

Applications de l'Internet

Partie 2 — Applications Client/Serveur

Université Claude Bernard Lyon 1

Sources et bibliographie

- D'après le cours d'Olivier Glück (Université Lyon 1)
- Figures et tableaux repris des supports d'origine
- RFC (rfc-editor.org) et documentation des protocoles

Plan de la partie 2

- Introduction / Rappel
- Applications de connexion à distance (SSH, X)
- Transfert de fichiers (FTP, TFTP, NFS, SMB)
- Annuaire : NIS, DNS, LDAP
- Messagerie électronique (SMTP, IMAP, MIME)

Plan de la partie 2

- **Introduction / Rappel**
- Applications de connexion à distance (SSH, X)
- Transfert de fichiers (FTP, TFTP, NFS, SMB)
- Annuaire : NIS, DNS, LDAP
- Messagerie électronique (SMTP, IMAP, MIME)

Introduction / Rappels

La couche application

- Gère les logiciels applicatifs, au-dessus des couches inférieures
- Repose sur le mode Client/Serveur
- Distinguer l'application et le protocole applicatif
 - Le protocole applicatif définit les échanges entre les parties cliente et serveur de l'application
 - Une interface (API) permet au protocole applicatif d'utiliser les services de bout-en-bout fournis par un protocole de transport sous-jacent
 - Le protocole définit le format des messages et l'ordre des échanges

Quel service de transport ? - les critères

- Pertes : l'application tolère-t-elle des pertes ? (fiabilité)
- Débit : besoin d'une bande passante minimale ?
- Bande passante : Quelle est la taille minimale du tuyau de communication ?
- Délai : contrainte temps réel (téléphonie, jeux, visio) ?
- Ces critères orientent le choix TCP ou UDP

Quel service de transport ? - Fiabilité

- Certaines applications nécessitent une fiabilité à 100%
 - Courrier électronique (SMTP)
 - Transfert de fichiers (FTP)
 - Accès distant (Telnet)
 - Transfert de documents Web (HTTP)
 - Applications financières
- D'autres peuvent tolérer des pertes (loss-tolerant applications)
 - Applications multimédia : audio/vidéo (une perte d'une faible quantité de données n'induit qu'une petite irrégularité dans l'écoute ou la vision du film)

Quel service de transport ? – Bande Passante

- Certaines applications requièrent une bande passante minimale
 - Téléphonie sur Internet : si la voix est codée à 32 Kbps, les données doivent être transmises et reçues à ce débit
 - Applications multimédia
- D'autres utilisent la bande passante disponible (applications élastiques)
 - Courrier électronique, transfert de fichiers, accès distant, Web
 - Plus il y a de bande passante, mieux c'est !

Quel service de transport ? – Délai

- Certaines applications nécessitent un délai de bout-en-bout faible (moins de quelques centaines de ms)
 - Applications temps réel interactives :
 - Téléphonie sur Internet
 - Environnements virtuels
 - Téléconférence
 - Jeux en réseau
- Pour les applications non temps réel, un délai court est préférable, mais pas de contrainte forte

Sensibilité des applications au transport

Application	Pertes	Bande passante	Sensibilité temp.
Transfert de fichiers	sans perte	élastique	Non
e-mail	sans perte	élastique	Non
Pages Web	sans perte	élastique	Non
Audio/vidéo temps réel	tolérant	audio: 5Kb - 1Mb vidéo: 10Kb - 5Mb	Oui, 100's ms
Audio/vidéo enregistré	tolérant	idem	Oui, quelques s
Jeux interactifs	tolérant	quelques Kbps	Oui, 100's ms
Applis financières	sans perte	élastique	Oui et non

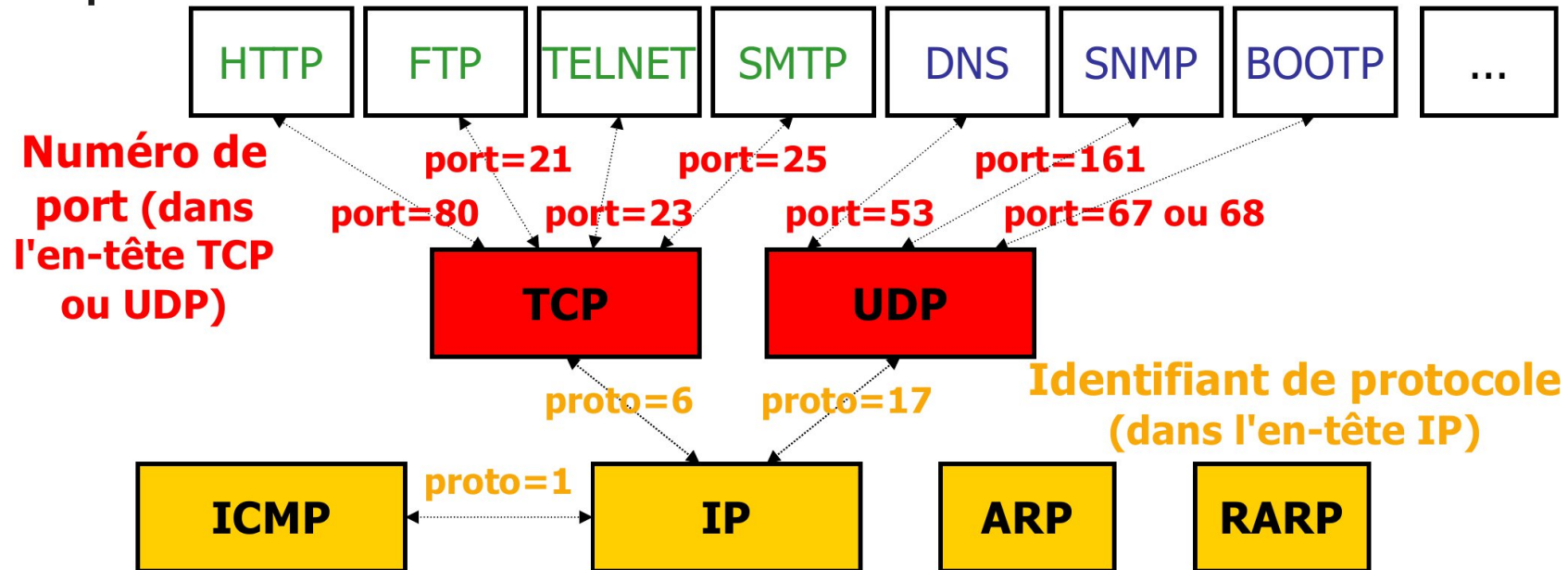
Quel service de transport ? - TCP ou UDP

- TCP : orienté connexion, fiable, contrôle de flux et de congestion
 - orienté connexion : connexion nécessaire entre le client et le serveur
 - transport fiable entre le processus émetteur et récepteur
 - contrôle de flot : l'émetteur ne submerge pas le récepteur
 - contrôle de congestion : réduit le débit de l'émetteur quand le réseau est congestionné
 - ne propose pas :
 - de garantie de délai,
 - de bande passante minimale
- TCP pour la fiabilité (fichiers, mail, web)
- UDP : sans connexion, sans garantie, simple et rapide
 - transfert de données non fiable
 - ne propose pas
 - de connexion,
 - de fiabilité,
 - de contrôle de flot,
 - de contrôle de congestion,
 - de garantie temporelle,
 - de bande passante
 - beaucoup plus simple que TCP (UDP=IP) donc plus rapide
 - pas de limitation du débit
- UDP pour la latence
- QUIC : transport moderne sur UDP (fiable + chiffré, base de HTTP/3)

Principales applications de l'Internet

- Chaque application s'appuie sur un protocole applicatif et un transport
- Identifiée par un numéro de port (bien connu si < 1024)
- Exemples : HTTPS (443), DNS (53), SMTP (25), SSH (22)
- Référence : IANA (numéros de ports)

Applications, ports et protocoles



Tous les ports sur
<http://www.iana.org/assignments/port-numbers>

Plan de la partie 2

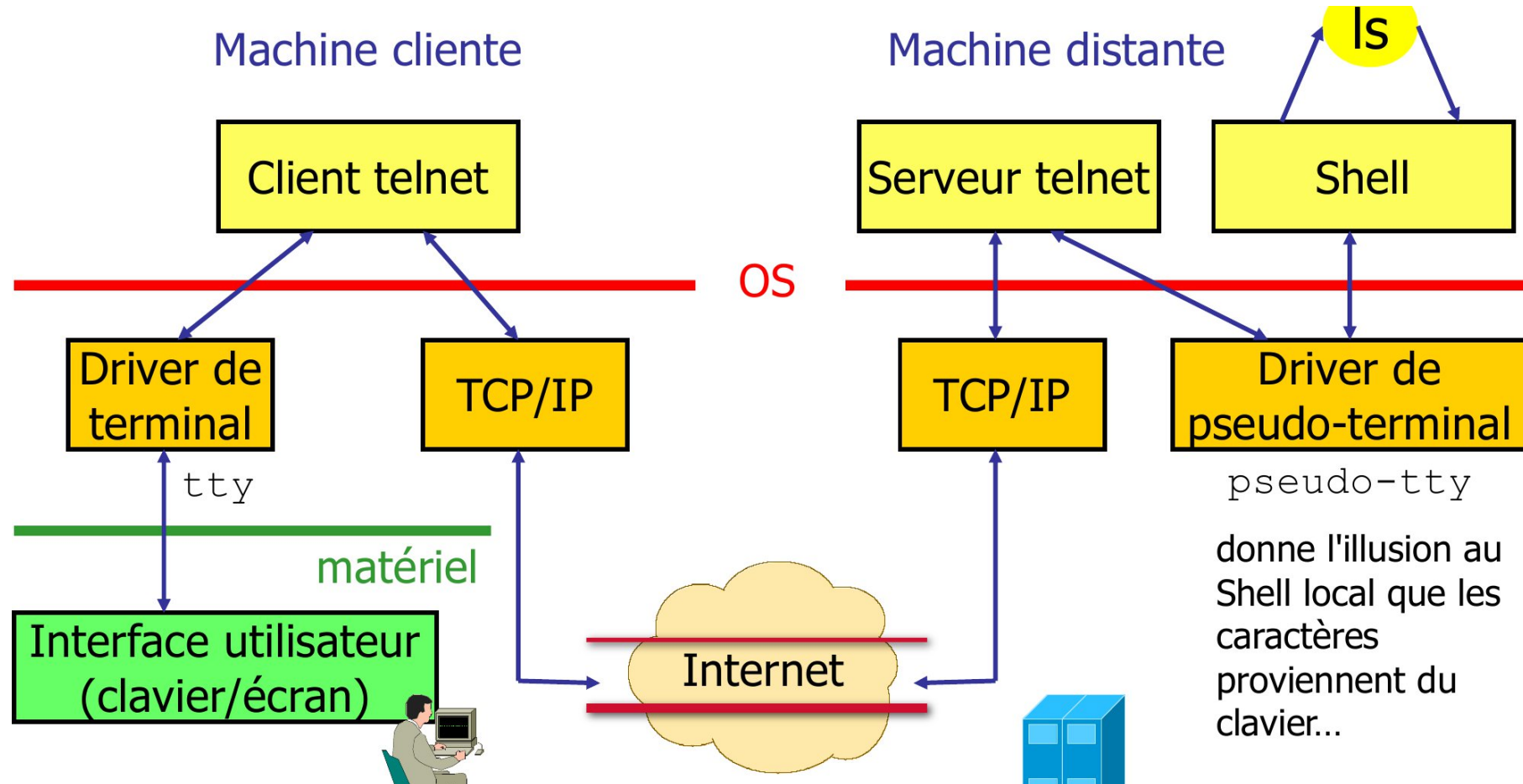
- Introduction / Rappel
- **Applications de connexion à distance (SSH, X)**
- Transfert de fichiers (FTP, TFTP, NFS, SMB)
- Annuaire : NIS, DNS, LDAP
- Messagerie électronique (SMTP, IMAP, MIME)

Applications de connexion à distance

Connexion à distance : principe

- Exécuter un shell sur une machine distante
- Le client capture le clavier et affiche l'écran ; le serveur exécute
- Historiquement : telnet, rlogin, rsh (en clair, non sécurisés)
- Aujourd'hui : SSH, qui chiffre toute la session
 - telnet : le standard (existe sur de nombreuses plate-formes)
 - rlogin : uniquement entre machines unix
 - ssh : sécurisé (authentification + cryptage), peut transporter le DISPLAY

Connexion locale vs connexion distante



Telnet (historique)

- Un des premiers protocoles de l'Internet (RFC 854/855)
- Émulation de terminal sur le réseau (port 23)
- Tout circule en clair, mots de passe compris → à proscrire
- Remplacé par SSH

rlogin et rsh (historique)

- rlogin : login distant « à la BSD » (port 513)
- rsh : exécution d'une commande distante (port 514)
- Authentification par fichiers de confiance (.rhosts) : dangereuse
- En clair et non sécurisés → abandonnés au profit de SSH

SSH : un shell distant sécurisé

- Chiffre toute la session (confidentialité, intégrité)
- Authentification forte du serveur et du client
- Remplace telnet, rlogin et rsh
- Permet aussi le transfert de fichiers (scp, sftp) et des tunnels

