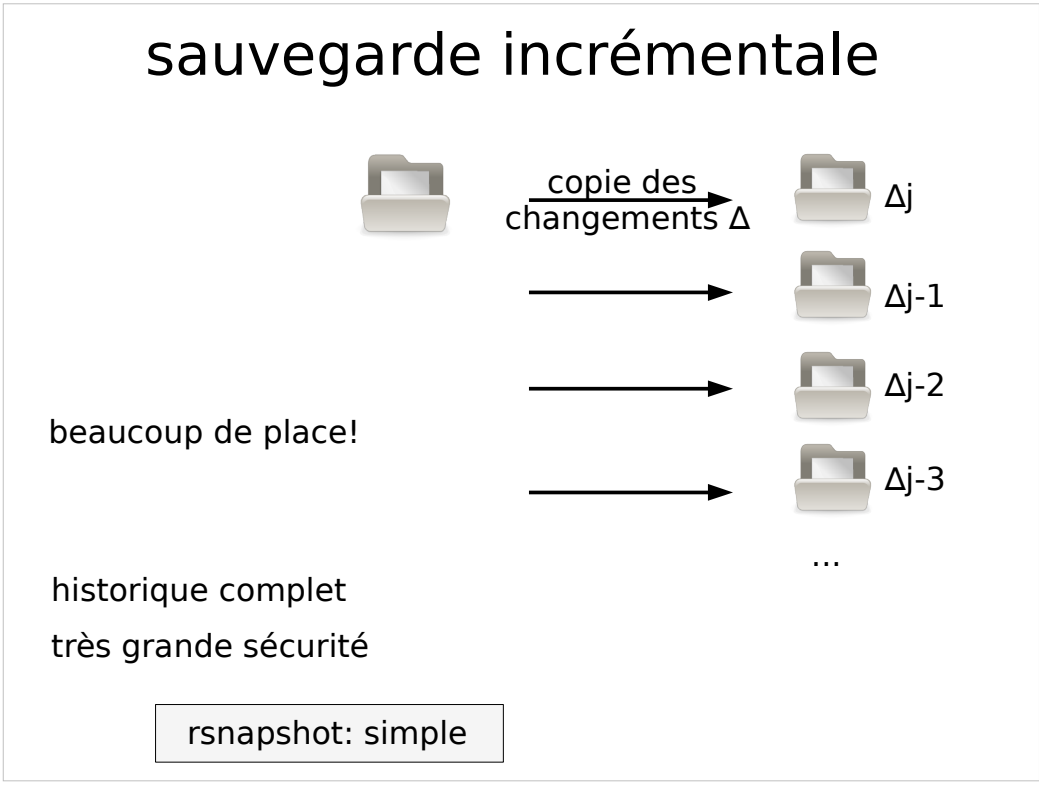
Un exemple de sauvegarde simple est de dupliquer les données.

Par exemple: chaque jour, automatiquement, un disque dur est entièrement copié vers un autre qui se trouve dans un local distant.

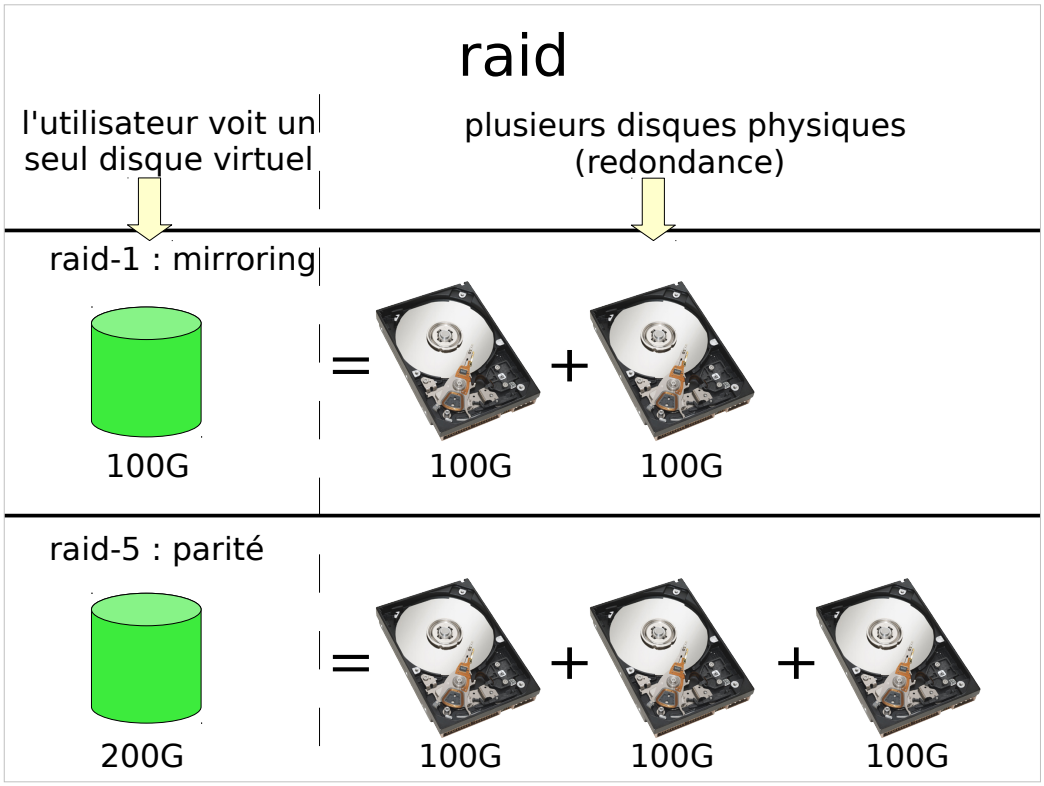
Le disque du local distant est réutilisé chaque jour: les données précédentes sont écrasées.

En cas de panne du premier disque , on retrouve les données dans le 2e.

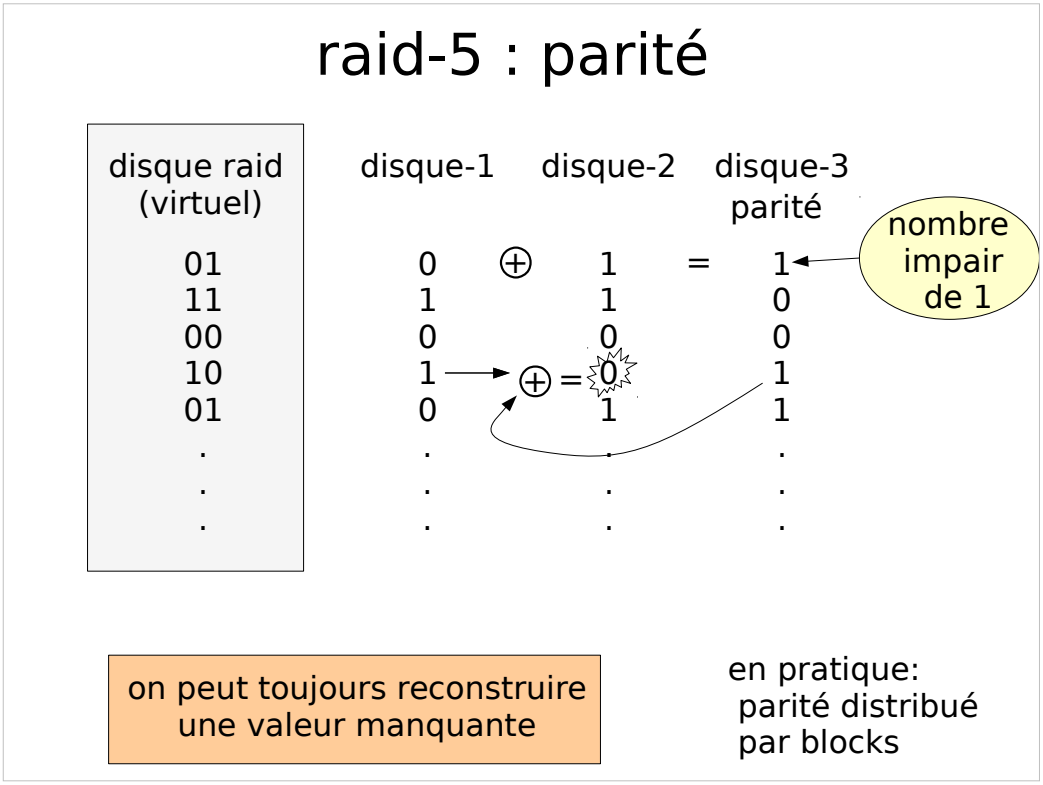
En cas de perte de données non remarquée (fichiers effacés par erreur, piratage...), les données sont perdues définitivement : il n'y a pas d'**historique**.