SSH : un shell distant sécurisé

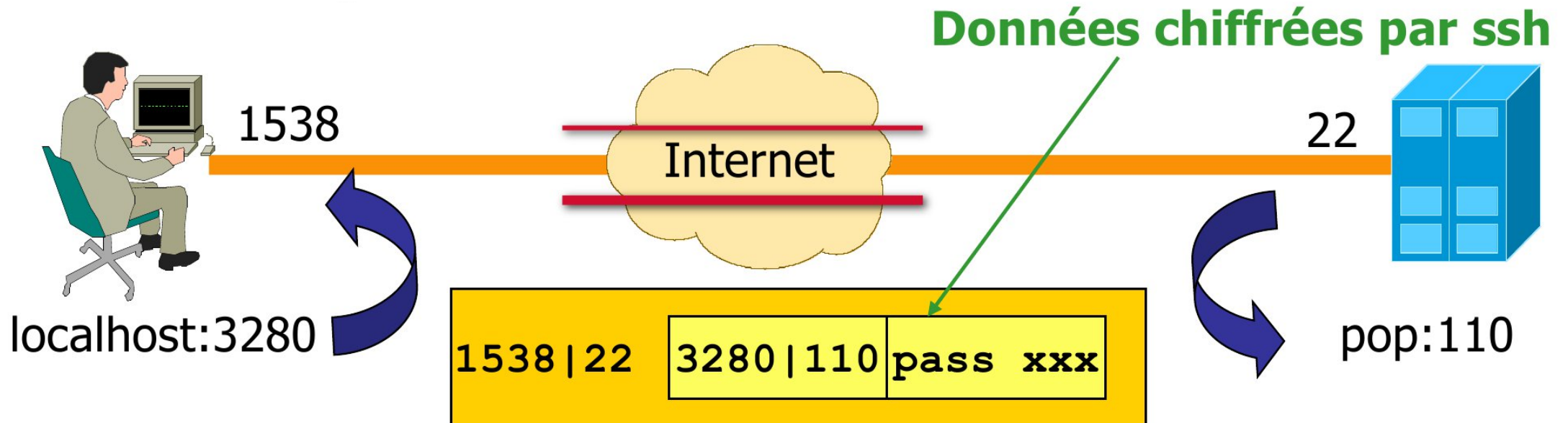
- Les communications sont cryptées
- Authentification à base de clés
- Un des seuls protocoles de connexion à distance qui passe les pare-feux de nos jours
- Permet de transporter des fenêtres graphiques via le tunnel SSH (multiplexage de plusieurs flux dans la connexion)
- Connexion TCP sur le port 22 côté serveur
- Pas encore de RFC (ietf-internet-draft)

SSH : syntaxes et usages

- Connexion : `ssh utilisateur@hôte`
- Commande distante : `ssh utilisateur@hôte commande`
- Copie de fichiers : `scp`, `sftp`
 - `scp file1 file2 user@hostname:`
 - `scp -r dir user@hostname:/tmp`
- Redirection de ports (tunnels) : `ssh -L`

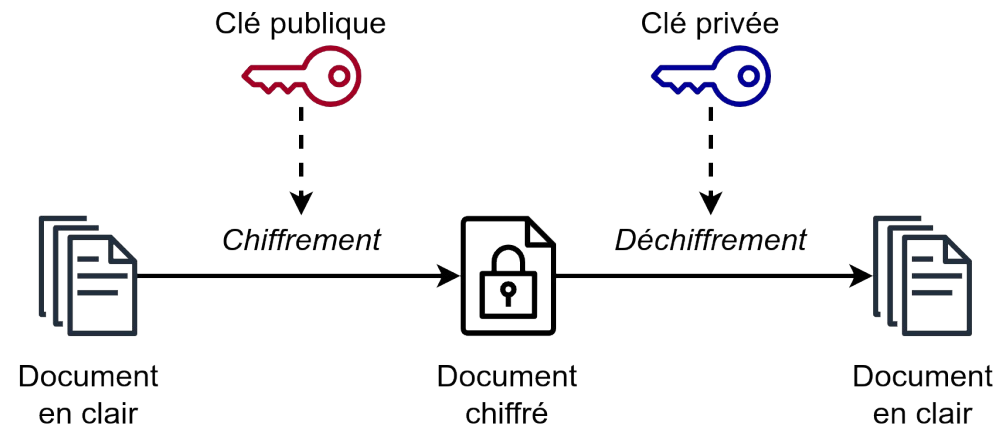
SSH : tunnels (port forwarding)

- On peut faire passer n'importe quoi dans un tunnel ssh :
- `ssh -L 3280:pop.univ-lyon1.fr:110 pop.univ-lyon1.fr`
- Tout ce qu'on envoie sur localhost/3280 arrive sur pop/110 mais les données sont chiffrées (encapsulées dans le protocole ssh)



SSH : le chiffrement - principe

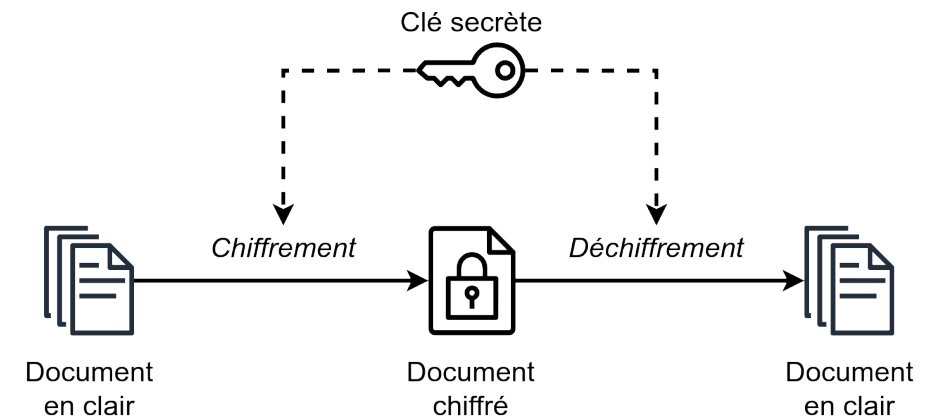
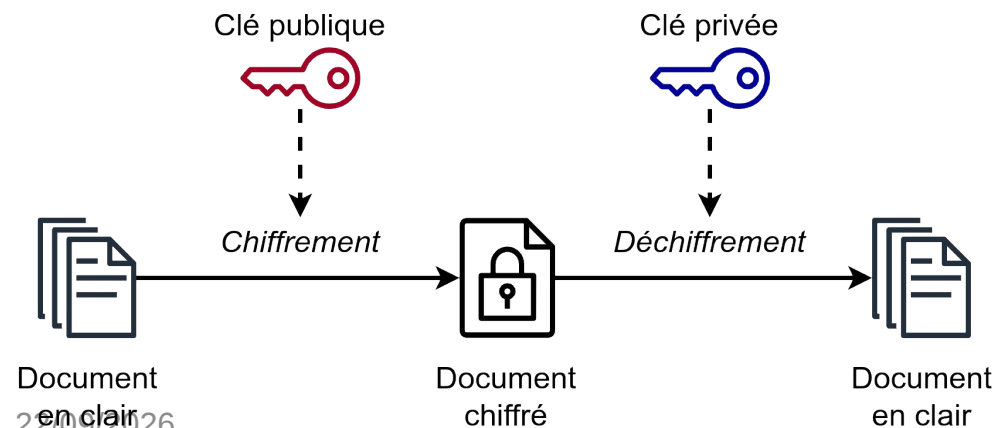
- Principe



- La qualité de la sécurité dépend
 - du secret de la clé
 - de la longueur de la clé (plus il y a de bits, plus il est difficile d'essayer toutes les clés)
 - de la difficulté d'inversion de l'algorithme de chiffrement

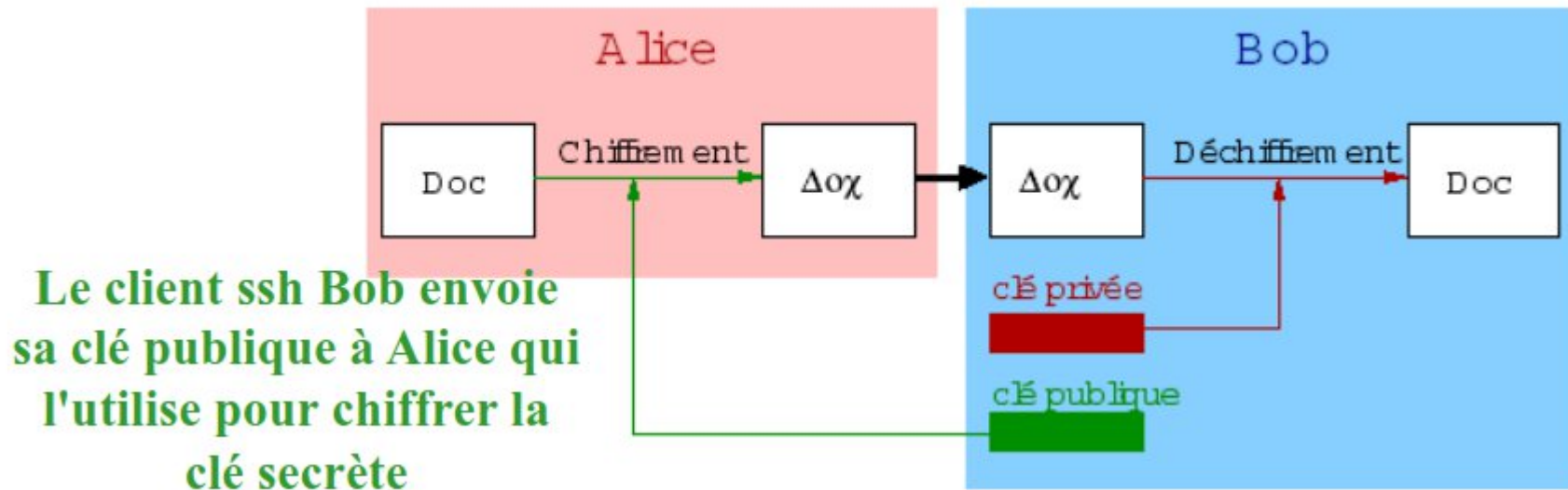
SSH : le chiffrement

- Asymétrique (RSA...) pour l'authentification et l'échange de clé
 - utilisation d'une clé publique pour le chiffrement et d'une clé privée pour le déchiffrement
- Symétrique (AES...) pour chiffrer les données de la session
 - même clé privée secrète partagée utilisée pour le chiffrement et le déchiffrement
- Le symétrique est bien moins coûteux → utilisé pour le flux
- Une clé de session est négociée à l'établissement



Chiffrement : clé de session (asymétrique + symétrique)

Assymétrique : RSA (DSA - El-Gamal)



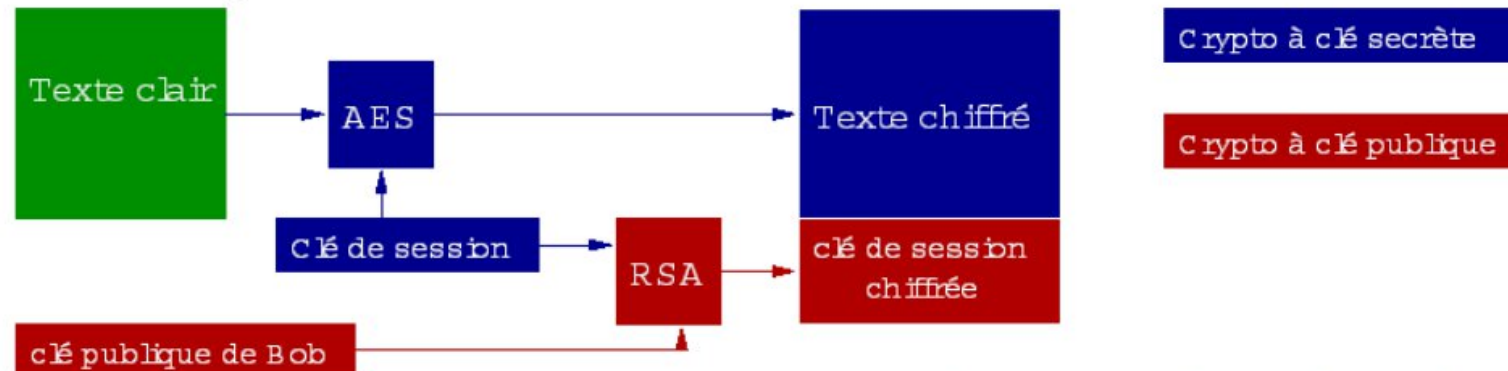
SSH : authentication par clés

- Algorithme asymétrique pour l'authentification (généralement RSA : basé sur l'arithmétique modulo)
- Par mot de passe, ou par bi-clé (clé publique / privée)
- La clé publique est déposée sur le serveur (authorized_keys)
- Plus sûr et pratique (pas de mot de passe en clair)
- Agent SSH pour ne pas ressaisir la phrase de passe

SSH : chiffrement

Fonctions à clé publique très coûteuses → utilisation d'une **clé de session**

Chiffrement par Alice :



ssh de Bob vers Alice, Bob envoie sa clé publique à Alice, Alice fabrique une clé secrète (la clé de session) pour l'envoyer chiffrée à Bob. C'est cette clé secrète qui sera utilisée ensuite pour chiffrer/déchiffrer de manière symétrique avec AES.