Le système précédent est insuffisant.
Un système de sauvegarde doit conserver un historique, permettant de remonter dans le temps pour retrouver les données telles qu'elles étaient avant leur perte.
Par exemple, si on copiait chaque jour un disque dur en entier vers une sauvegarde le système prendrait trop de place.
Les sauvegardes incrémentales permettent de ne conserver que ce qui a changé entre deux copies.



Le RAID est un système qui permet de dupliquer, en temps réel (immédiatement) des données entre plusieurs disques.
L'utilisateur ne voit qu'un seul disque "virtuel", mais les données sont physiquement reparties sur plusieurs disques physiques.
Exemple: RAID-1 (très simple) : Les deux disques physiques sont en permanence identiques. Tout écriture sur le disque "virtuel" se fait immédiatement sur les deux disques physiques.



Le RAID-5 (et d'autres) utilise un système de parité.
Dans cet exemple il y a 3 disques physiques, mais on pourrait en avoir plus.
Pour simplifier, on peut imaginer que les données sont reparties sur tous les disques sauf le dernier. Le dernier sert alors de parité. Si un disque tombe en panne, on peut retrouver ses données, grâce à la parité.

raid : utilisation

⚠ **le raid n'est pas une sauvegarde** ⚠

il n'y a pas d'historique

panne disque

interruption de service

performance

complexité = danger

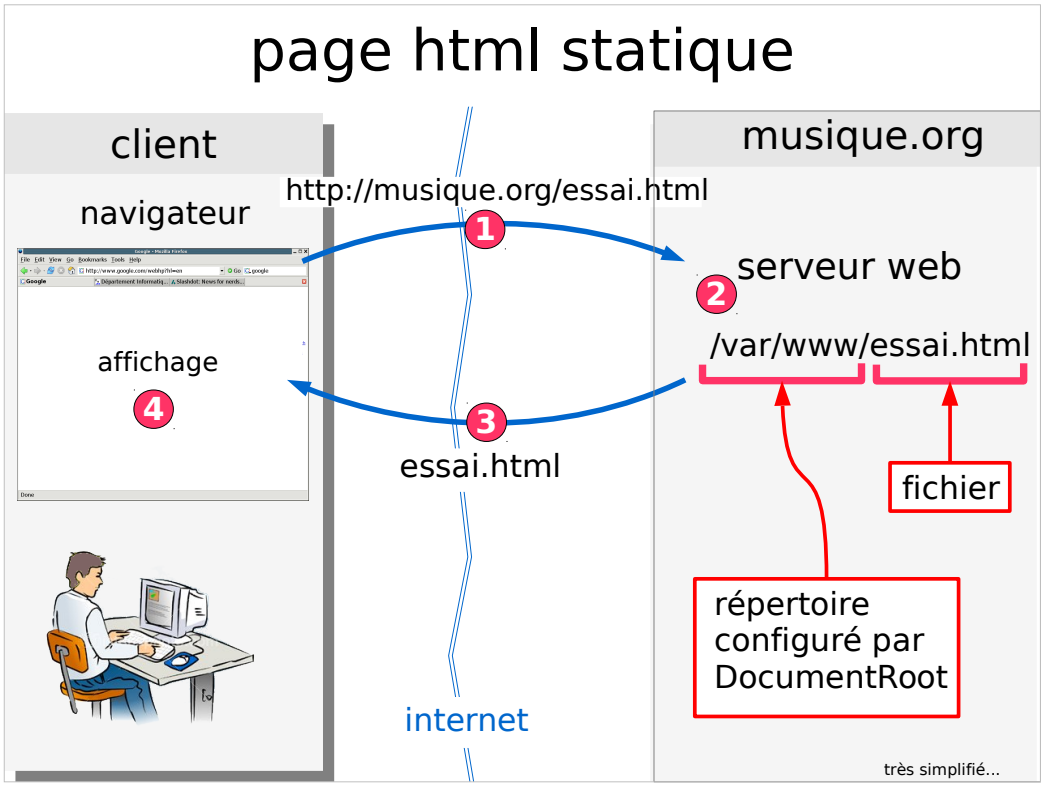
Dans le RAID, il n'y a **aucun historique**, une donnée supprimée est supprimée immédiatement sur les deux disques. Le RAID n'est donc PAS un système de sauvegarde.

En cas de panne d'un (seul) disque dur, le système peut continuer à fonctionner, **sans interruption**!

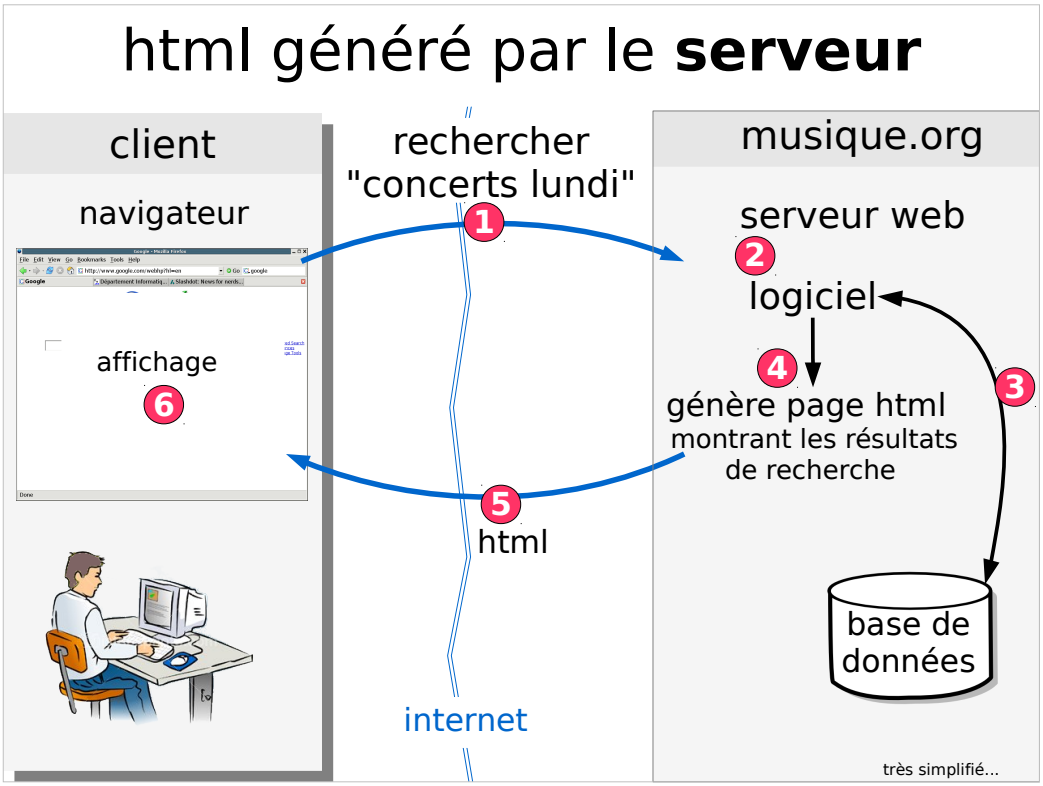
Le RAID reste complexe et son utilisation demande des compétences. Les pannes n'étant pas fréquentes, un administrateur peu habitué peut perdre toutes les données. Il est donc indispensable d'avoir un vrai système de sauvegarde, en plus du RAID.