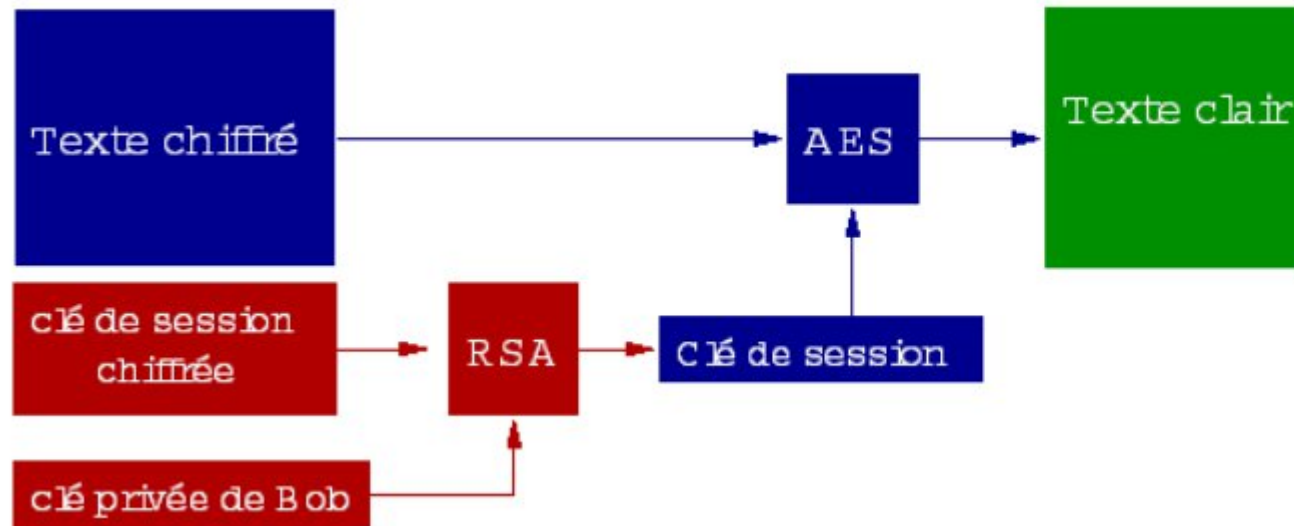
Longueur des clés sûres (2003) :

clé secrète : 128 bits

clé publique/privée : 1024 bits

SSH : déchiffrement

Déchiffrement par Bob :



Crypto à clé secrète

Crypto à clé publique

SSH : authentication

- 4 méthodes sont essayées dans l'ordre par sshd
 - authentication automatique (souvent désactivée sur le serveur car considérée insecured)
 - avec /etc/hosts.equiv ou /etc/ssh/ssh_known_hosts
 - avec ~/.rhosts ou ~/.shosts
 - authentication automatique forte "améliorée"
 - idem (avec fichiers rhosts ou hosts.equiv) mais combinée avec une authentication des hosts par RSA : le serveur vérifie la clé en provenance de la machine cliente
 - Mot de passe demandé lors de la première connexion puis ajout du couple (@IP, clé publique) du serveur dans le fichier du client
 - Si la clé publique change pour cette @IP alors on redemande une authentication forte par mot de passe
 - /etc/ssh/ssh_known_hosts ou ~/.ssh/ssh_known_hosts

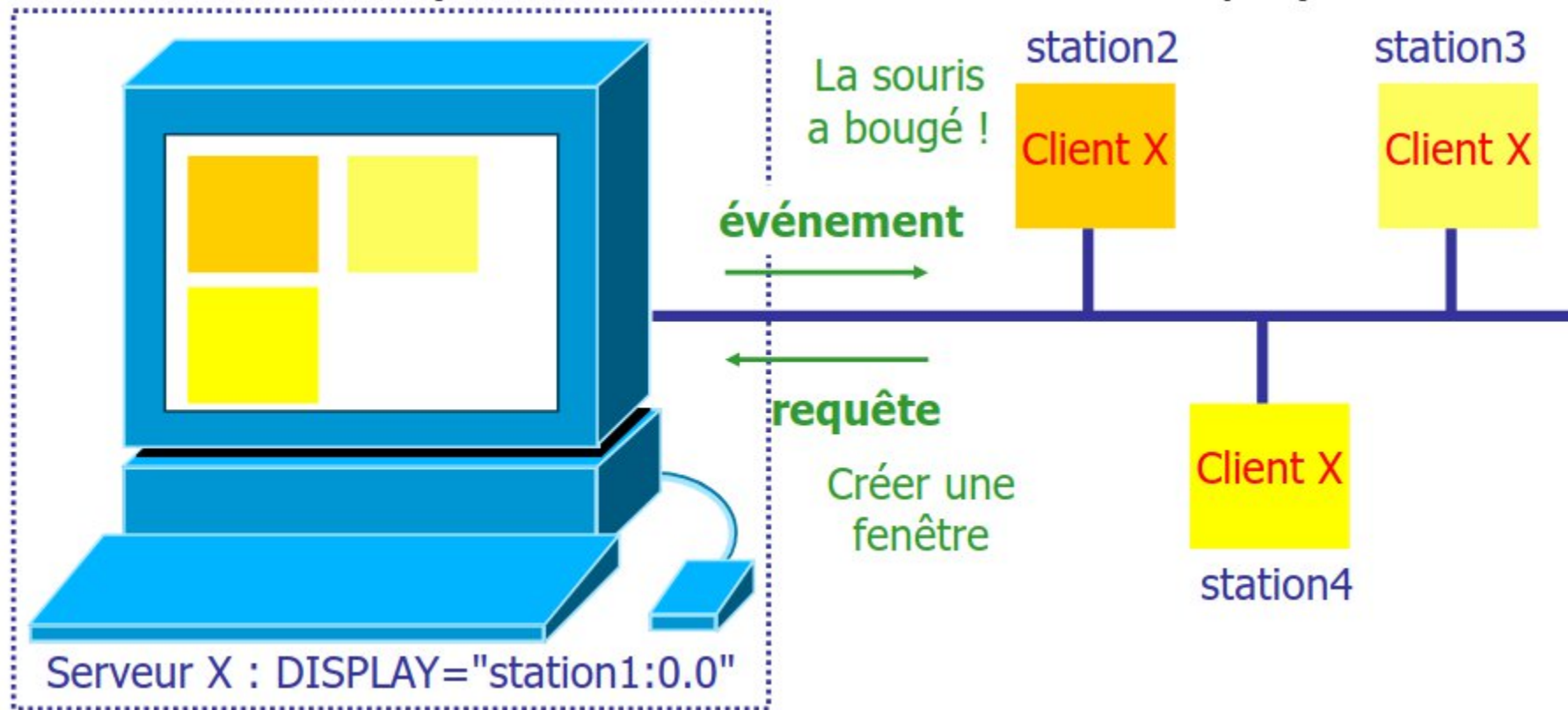
SSH : authentication

- 4 méthodes sont essayées dans l'ordre par sshd
 - authentification automatique forte par RSA (asymétrique)
 - le client génère un couple (clé_pub, clé_pri) avec ssh-keygen et copie la clé publique dans le fichier ~/.ssh/authorized_keys sur le serveur ; liste les clés publiques autorisées (équivalent du ~/.rhosts !)
 - le client envoie au serveur sa clé publique ; si elle est dans le fichier, le serveur génère un nb aléatoire de 256 bits qu'il chiffre avec la clé envoyée par le client ; le client déchiffre avec sa clé privée puis la renvoie avec hachage MD5 ; le serveur calcule le hachage MD5 et vérifie
 - authentification par saisie du mot de passe mais ce dernier est chiffré par RSA avant d'être envoyé au serveur

X Window : multi-fenêtrage réparti

- Système d'affichage graphique réparti (Unix)
- Un serveur X : gère le matériel (clavier, écran, ...) et leur utilisation par les applications graphiques ; accessible sur le port TCP 6000+n où n est le numéro de DISPLAY
- Des clients X : applications graphiques qui nécessitent un serveur X (xemacs, xterm, xcalc, xv, ...)
- Le protocole X : fait communiquer les clients et le serveur

Architecture de X (serveur X et clients distants)



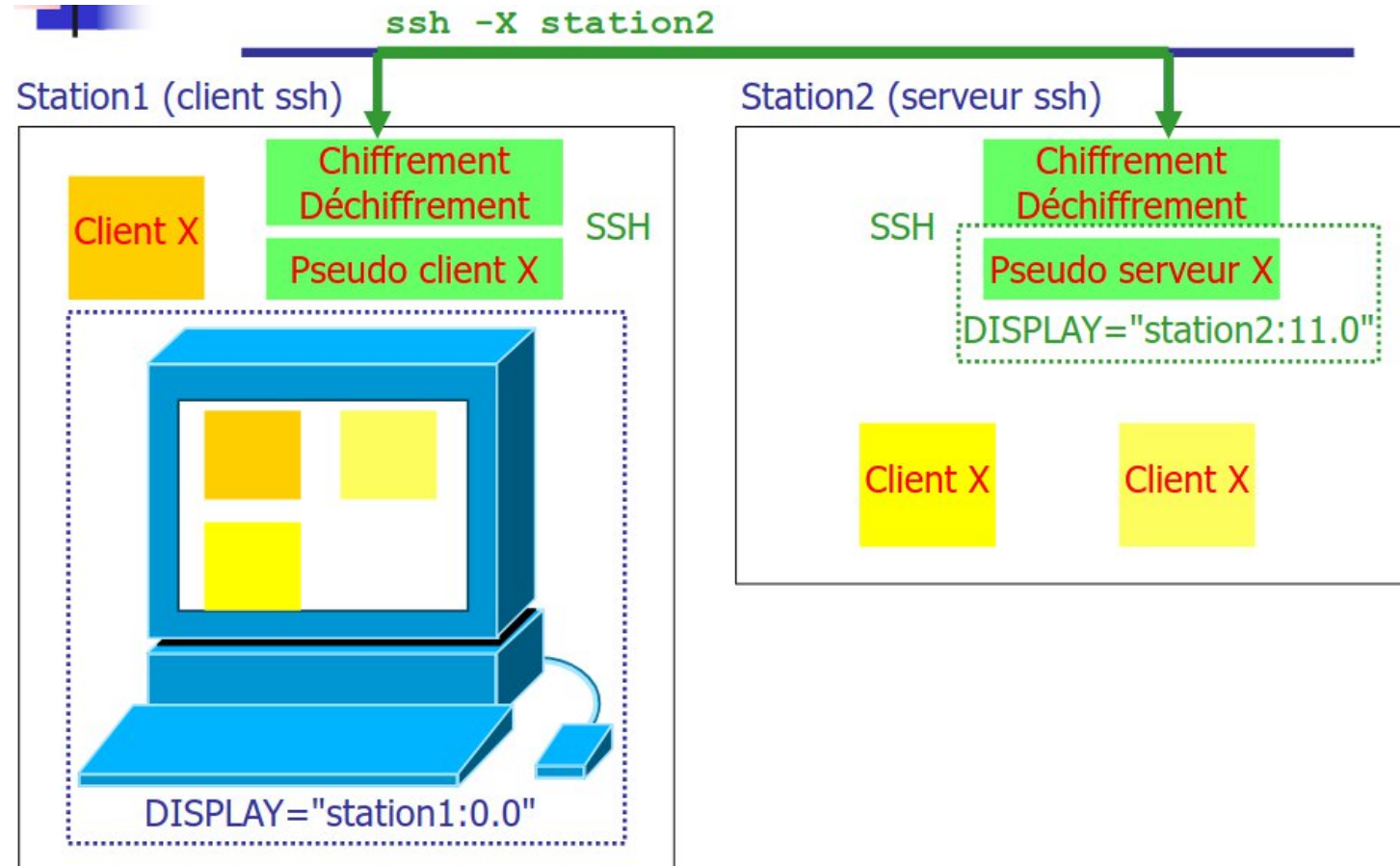
X Window : multi-fenêtrage réparti

- Système d'affichage graphique réparti (Unix)
- Chaque client X peut définir ses caractéristiques
 - spécifications standards
 - fontes, géométrie de la fenêtre, background, foreground, borderwidth, couleurs...
 - spécifications particulières à l'application
 - affichage ou non d'un ascenseur...
- Gestion de fenêtres : Window Manager
 - un client X particulier qui gère
 - déplacement/redimensionnement de fenêtre
 - créer/détruire/iconifier des fenêtres
 - lancer ou terminer des applications X

SSH : X11 forwarding

- SSH encapsule le protocole X dans un tunnel chiffré (ssh -X)
- Positionne la variable DISPLAY automatiquement
 - si la variable \$DISPLAY du client ssh est positionnée, ssh -X permet au serveur d'exporter les fenêtres graphiques lancées à partir de la connexion ssh vers le \$DISPLAY du client (un "proxy X server" est créé sur la machine serveur pour transférer les connexions X vers le client via la session ssh)
- Affiche localement une application graphique exécutée à distance
- Sécurise X, qui n'est pas chiffré nativement

SSH X11 forwarding



Plan de la partie 2

- Introduction / Rappel
- Applications de connexion à distance (SSH, X)
- **Transfert de fichiers (FTP, TFTP, NFS, SMB)**
- Annuaire : NIS, DNS, LDAP
- Messagerie électronique (SMTP, IMAP, MIME)

Applications de transfert de fichiers

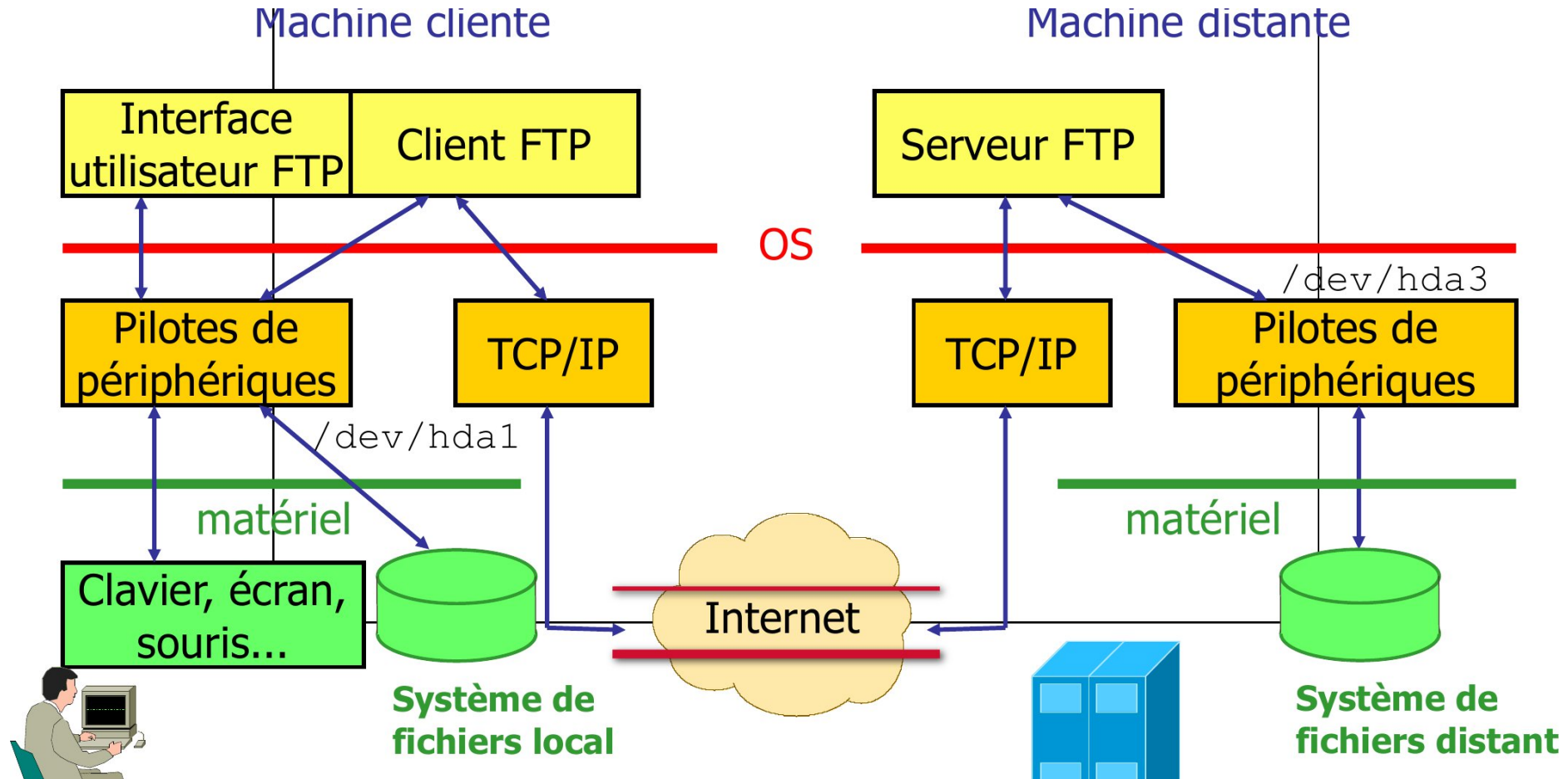
Protocoles de transfert de fichiers

- Copier un fichier d'un système de fichiers vers un autre, via le réseau
- L'hétérogénéité concernant les fichiers est dépendante d'un système à l'autre
 - de la façon de représenter les noms de fichier (longueur, caractère espace,...)
 - des droits d'accès au fichier (lecture, écriture, exécution, propriétaire, ...)
 - de la représentation des données contenues dans le fichier (saut de ligne...)
 - --> mode ascii : transfert au format NVT avec conversion au format local (TYPE A)
 - --> mode binary : transfert sans conversion (TYPE I)
- Gérer l'hétérogénéité (noms, formats, droits) des deux systèmes

Protocoles de transfert de fichiers

- Mode Client/Serveur
 - Le client (initiateur de la connexion) interagit avec l'utilisateur, le système de fichiers local et les protocoles réseau
 - Le serveur (héberge les fichiers distants) interagit avec les protocoles réseau et le système de fichiers distant
- Deux familles : transfert explicite (FTP) et accès transparent (NFS, SMB)

Transfert de fichiers vers/depuis un hôte distant

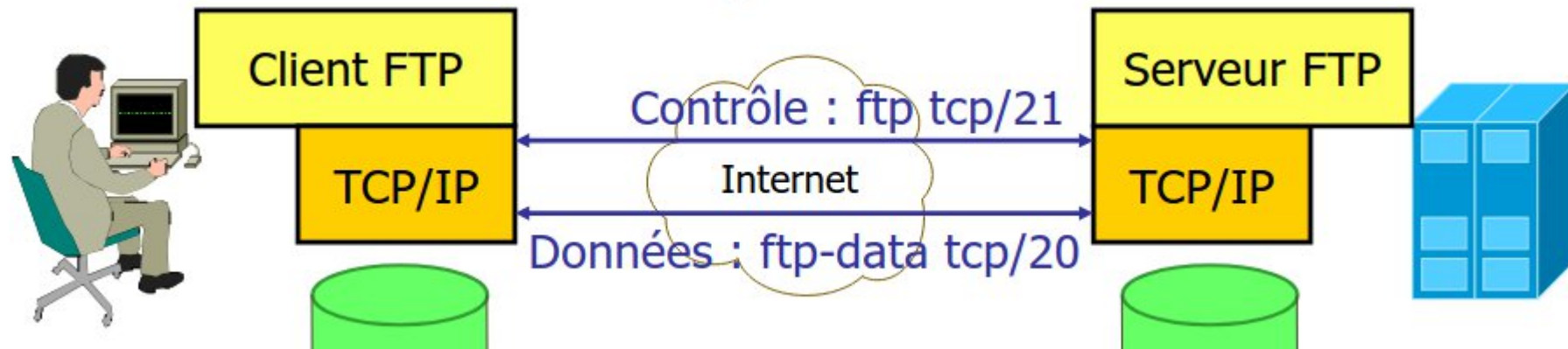


FTP : File Transfer Protocol (RFC 959)

- Standard historique de transfert de fichiers (port 21)
- Authentification par login / mot de passe - en clair !
- Deux connexions séparées :
 - Port 21 pour le contrôle
 - Port 20 pour les données
- Aujourd'hui : préférer SFTP / FTPS (chiffrés)

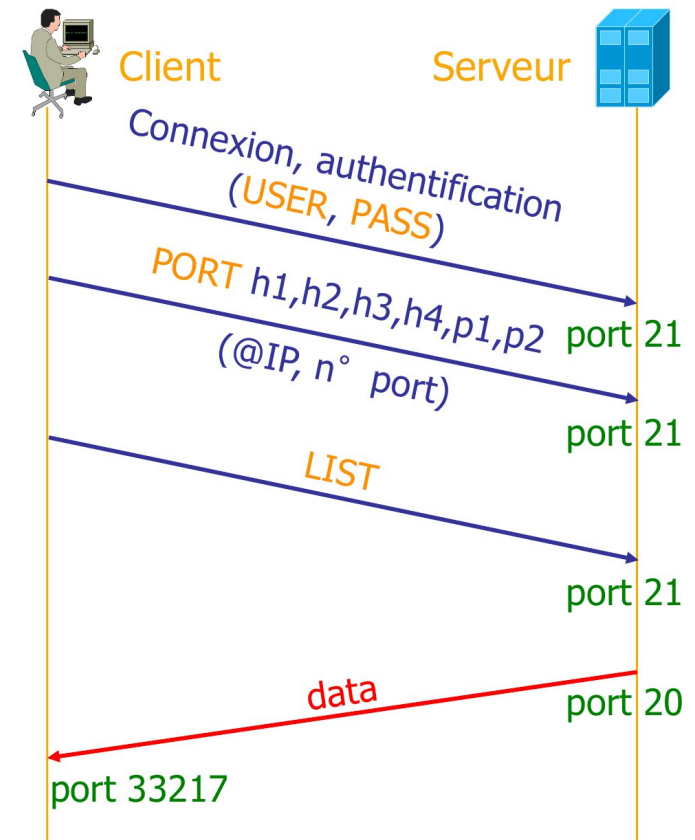
FTP : deux connexions (contrôle et données)

- Connexion de contrôle (port 21) : dialogue commandes/réponses, permanente
- Connexion de données (port 20) : transfert effectif, temporaire
- Mode actif : le serveur ouvre la connexion de données vers le client
- Mode passif : le client l'ouvre (compatible pare-feu / NAT)



FTP : connexions contrôle et données

- Scénario d'une connexion
 - le client FTP se connecte sur le port 21 du serveur
 - le port 21 sert à envoyer des commandes au serveur FTP (put, get, cd, ...)
 - si une commande nécessite que des données soient reçues ou transmises (ls, get, put, ...), le client envoie une commande PORT au serveur indiquant un port ($p1 \cdot 256 + p2$) sur lequel le serveur va créer une connexion ftp-data depuis son port 20
 - la connexion ftp-data est close dès que le transfert est terminé



FTP : connexions contrôle et données

- Connexion contrôle (ftp) :
 - échange des requêtes/réponses (dialogue client/serveur)
 - permanente, full-duplex, besoin de fiabilité (et faible délai !)
 - initiée par le client
- Connexion données (ftp-data) :
 - envoi de fichier ou liste de fichiers/répertoires (données)
 - temporaire, full-duplex, besoin de débit (et fiabilité !)
 - initiée par défaut par le serveur
 - ouverture active (connect()) du serveur vers le client (depuis le port 20 vers le port ??)
 - la connexion est fermée dès que le caractère EOF est lu
- TCP ou UDP ?

FTP : connexions contrôle et données - Active transfer & Passive transfer

- Active transfer : la connexion ftp-data est initiée par le serveur
 - --> problème de firewall ou de NAT : impossibilité de créer la connexion à partir du serveur même s'il connaît le numéro de port du client
- Passive transfer : ftp-data initiée par le client
 - intégré dans les navigateurs, paramétrable sur certains clients
 - fonctionnement : le client envoie la commande PASV au lieu de PORT sur le port 21 (RFC 1579 : Firewall-Friendly FTP) ce qui revient à demander au serveur de faire un listen() sur le port 20

FTP : requêtes et réponses

- Requêtes : USER, PASS, RETR, STOR, LIST, CWD, TYPE...
- Réponses : un code à 3 chiffres + un message, lisibles en ASCII
- 1yz...5yz : information, succès, intermédiaire, erreur temporaire, définitive
- Le dialogue transite sur la connexion de contrôle

Le code est un nombre sur trois chiffres signifiant :

Status	Signification	Status	Signification
		x0z	Erreur de syntaxe
1yz	Réponse positive préliminaire (une autre réponse suivra)	x1z	Réponse Informative (HELP...)
2yz	Réponse positive finale (une autre requête est possible)	x2z	Relatif à une connexion
3yz	Réponse positive intermédiaire (une autre requête doit suivre)	x3z	Relatif à l'authentification
4yz	Réponse négative temporaire (la même requête peut réussir plus tard)		
5yz	Réponse négative définitive (la requête n'est pas acceptée)	x5z	Relatif au système de fichier

FTP : les codes de réponse

Le code est un nombre sur trois chiffres signifiant :

Status	Signification	Status	Signification
		x0z	Erreur de syntaxe
1yz	Réponse positive préliminaire (une autre réponse suivra)	x1z	Réponse Informative (HELP...)
2yz	Réponse positive finale (une autre requête est possible)	x2z	Relatif à une connexion
3yz	Réponse positive intermédiaire (une autre requête doit suivre)	x3z	Relatif à l'authentification
4yz	Réponse négative temporaire (la même requête peut réussir plus tard)		
5yz	Réponse négative définitive (la requête n'est pas acceptée)	x5z	Relatif au système de fichier