3ère partie

rappels http - client serveur



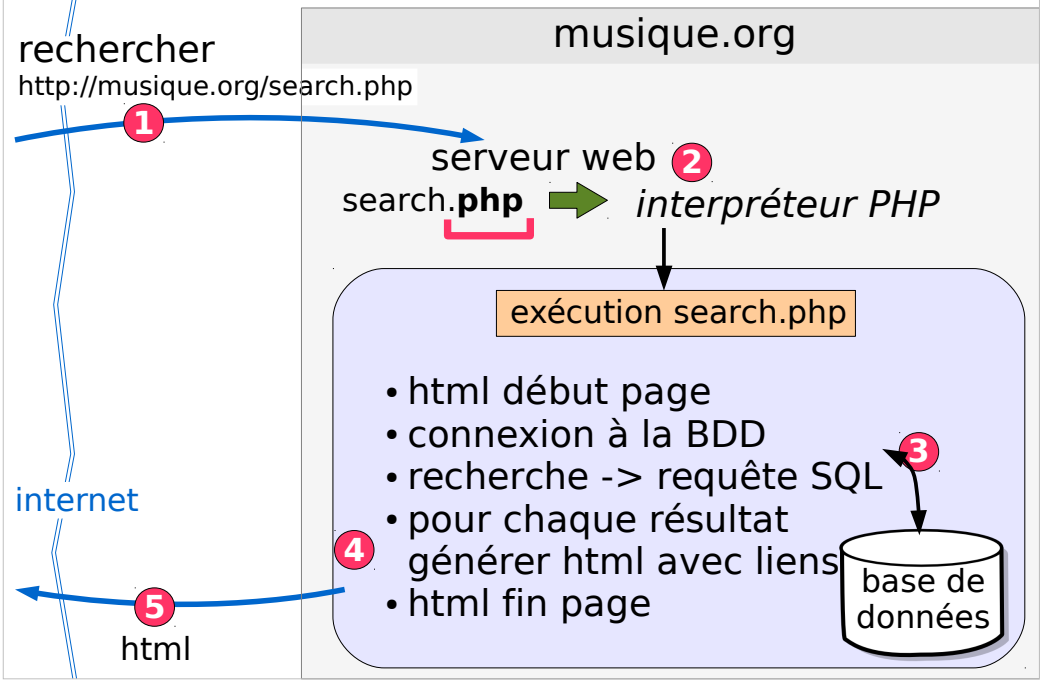
Dans la section suivante, on va aborder l'administration de serveurs web. Pour bien comprendre le contexte, on va revoir quelques principes vus en programmation web.
Exemple:
1) Un utilisateur ouvre une page dans son navigateur. Le navigateur demande au serveur de lui fournir la page "essai.html"
2) La page essai.html correspond à un fichier essai.html, le serveur va chercher ce fichier dans un répertoire. Le nom de ce répertoire (/var/www) se trouve dans la configuration du serveur web (Apache) sous une directive appelée "DocumentRoot"
3) Le serveur envoie le HTML au navigateur
4) Le navigateur affiche le HTML.

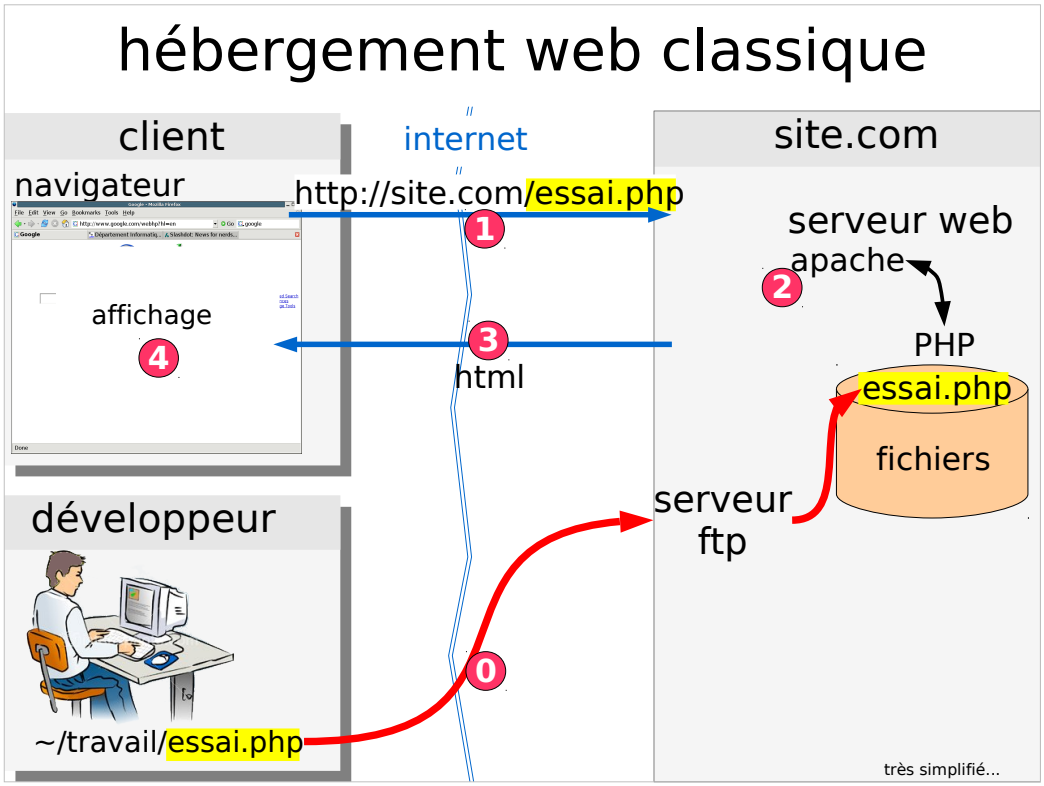


Dans beaucoup de cas, le HTML ne se trouve pas directement dans un fichier HTML sur le serveur, mais est le résultat de l'exécution d'un programme (par exemple PHP). Ce programme est exécuté par le serveur et peut éventuellement chercher des informations dans une base de données (par exemple MySQL).
Le HTML généré par le programme est envoyé au navigateur, qui l'affiche.