71

SFTP : le transfert de fichiers sécurisé

- SFTP = SSH File Transfer Protocol (à ne pas confondre avec FTPS)
- Fonctionne au-dessus de SSH (port 22) : une seule connexion, chiffrée
- Authentification SSH (mot de passe ou clés) ; confidentialité et intégrité
- Remplace FTP (en clair) et traverse facilement pare-feu / NAT

SFTP en pratique et SFTP vs FTPS

- Client interactif sftp : get, put, ls, cd, mkdir...
- Copie simple : scp ; synchronisation : rsync au-dessus de SSH
- SFTP ≠ FTPS :
 - FTPS = FTP + TLS (deux connexions)
 - SFTP = sous-système de SSH (une connexion)

TFTP : Trivial File Transfer Protocol (RFC 1350)

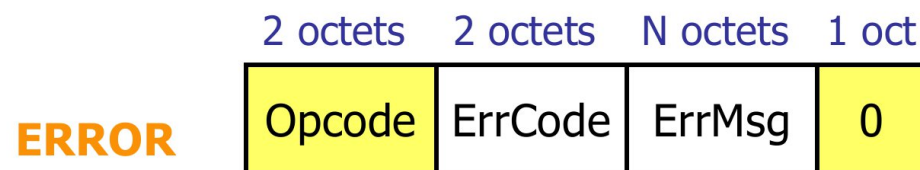
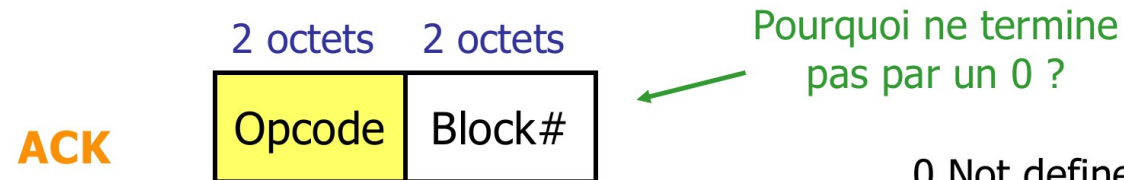
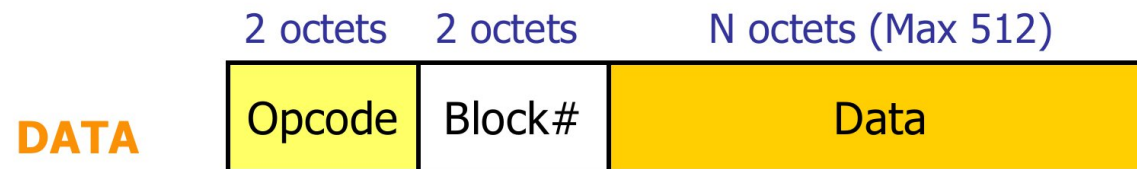
- Version minimale, au-dessus d'UDP (port 69)
- Pas d'authentification ni de contrôle d'accès
 - Comme il n'y a pas d'authentification, les accès sur le serveur sont limités aux répertoires passés en arguments du démon tftpd (/tftpboot par défaut)
- Fiabilité gérée par l'application (acquittements, ré-émission)
- Usage typique : amorçage réseau (boot PXE), équipements

TFTP : les types de messages

- Les 5 types de messages

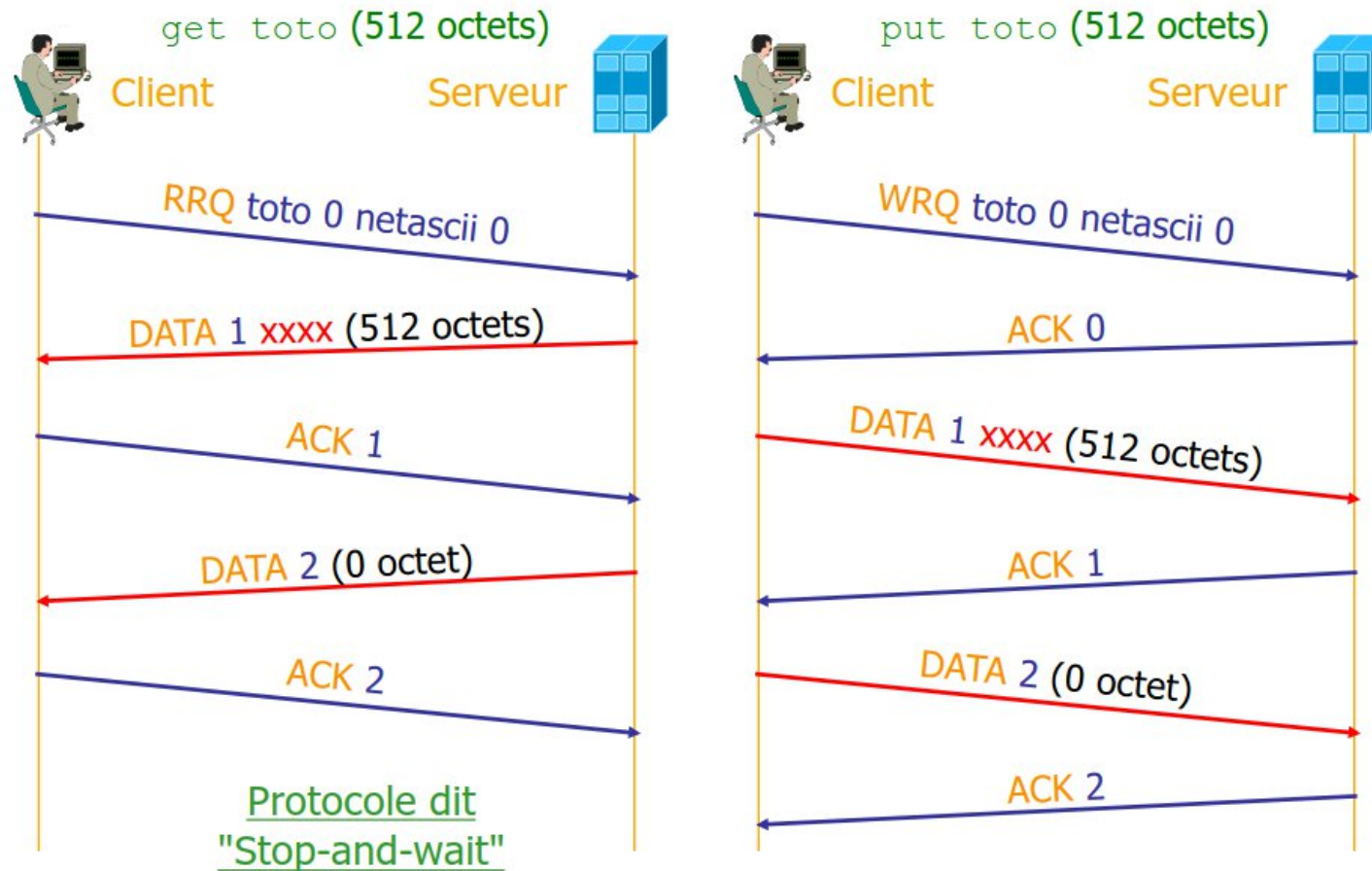
Opcode	Opération	Description
1	RRQ	Read request
2	WRQ	Write request
3	DATA	Data
4	ACK	Acknowledgment
5	ERROR	Error (sert aussi d'ACK)

TFTP : format des messages



- 0 Not defined.
- 1 File not found.
- 2 Access violation.
- 3 Disk full or allocation exceeded.
- 4 Illegal TFTP operation.
- 5 Unknown transfer ID.
- 6 File already exists.

TFTP : exemples d'échanges (stop-and-wait)



Accès transparent aux fichiers (NFS, SMB)

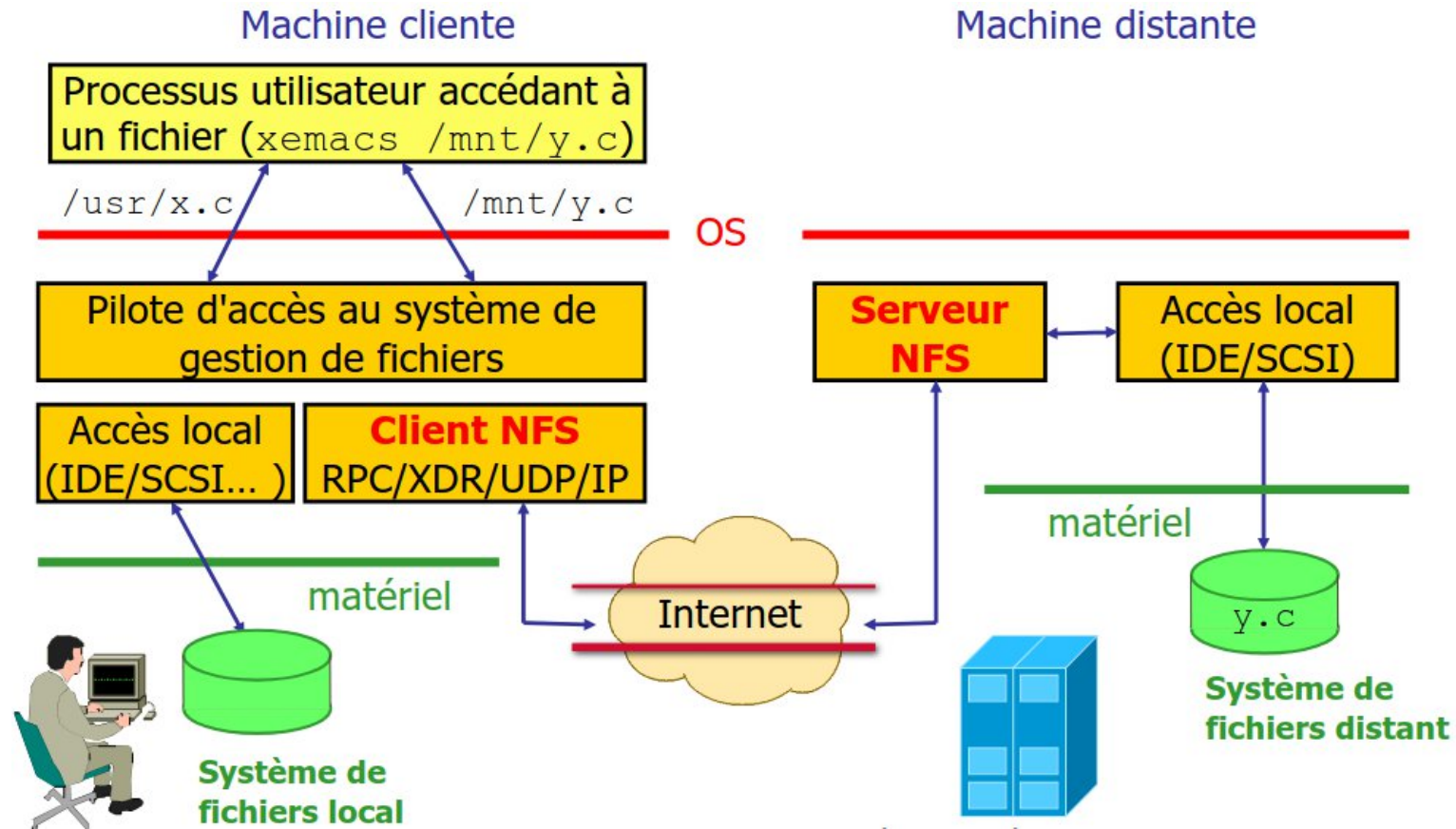
Accès transparent aux fichiers distants

- Le système de fichiers distant est monté localement (transparent)
 - l'accès aux fichiers distants est complètement transparent pour l'utilisateur
 - tout se passe comme si le système de fichiers distant était local
 - l'utilisateur peut éditer le fichier, le modifier, ... ; les modifications seront répercutées sur le système de fichiers distant
- L'utilisateur manipule les fichiers comme en local
- NFS - Network File System (monde Unix)
- SMB - Server Message Block / CIFS - Common Internet File System (monde Windows)
- S'appuient sur des appels de procédures distantes

NFS : Network File System

- Proposé par Sun (1985) ; RFC 1094 et suivantes
- Fondé sur les RPC Sun et la représentation XDR
- Le serveur exporte des répertoires, le client les monte
- Indépendant du système et du transport
 - a priori, les client et serveur NFS devraient être des processus utilisateur s'exécutant au-dessus de UDP/IP
 - en fait, le client et serveur NFS s'exécutent dans le noyau (sockets)
 - le client pour rendre transparent l'accès à un fichier via NFS (pas de différence d'utilisation d'un fichier local et d'un fichier distant)
 - le serveur pour des raisons d'efficacité

NFS : principe de fonctionnement



NFS : les éléments d'accès aux fichiers (VFS)

```
root@192.168.69.1# tail -1 /etc/fstab
192.168.69.2:/home /nfshome nfs defaults,noauto 0 0
root@192.168.69.1# mkdir /nfshome
root@192.168.69.1# mount /nfshome
root@192.168.69.1# ls -l /nfshome
drwxr-xr-x  2 1003   5000      1024 fév 16 13:32 olivier
root@192.168.69.1# df
Filesystem            1k-blocks    Used   Available   Use% Mounted on
/dev/hda2              3898108  2766592    930304    75%  /
192.168.69.2:/home    16128636  1493328   13815996   10%  /nfshome
```