En TP, on va apprendre à configurer le serveur web Apache et à installer une base de données MySQL.

html généré par le **serveur**

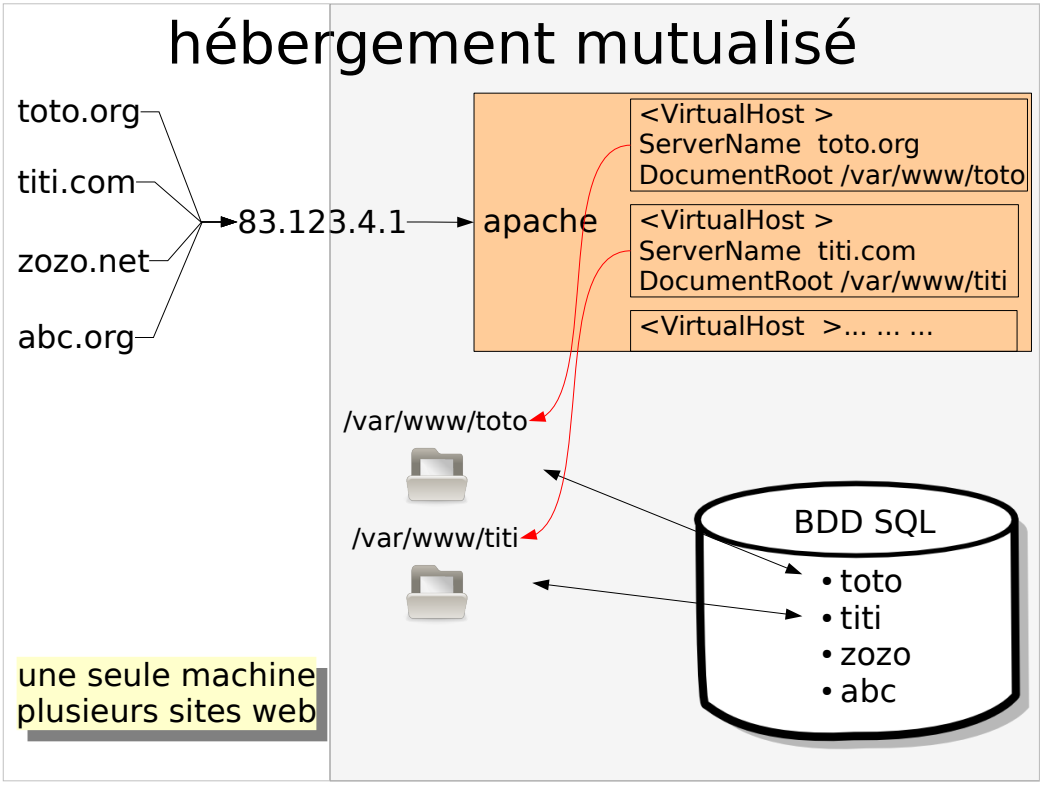




Les sites web sont souvent hébergés sur un serveur sur lequel le développeur web n'a qu'un accès limité (il n'est pas administrateur). Il peut alors déposer les fichiers de son site (par exemple HTML, CSS, PHP, …) à l'aide d'ftp ou sftp.

4ème partie

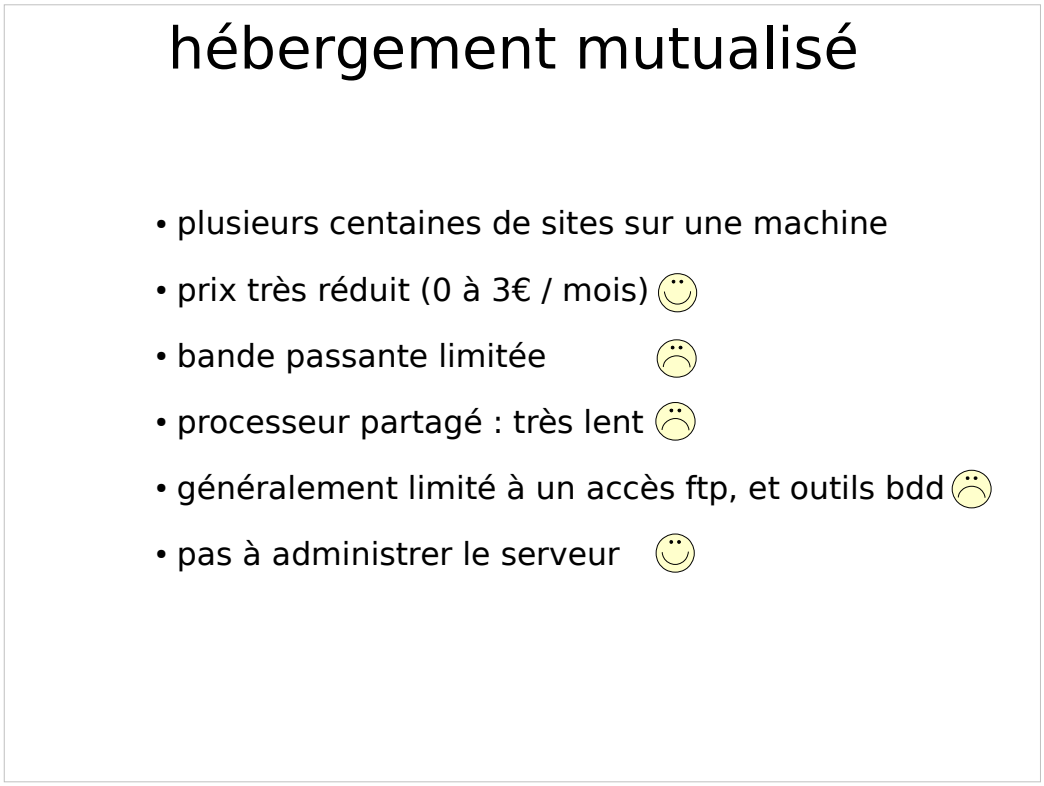
architectures web



Il existe des sites web de toutes les tailles. Les sites web très fréquentés (Google, Facebook...) ont besoin de centaines de milliers de machines.

A l'inverse, une seule machine peut facilement héberger un grand nombre de petits sites web. C'est qu'on appelle l'hébergement mutualisé.

Le serveur web (apache) distingue chaque site en fonction de son nom de domaine (dans l'URL). Chaque site a alors son propre fichier de configuration Apache « VirtualHost ». La directive DocumentRoot permet de donner à chaque site un répertoire différent pour ses fichiers.
On crée généralement une base de données différente pour chaque site.



De nombreuses entreprises (OVH, Online, 1&1, ...) ont des offres d'hébergement mutualisé.
Il est destiné a des petits sites web, ayant une fréquentation modérée. La qualité du service disponible (vitesse, outils, ...) est très variable en fonction des offres et du prix.

Avec hébergement mutualisé, l'administration du serveur est assuré par l'hébergeur. C'est un avantage important (pas besoin d'avoir des compétences en administration).
C'est aussi un problème : vous ne pouvez pas adapter la configuration du système à vos besoins.

serveur dédié

- location d'une machine hébergé dans un datacenter
- plus cher : 30€ / mois
- contrôle complet sur la machine
- administrateur système requis

De nombreuses entreprises (OVH, Soyoustart, Online, ...) ont des offres de serveurs dédiés. Un serveur dédié est une machine physique (un ordinateur) qui se trouve dans un datacenter, que vous louez. Vous n'avez pas d'accès physique à la machine. L'entreprise vous fourni des outils de gestion à distance (reboot, installation, …). Vous avez un contrôle complet (root) sur la machine. Vous devez donc avoir des compétences en administration système.

virtualisation

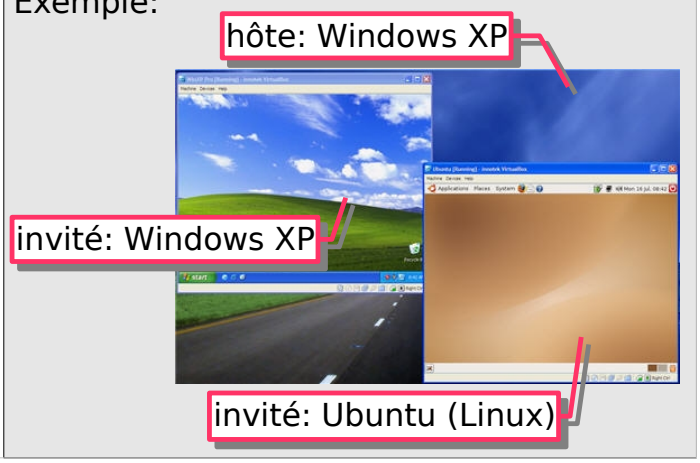
Virtualisation : exécuter un ou plusieurs systèmes **invités** dans un système d'exploitation **hôte**

très pratique!

logiciels: (bureau)
VirtualBox (libre!)
Parallels
VMware
...

logiciels: (serveur)
xen (libre)
kvm (libre)
VMware
[Docker]
...

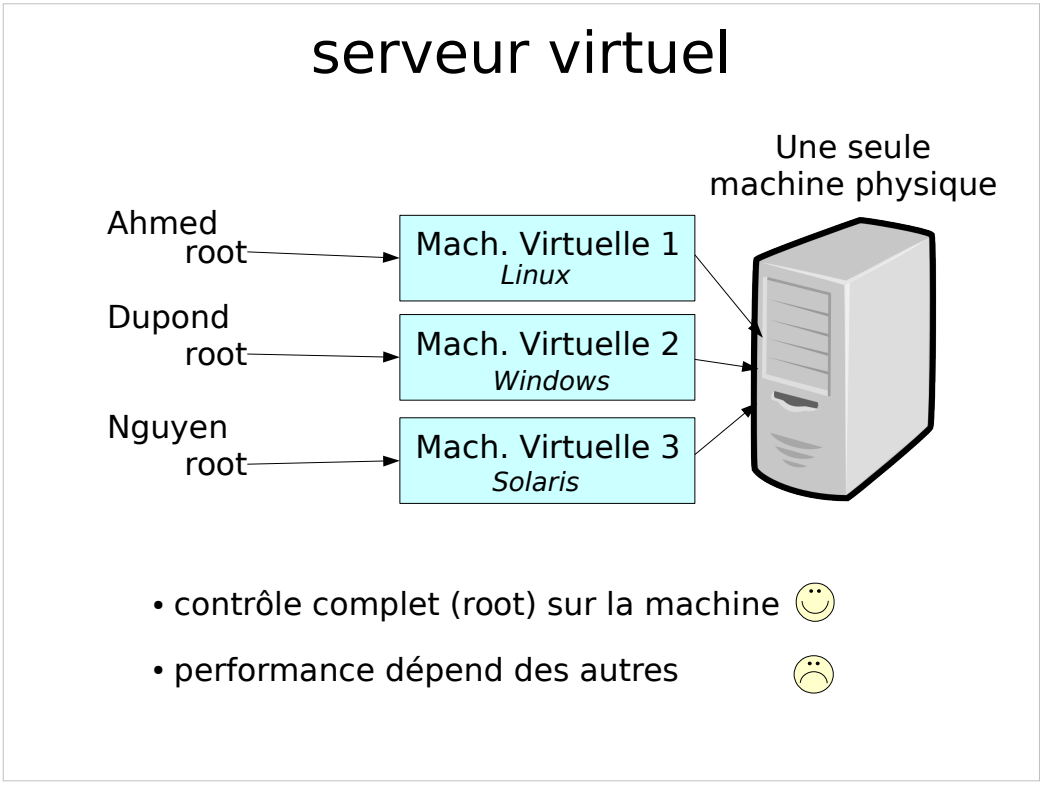
Exemple:



La Virtualisation permet d'exécuter un système d'exploitation à l'intérieur d'un autre.

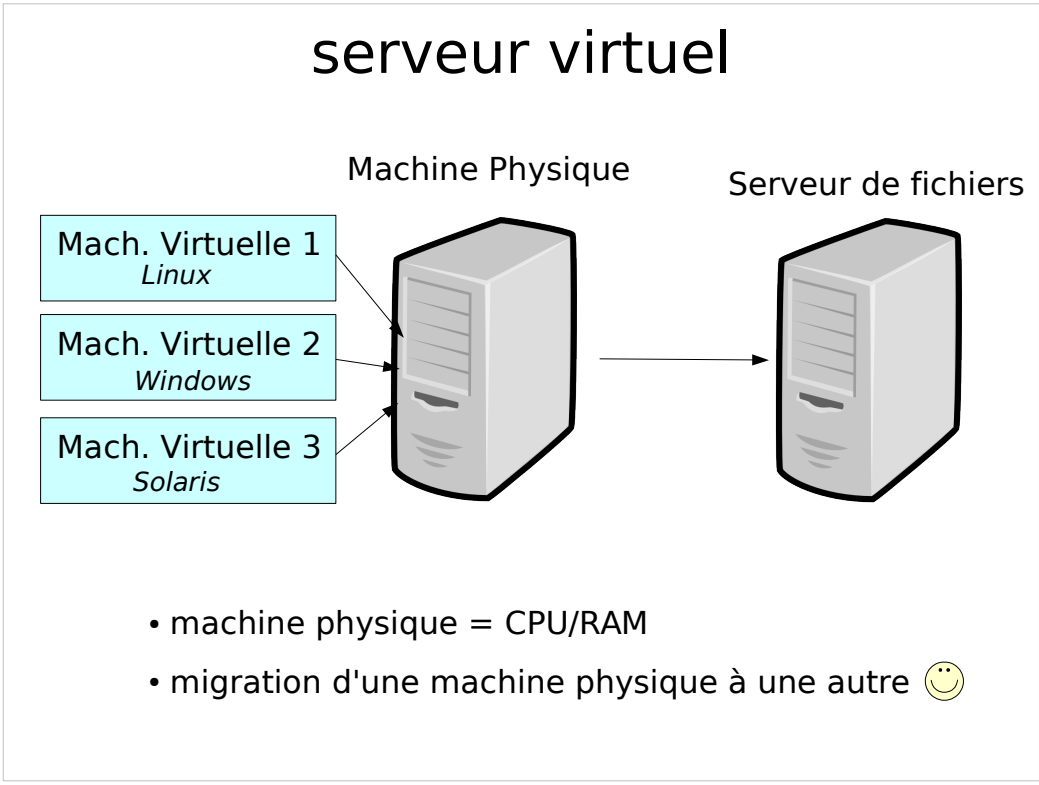
Par exemple, à l'intérieur de votre Windows, vous pouvez créer une « machine virtuelle » exécutant un système Linux complet. C'est pratique pour expérimenter sur une machine de bureau.

On peut aussi exécuter plusieurs systèmes Linux à l'intérieur d'un système Linux (hôte). Cette approche est très utilisée sur des serveurs.



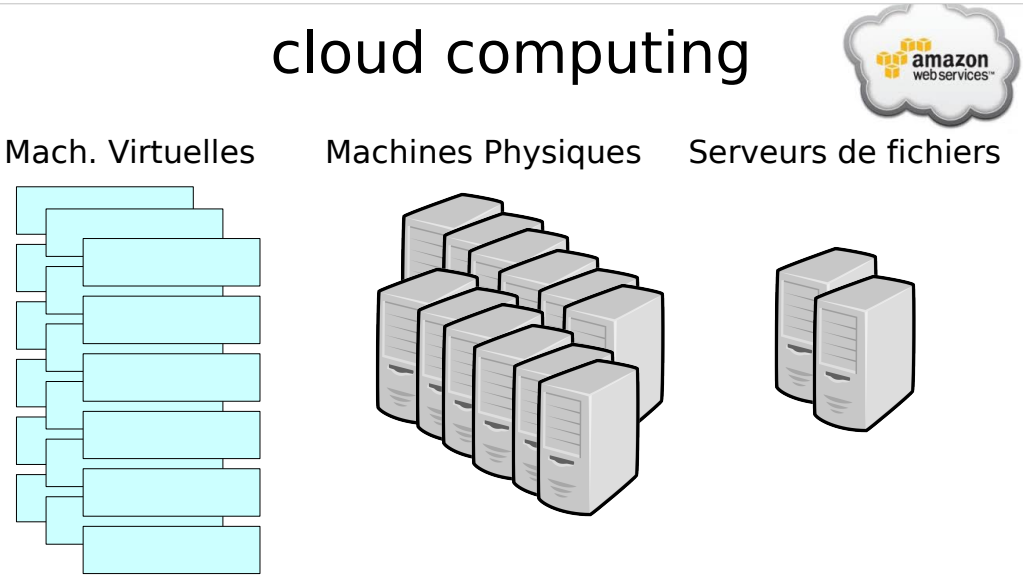
Dans l'hébergement mutualisé, vu précédemment, plusieurs utilisateurs géraient des sites web différents sur une même machine physique. Cette machine n'avait qu'un seul système d'exploitation. Si on leur donnait un accès root, ils pourraient détruire les sites des autres utilisateurs.