Client NFS



`open(), read(), write()`

```
root@192.168.69.2# cat /etc/exports
# /etc/exports: the access control list for filesystems which
# may be exported to NFS clients.  See exports(5).
/home 192.168.69.0/255.255.255.0(rw,root_squash,async)
```



Serveur NFS

NFS : sécurité et évolutions

- Principe d'authentification :
 - (uid, gid)local est mappé en (uid, gid)distant -> équivalence entre les droits locaux et distants
 - problème pour root :
 - quels droits possède le root d'une machine cliente sur les fichiers exportés par un serveur NFS ?
 - par défaut, root est mappé en nobody pour des raisons de sécurité (sinon il faut mettre no_root_squash dans /etc/exports)
- Règle de non transitivité de NFS :
 - si A exporte /home à B ; si B monte A:/home dans /home2 et exporte /home2 à C alors C n'aura pas accès au /home de A
 - sinon il n'y aurait aucun contrôle possible des règles d'exportation donc pas de sécurité... (B pourrait exporter à C alors que A ne veut pas autoriser C)

NFS : sécurité et évolutions

- NFSv2 / v3 : authentification faible (AUTH_SYS) → à isoler
- NFSv4 : sécurité renforcée (Kerberos), au-dessus de TCP
- Chiffrement et contrôle d'accès à prévoir
- Toujours très utilisé en environnement Unix / Linux

NFS : un serveur sans état

- Le serveur NFS est sans état : entre deux requêtes, il ne conserve aucune information sur
 - les accès précédents à un fichier donné,
 - les fichiers ouverts, ...
 - le LOOKUP correspond au open() mais il n'y a pas de procédure correspondant au close()
 - après un LOOKUP, le serveur ne sait pas si le client va effectivement "utiliser" le fichier ou non
- Pourquoi sans état ?
 - en cas de crash du serveur NFS
 - permet de simplifier son redémarrage
 - transparent pour le client : il n'a pas besoin de réitérer certaines requêtes en cas de crash

NFS : au dessus de TCP

- En principe, pour plus de réactivité, NFS utilise les RPC sur UDP (dans le cadre d'un LAN)
- Mais pour étendre NFS aux WAN, les dernières versions permettent d'exécuter NFS sur RPC/TCP
- Caractéristiques de NFS sur TCP
 - le serveur fait une ouverture passive sur le port TCP/2049
 - quand un client monte une partition NFS, cela se traduit par une ouverture active --> une connexion TCP par point de montage (soit une par file system)
 - toutes les applications qui utilisent ce système de fichiers partagent la même connexion TCP
 - plus résistant aux pannes du serveur NFS : le client essaie régulièrement de rétablir la connexion et conservent les requêtes RPC en attente

SMB : Server Message Block

- Partage de fichiers et d'imprimantes (monde Windows)
- Historiquement sur NetBIOS, aujourd'hui sur TCP (port 445)
- Authentification et contrôle d'accès par ressource
- Deux niveaux de protection : par partage et par ressource

SMB : Server Message Block

- Chaque machine (client ou serveur) possède un nom sur 15 caractères
- SMB ajoute un 16ième caractère pour distinguer
 - les serveurs de fichiers et d'imprimantes, les clients, ...
- Notion de domaine
 - un ensemble d'utilisateurs (avec nom et mot de passe) et de serveurs (avec des droits d'accès)
- un primary domain server contient la base de données des utilisateurs et de leurs mots de passe
- Un serveur partage une ou plusieurs ressources
 - fichiers, imprimantes, ...
 - à chaque triplet (domaine, serveur, ressource) correspond
 - un nom unique \\server\resource_name

SMB : Server Message Block

- Deux niveaux de protection des accès :
 - au niveau de chaque utilisateur : basé sur le nom des utilisateurs (user, passwd), permet de gérer l'accès aux ressources voire aux éléments d'une ressource
 - au niveau de chaque ressource : un mot de passe commun à tous les utilisateurs est associé à une ressource pour y autoriser l'accès
- CIFS : Common Internet File System
 - dernière version de SMB proposant un meilleur passage à l'échelle (extensibilité)
 - divulgation du protocole SMB par Microsoft à l'IETF en 1996 sous ce nom --> a permis l'apparition de Samba

SMB et Samba

- Samba : implémentation libre de SMB sous Unix / Linux
- Partage entre machines Unix et Windows (dans les deux sens)
- Partage d'imprimantes
- Peut jouer le rôle de serveur dans un domaine

Plan de la partie 2

- Introduction / Rappel
- Applications de connexion à distance (SSH, X)
- Transfert de fichiers (FTP, TFTP, NFS, SMB)
- **Annuaire : NIS, DNS, LDAP**
- Messagerie électronique (SMTP, IMAP, MIME)

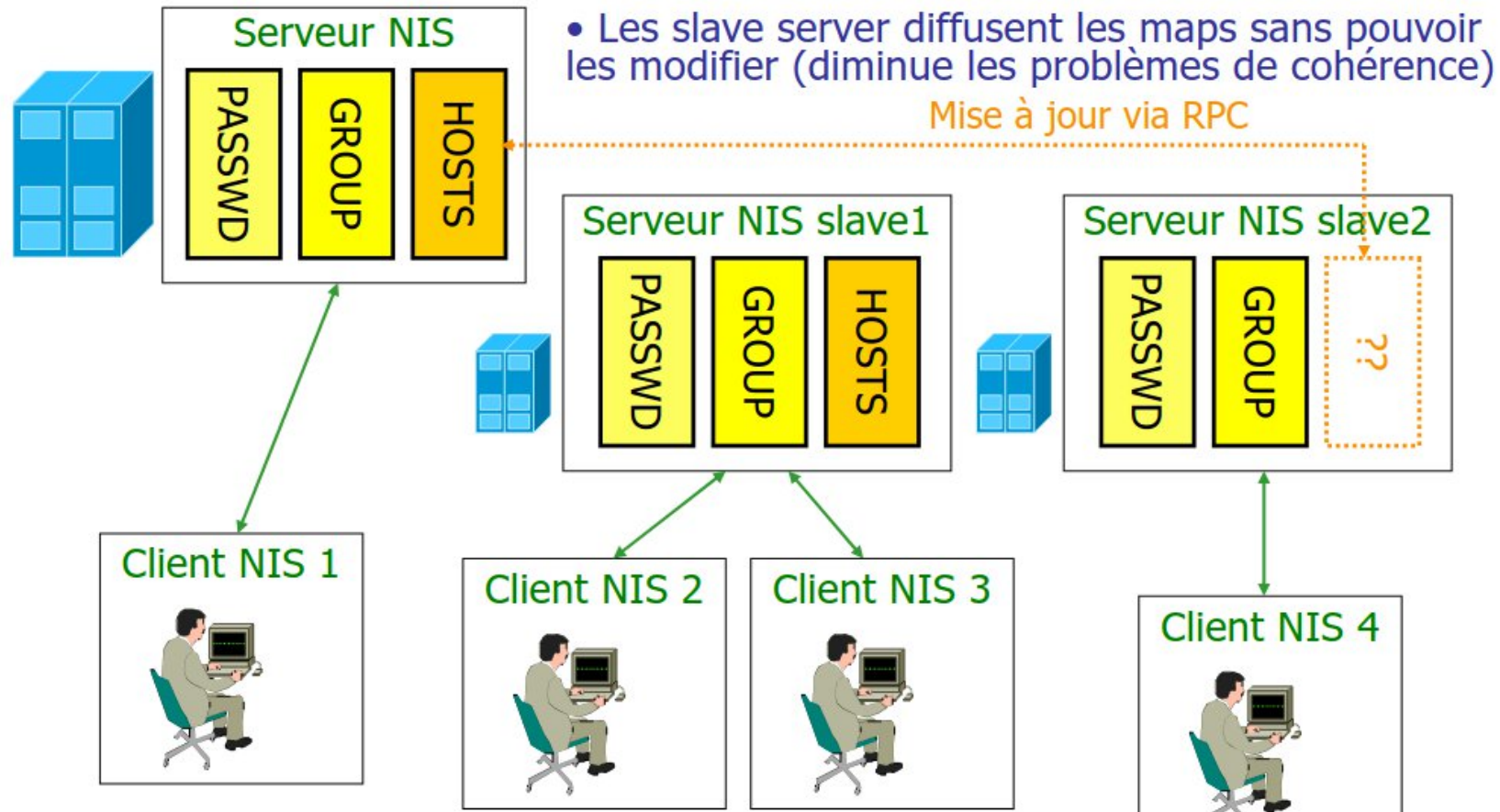
NIS : un annuaire répliqué

- Network Information Service (ex-Yellow Pages), proposé par Sun
- Centralise des informations communes (comptes, groupes, hôtes)
- Bases partagées entre les machines d'un parc (maps)
- Simplifie l'administration : un seul point de gestion

NIS : architecture

- Un serveur maître (master) détient les maps de référence
- Des serveurs esclaves (slaves) en réplique (redondance)
- Les clients interrogent un serveur pour comptes / hôtes
- Mise à jour propagée du maître vers les esclaves (via RPC)

NIS : architecture (maître, esclaves, clients)



NIS : limites et évolutions

- Sécurité faible (données peu protégées, confiance réseau)
 - Pas d'authentification des clients NIS : il suffit de connaître le nom de domaine pour interroger le serveur et connaître le contenu de ses maps
- Adapté à un réseau local simple
- Remplacé aujourd'hui par LDAP (souvent avec Kerberos)
- LDAP : annuaire standard, sécurisé et fédérateur

DNS : Domain Name System

DNS : à quoi ça sert

- Traduire les noms de machines en adresses IP (et l'inverse)
- Annuaire distribué à l'échelle de l'Internet
- Indispensable : on manipule des noms, pas des adresses
- Autres usages : routage du courrier (MX), alias, répartition

Protocole applicatif

- Base de données distribuée
- Les hôtes, routeurs, serveurs de noms communiquent pour effectuer la traduction
- DNS est utilisé par d'autres protocoles applicatifs mais n'est pas utilisé directement par l'application comme SMTP...
- Modèle Client/Serveur : un émetteur interroge un serveur de noms (serveur DNS)
- port 53/UDP (ou 53/TCP pour les mises à jour)

DNS : les services fournis

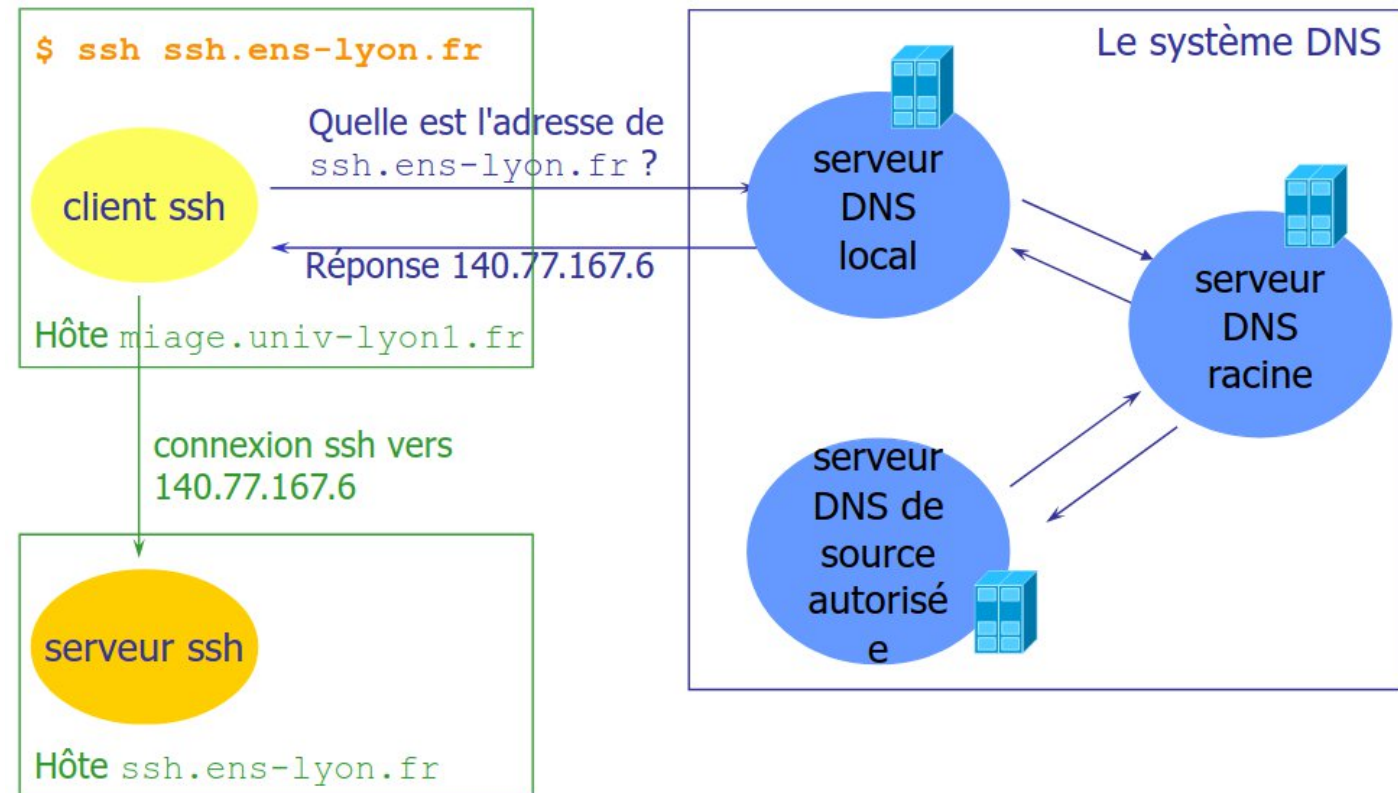
- Le service principal : la traduction d'adresses
- Autres services :
 - permettre le " Host aliasing" : donner un pseudonyme à une machine qui a un nom peu parlant
 - permettre le " Mail server aliasing" : un serveur Web et un serveur Mail peuvent avoir le même pseudonyme même s'ils n'ont pas la même adresse IP (2 machines différentes)
 - permettre la répartition de la charge : un nom de serveur Web ou Mail peut correspondre à plusieurs adresses IP (serveurs Web ou Mail répliqués) avec un système de rotation dans les réponses du serveur DNS
- Pour l'utilisateur, le DNS n'est qu'une boîte noire mais en réalité très compliquée
 - une requête DNS peut impliquer plusieurs serveurs de noms répartis dans le monde entier

Un système centralisé ? Non

- Impossible à l'échelle de l'Internet (taille, charge, pannes)
- Un seul serveur = point de défaillance unique et goulet
- D'où un système distribué et hiérarchique
- Réparti sur des millions de serveurs
 - Aucun serveur ne connaît toutes les correspondances nom <--> adresse IP
 - Si un serveur ne connaît pas une correspondance, il interroge un autre serveur jusqu'à atteindre le serveur détenant l'information désirée

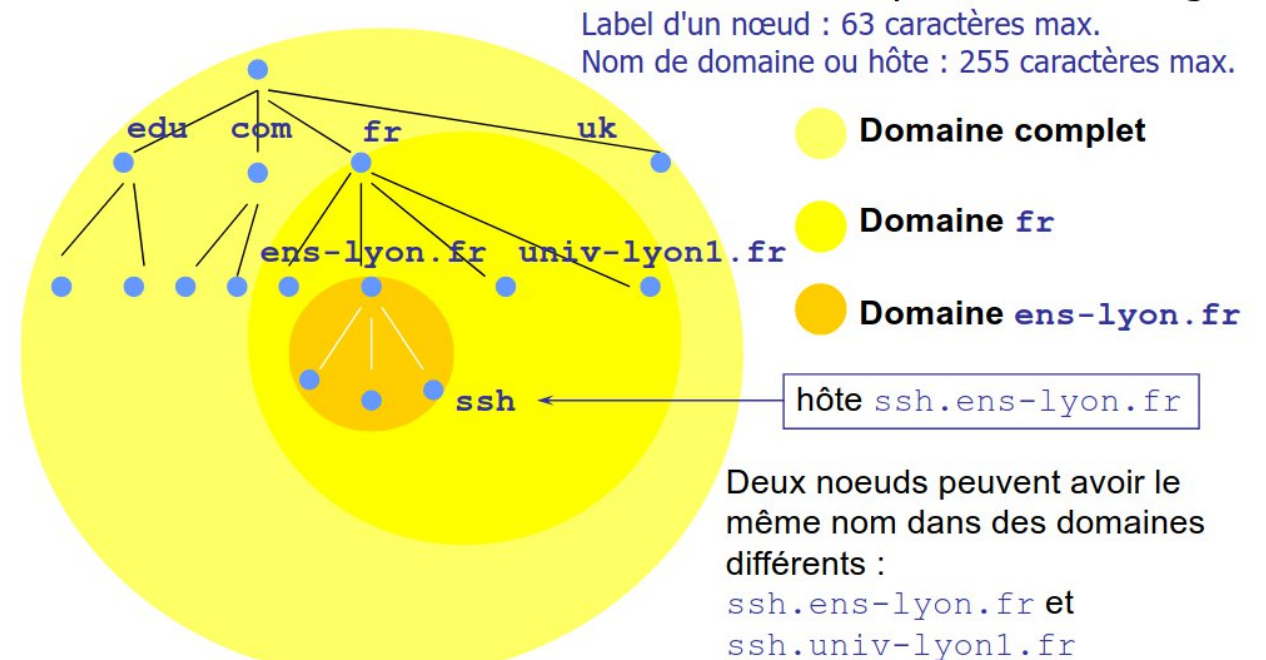
DNS : trois types de serveur

- Les serveurs de noms locaux (à qui s'adressent les requêtes locales ; en charge de la résolution)
- Les serveurs de noms racine (sont censés savoir comment s'approcher de la réponse)
- Les serveurs de noms de source autorisée (contiennent les correspondances "officielles")



La notion de domaine DNS

- L'espace de noms est un arbre (racine → TLD → sous-domaines)
- Un domaine = un sous-arbre entier de l'espace de noms
- Label ≤ 63 caractères ; nom complet ≤ 255
- Deux nœuds peuvent porter le même nom dans des domaines différents

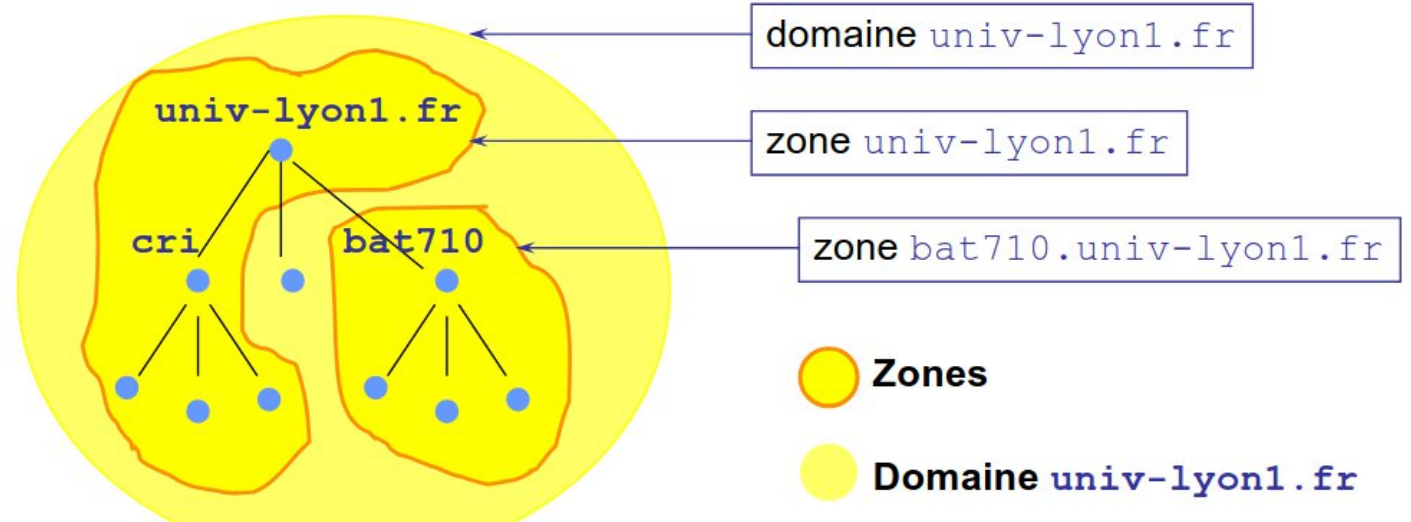


La notion de domaine DNS

- Le premier niveau de l'arbre
 - Top Level Domain (TLD)
 - géré par l'ICANN (Internet Corporation for Assigned Names and Numbers)
 - on distingue deux catégories de TLD
 - les " Generic TLD" : .com, .org, .gov, .gouv, .net, ...
 - les "Countries TLD" : .fr, .uk, .us, .jp, ... (240 en tout)
- La gestion des autres niveaux est laissée aux entités correspondantes (AFNIC pour .fr)
 - zone DNS : un sous-arbre de l'arbre administré séparément par un organisme qui gère la délégation des noms et sous-domaines de la zone

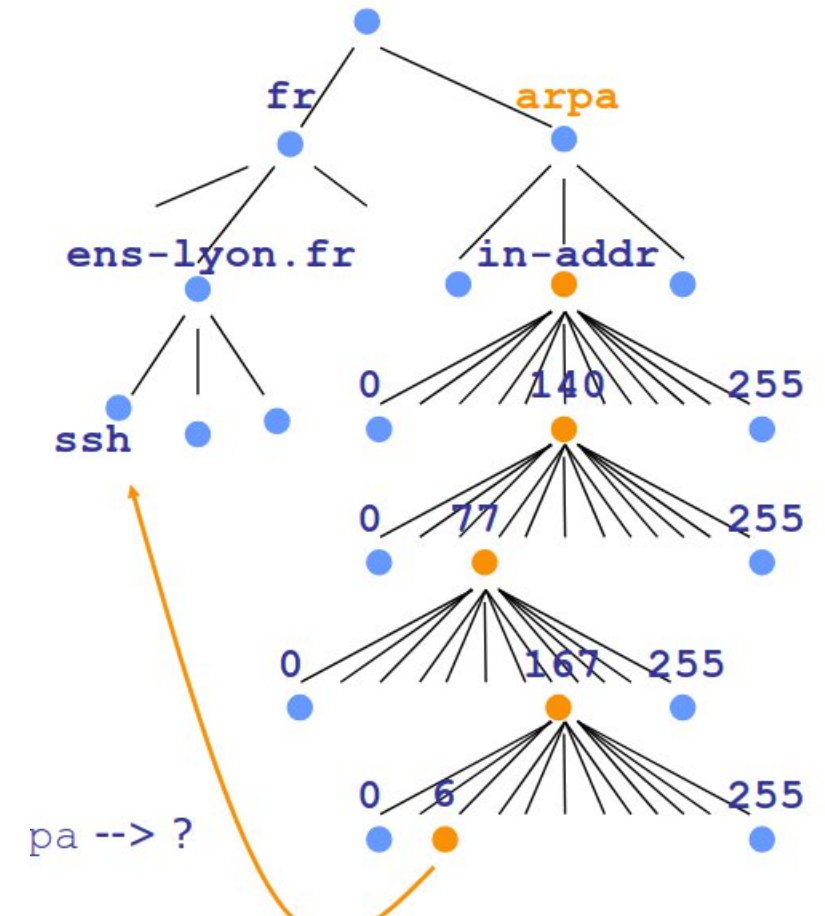
La notion de zone DNS

- Une zone = la partie d'un domaine gérée par une administration
- Au moins un serveur primaire et un secondaire (redondance)
- Le secondaire se synchronise depuis le primaire
- Délégation des sous-domaines à d'autres zones



La résolution de noms inverse

- Retrouver un nom à partir d'une adresse IP
- Domaine spécial in-addr.arpa (et ip6.arpa pour IPv6)
- Enregistrements de type PTR
- Utile pour la journalisation et certaines vérifications
- Résolution
 - ssh.ens-lyon.fr --> ?
- Résolution inverse
 - 6.167.77.140.in-addr.arpa --> ?