La virtualisation permet de créer plusieurs machines virtuelles dans une même machine physique. Chaque utilisateur peut avoir sa propre machine virtuelle, où il est root, sans empiéter sur les autres utilisateurs.



La virtualisation peut aller plus loin.
Au lieu d'utiliser les fichiers sur la machine physique, on peut les mettre sur un serveur de fichiers différent. De cette manière la machine physique ne fait que fournir du CPU et de la RAM. En cas de panne ou de maintenance, elle peut-être remplacée avec peu d'impact pour les utilisateurs.

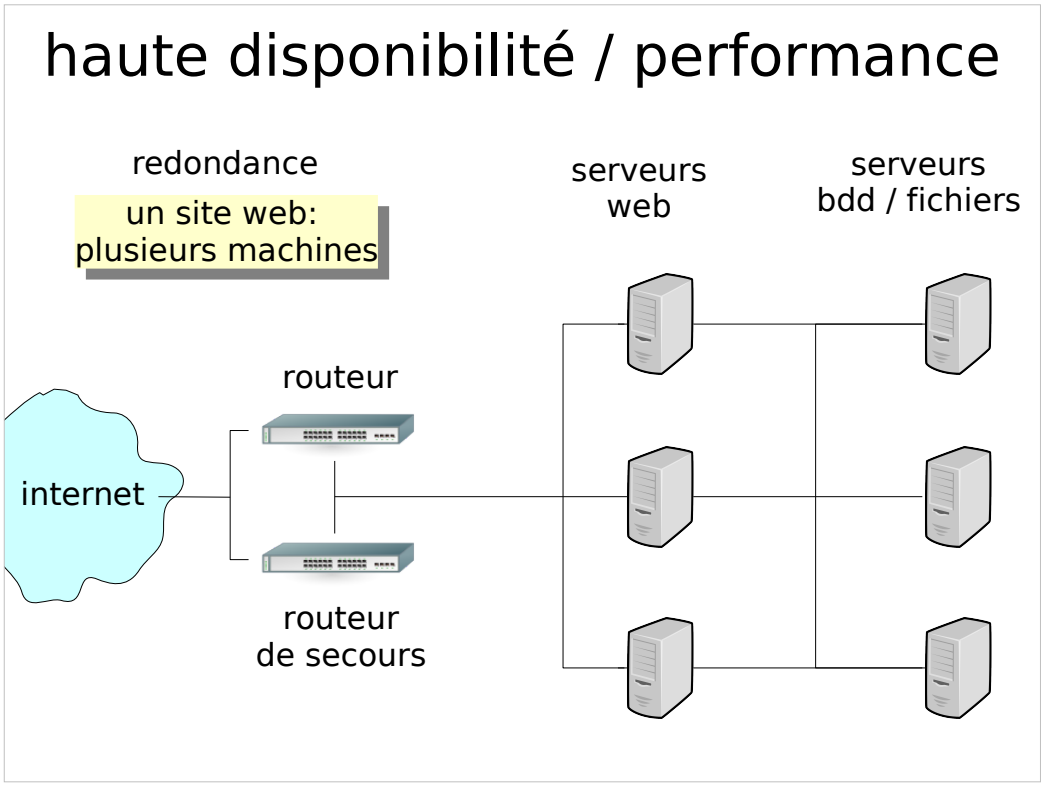
cloud computing



- très souple : changer la machine en un click
- moins sensible aux pannes matérielles
- en pratique : problèmes de performance, prix

Ce principe est très utilisé. Au lieu de louer un serveur dédié, on loue une machine virtuelle et de la place disque sur un serveur de fichiers. La machine virtuelle peut changer de machine physique, sans que ça affecte le service.
De nombreux hébergeurs proposent des services de ce type (OVH, Amazon EC2,…).

En pratique, le prix sont assez élevés et les performances pas toujours fiables.



Un site web important doit garantir sa disponibilité même en cas de panne. Il peut aussi faire face à un trafic trop élevé pour être géré par une seule machine physique.

Pour atteindre ces objectifs, il faut utiliser plusieurs machines. Des routeurs réseau vont alors distribuer le trafic sur plusieurs serveurs web (machines physiques). D'autres machines assureront les services supplémentaires (par exemple : base de données, ou fichiers).
La mise en place de ce type d'architectures complexes requiert des administrateurs système expérimentés.

5éme partie

Configuration Apache et sécurité

Que fait Apache ?

musique.org

https://musique.org/index.**html**

processus Apache

1) Utiliser config, VirtualHost
2) fichier HTML « statique » demandé
3) Envoyer simplement le fichier

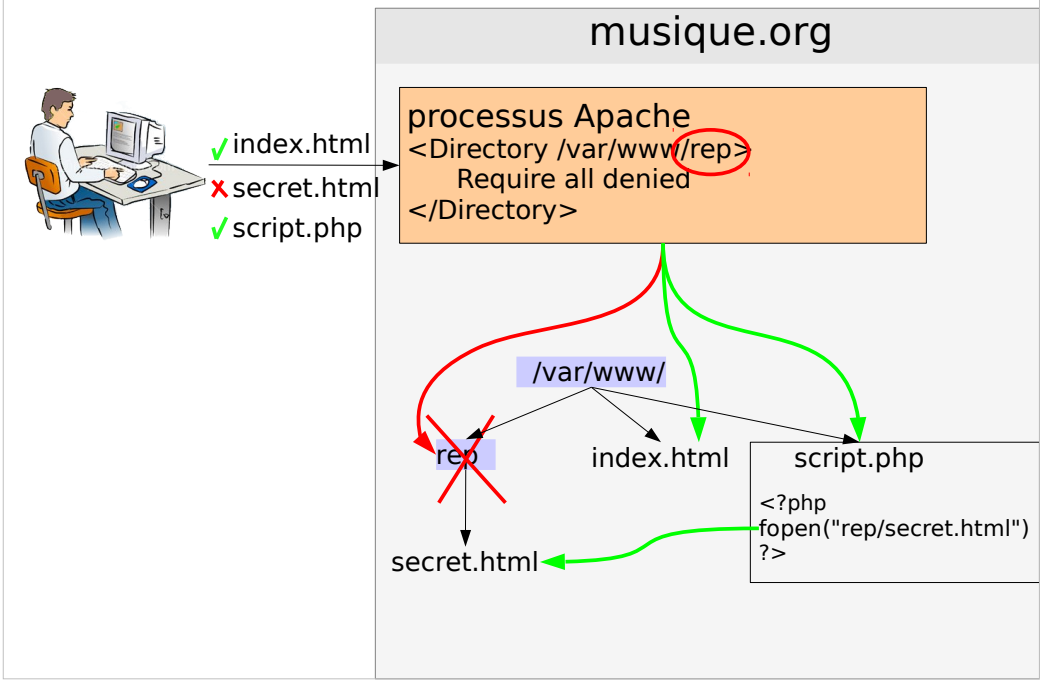
```
<VirtualHost >
   ServerName  musique.org
   DocumentRoot /var/www
   ...
```

https://musique.org/script.**php**

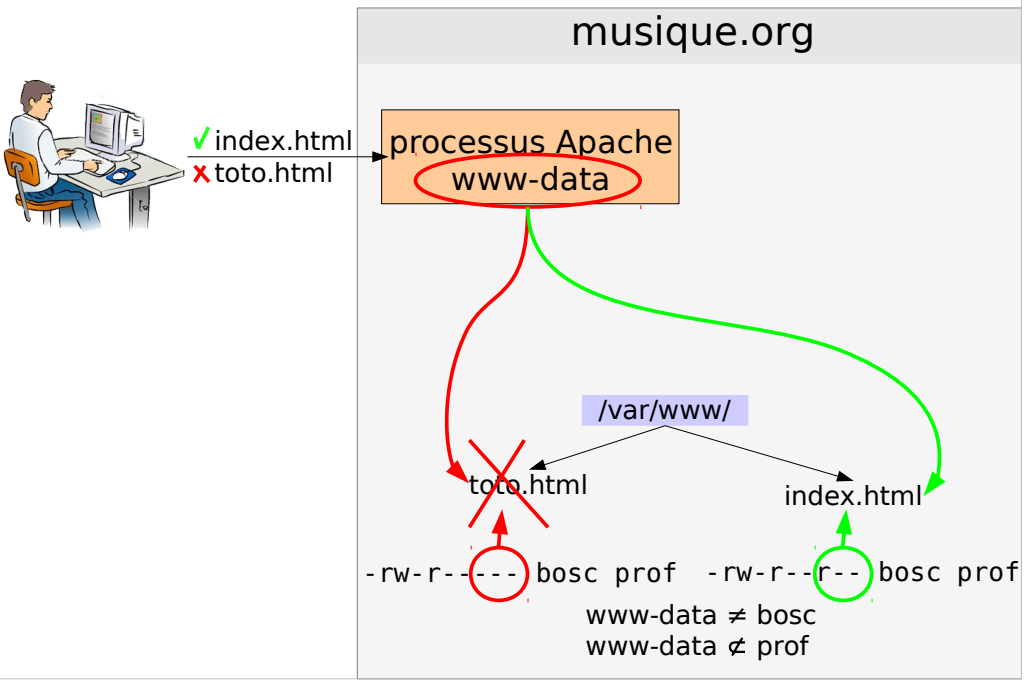
processus Apache

1) Utiliser config, VirtualHost
2) Fichier PHP « dynamique » demandé
3) Exécuter avec l'interpréteur PHP
4) Transmettre le HTML généré

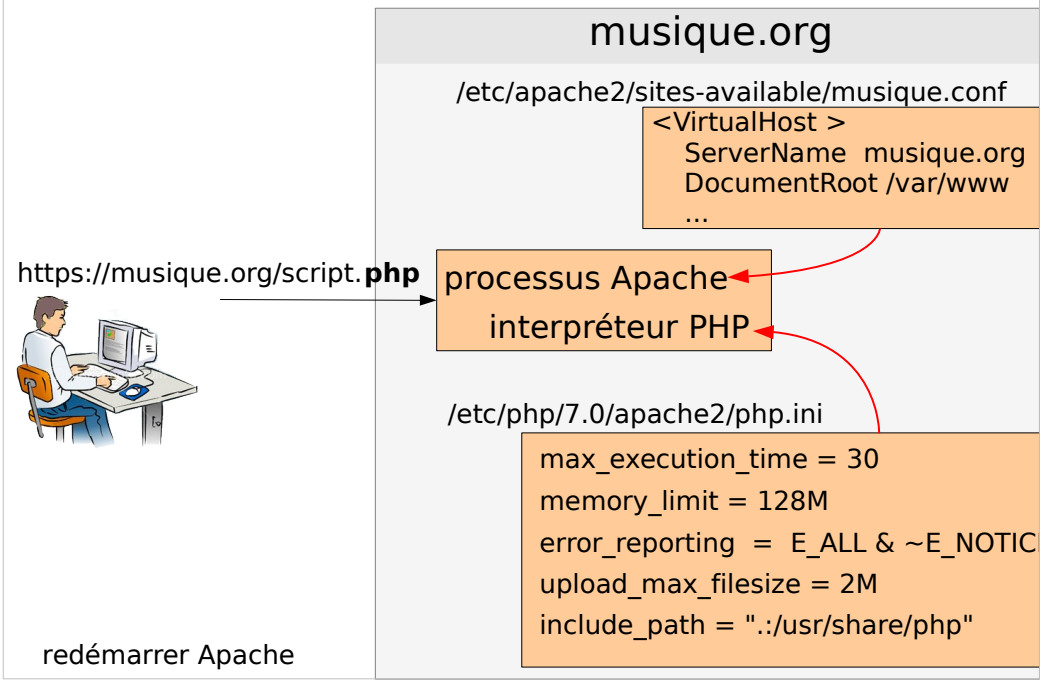
Contrôle d'accès Apache



Droits d'accès aux fichiers



Configuration du PHP

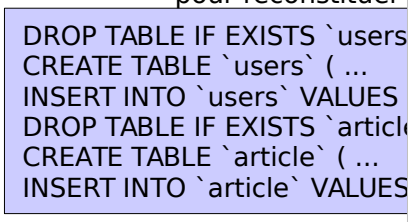


Administration MySQL

sauvegarde:

mysqldump -u toto -p mabase > mabase.sql

instructions SQL
pour reconstituer



restauration

mysql -u toto -p mabase < mabase.sql

création de bases

```
CREATE DATABASE mabase;
```

création d'utilisateurs sql et droits

```
GRANT ALL PRIVILEGES ON mabase.* TO toto@localhost
        IDENTIFIED BY 'motdepasse';
FLUSH PRIVILEGES;
```

Comptes et authentification

ne pas confondre:

- **comptes système**
  /etc/passwd
  /etc/shadow
  ex: "etudiant", www-data
- comptes à l'intérieur d'un **logiciel web**
  ex: Moodle : PHP, SQL
  table SQL avec login et mot de passe crypté
- **authentification apache**
  config Apache ex: .htaccess, .htpasswd
- **utilisateurs SQL**
  table mysql.user

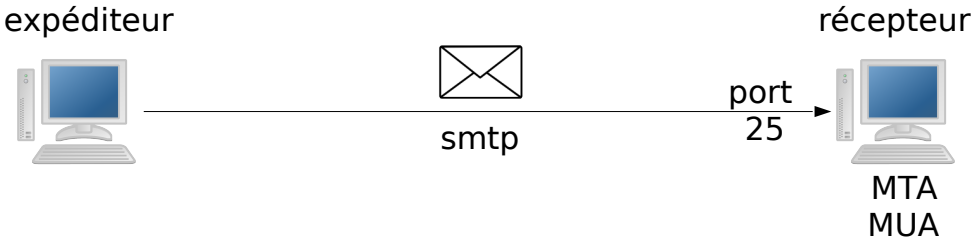
6éme partie

le courrier électronique

courrier : notions de base

- MUA : Mail User Agent
  thunderbird, webmail, outlook, ...
- MTA : Mail Transfer Agent
  sendmail, exim, postfix, exchange
- protocole SMTP

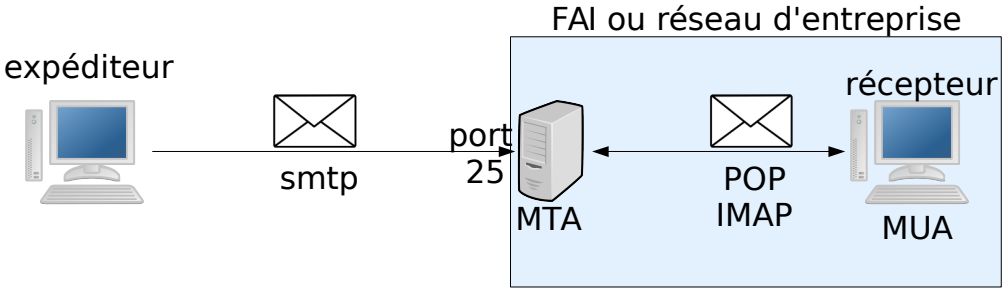
courrier : au plus simple



mais:

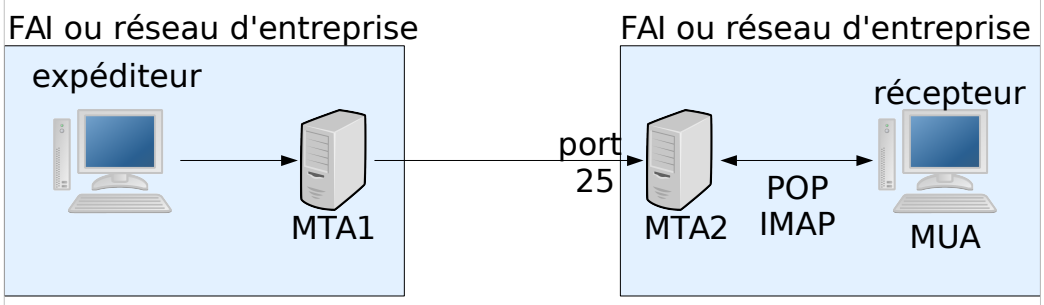
- récepteur doit rester connecté
- expéditeur doit rester connecté
- sécurité: pas de contrôle / filtrage

serveur de mail rentrant

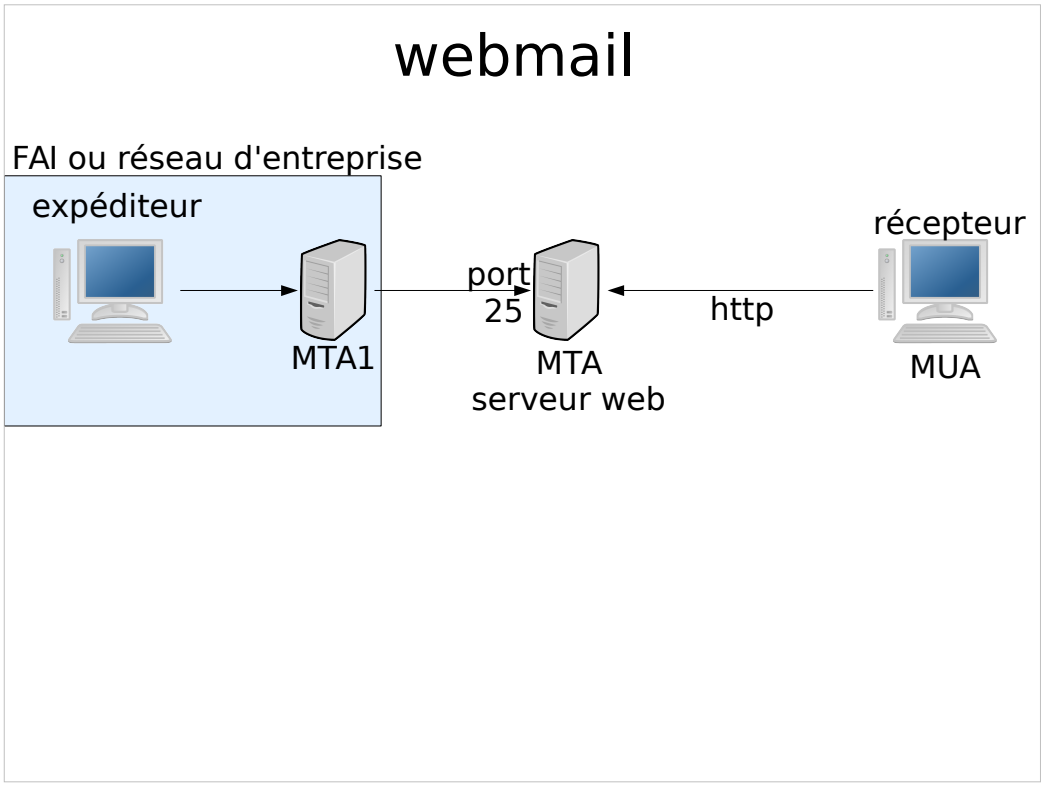


le serveur mail (MTA) stocke les mails
en attendant que le client (MUA)
vienne les demander par POP ou IMAP

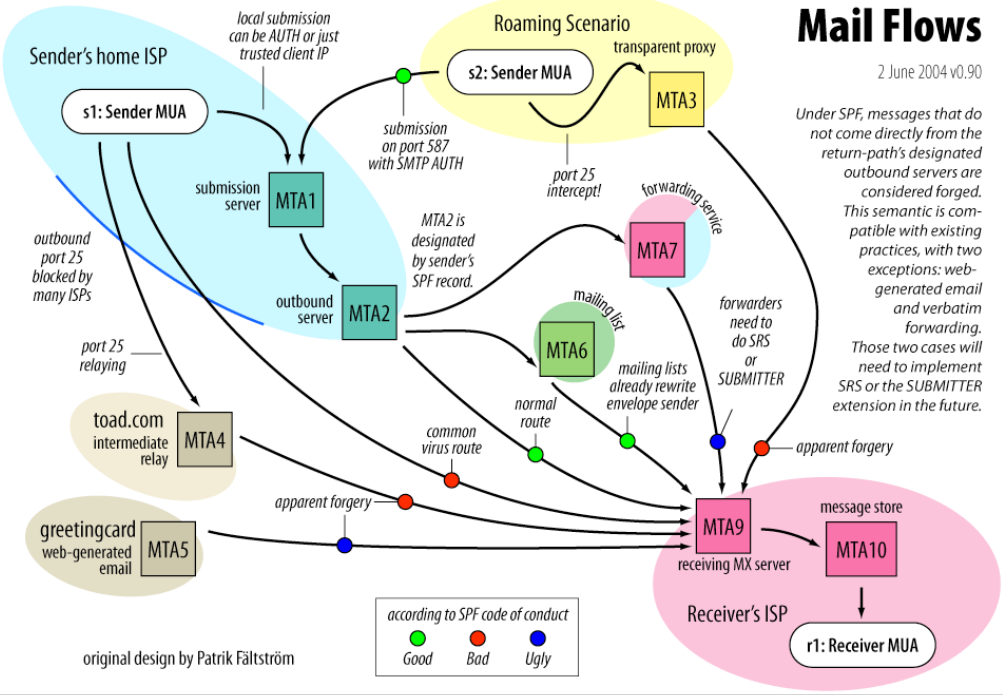
serveur de mail sortant



le serveur mail (MTA1) peut stocker les mails et ressayer jusqu'à ce que MTA2 les accepte


webmail
FAI ou réseau d'entreprise
expéditeur
MTA1
port
25
MTA
serveur web
http
récepteur
MUA

courrier : scénarios réalistes



Marcel.Bosc@iutv.univ-paris13.fr