Les types de serveurs DNS : root

- Serveurs racine (root) : sommet de la hiérarchie
 - il existe actuellement 13 serveurs racine dans l'Internet (liste sur <http://gnso.icann.org/gtld-registries/>)
 - chaque serveur DNS local connaît un serveur de noms racine qu'il peut interroger s'il ne connaît pas une correspondance
 - un serveur de noms racine connaît au moins les serveurs de source autorisée du premier niveau (.fr., ...)
- Serveurs de source autorisée (authoritative) pour une zone
- Serveurs locaux / résolveurs (cache) pour les clients
- Primaire (référence) et secondaires (répliques)

Les types de serveurs DNS : source

- Serveurs racine (root) : sommet de la hiérarchie
- Serveurs de source autorisée (authoritative) pour une zone
 - chaque hôte est enregistré auprès d'au moins deux "authoritative servers" (le primaire et le secondaire), qui stocke son adresse IP et son nom
 - un serveur de noms est dit de source autorisée pour un hôte s'il est responsable de la correspondance nom/@ pour cet hôte (serveur primaire de la zone)
 - un serveur de noms local n'est pas forcément de source autorisée (ex. bat710.univ-lyon1.fr)
- Serveurs locaux / résolveurs (cache) pour les clients
- Primaire (référence) et secondaires (répliques)

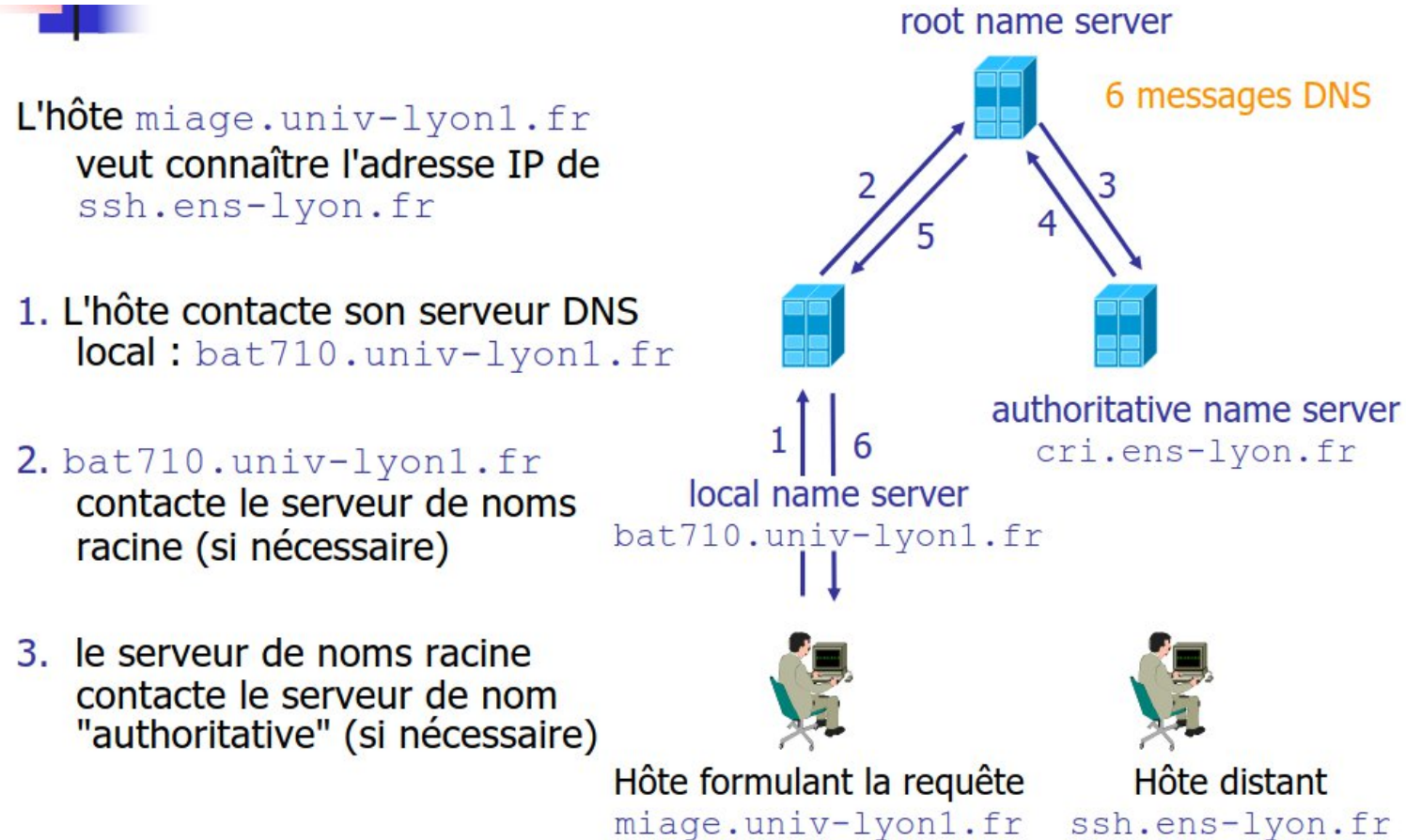
Les types de serveurs DNS : locaux

- Serveurs racine (root) : sommet de la hiérarchie
- Serveurs de source autorisée (authoritative) pour une zone
- Serveurs locaux / résolveurs (cache) pour les clients
 - chaque organisation a un serveur de noms local
 - serveur DNS par défaut de la zone
 - contient parfois les correspondances relatives à la zone de l'organisation
 - toutes les requêtes DNS en provenance de cette organisation vont vers ce serveur de nom local
- Primaire (référence) et secondaires (répliques)

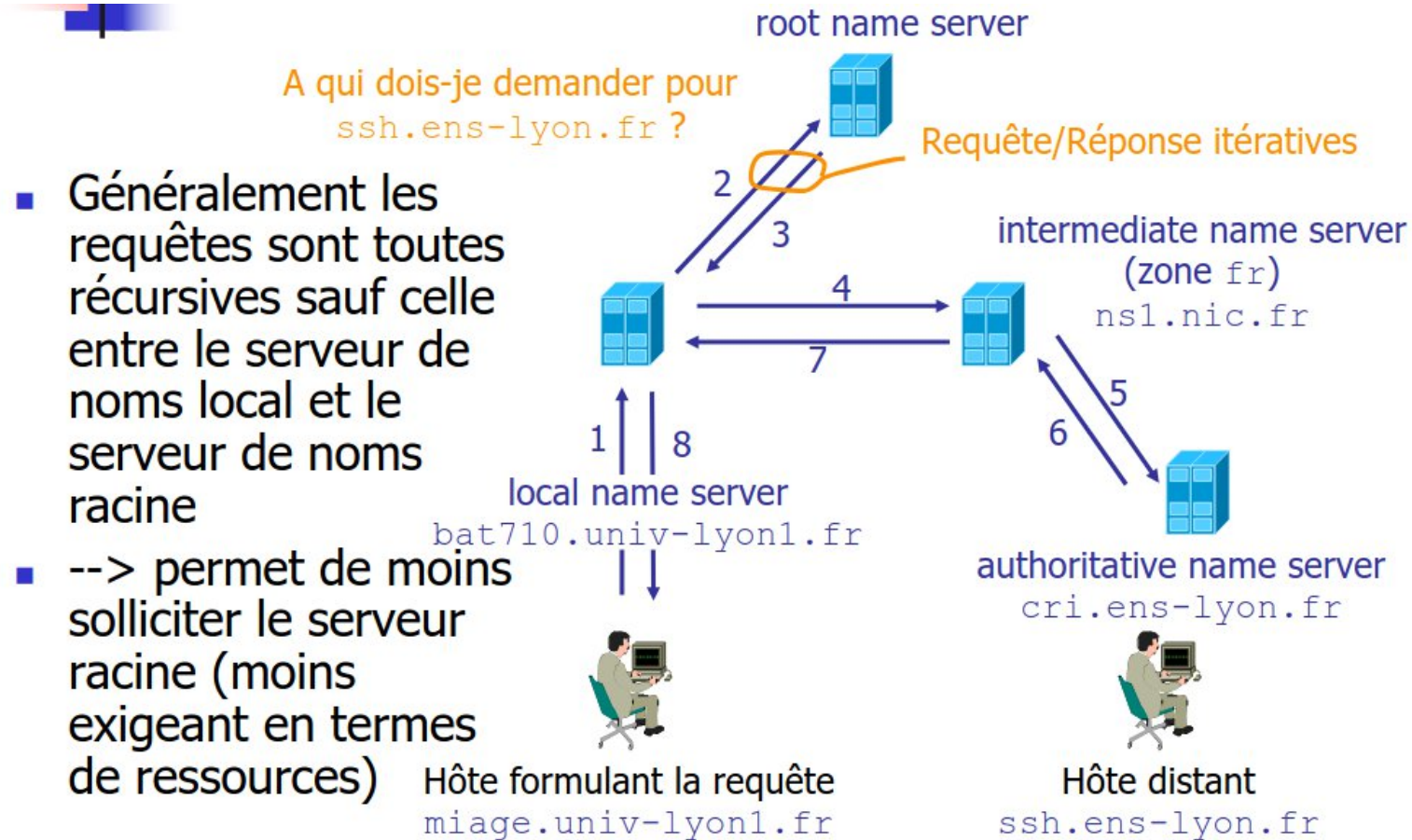
Résolution récursive vs itérative

- Récursive : le serveur fait tout le travail et renvoie la réponse finale
- Itérative : le serveur renvoie l'adresse du serveur suivant à contacter
- Le client interroge en récursif son résolveur local
- Le résolveur interroge en itératif la hiérarchie

Principe d'une résolution récursive



Principe d'une résolution itérative



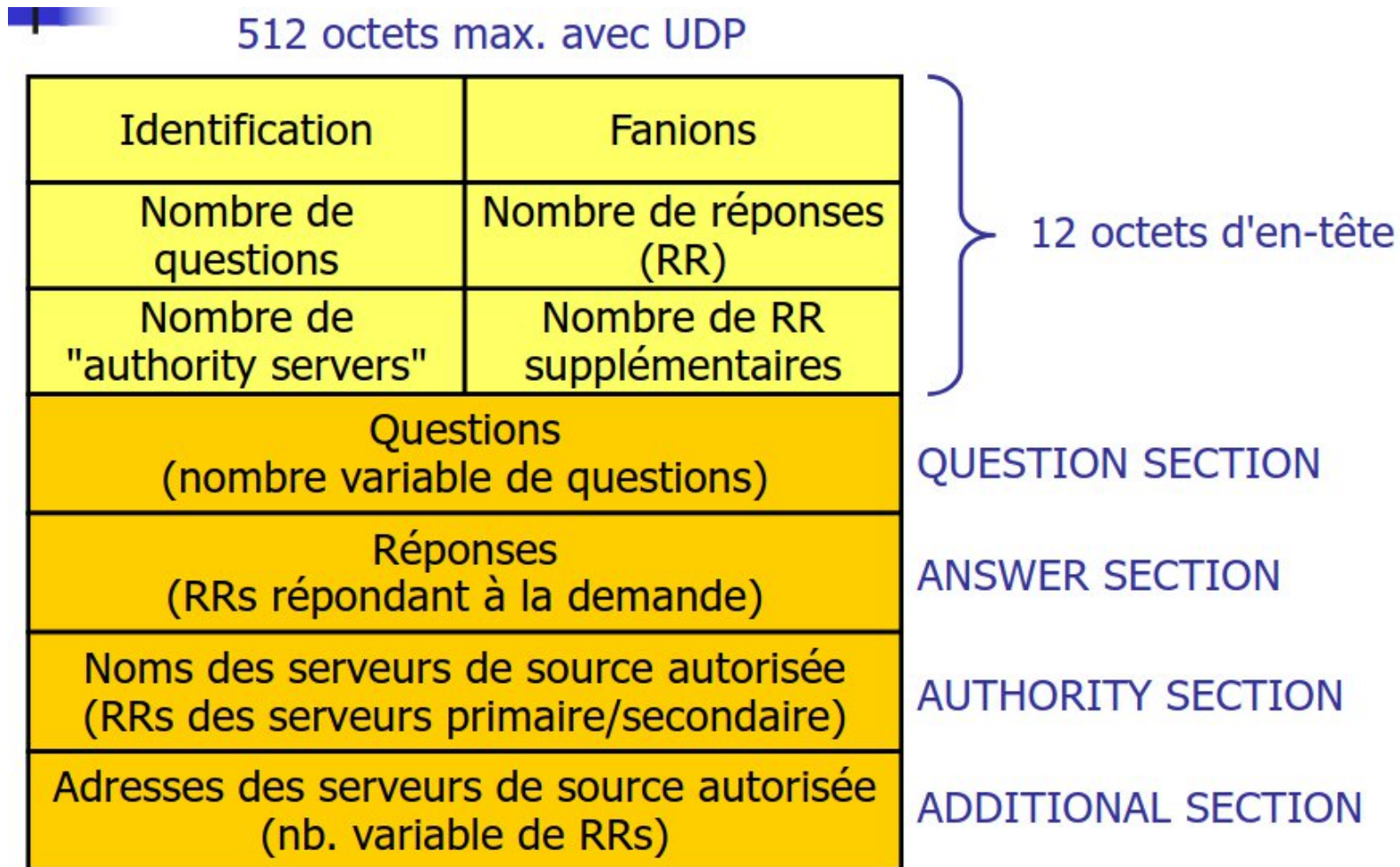
La mise en mémoire cache

- Chaque serveur garde en cache les réponses obtenues
- Réduit le temps de réponse et la charge des serveurs
- Durée de validité fixée par le TTL de l'enregistrement
 - Comme la mémoire n'est pas infinie et que les données du cache peuvent ne plus être valables au bout d'un certain temps, les données expirent du cache après un certain temps TTL (environ 2 jours)
- Compromis fraîcheur / performance

Les messages DNS (RFC 1034, 1035)

- Un message de réponse DNS contient un ou plusieurs RR (Resource Record)
 - l'unité de stockage d'une correspondance dans le cache RR = (Nom, Type, Classe, TTL, Valeur)
 - Type (16 bits) : type de l'enregistrement ; la signification de Nom (N octets) et Valeur dépend du type
 - Classe (16 bits) : famille de protocoles ; Classe=1 pour Internet (IN)
 - TTL (32 bits) : temps en secondes pendant lequel il faut conserver l'enregistrement dans le cache
 - Valeur = RDLENGTH (16 bits) + RDATA (N octets)
- Un message de requête DNS contient des questions du type (Nom, Type, Classe)
- 12 octets d'en tête

DNS : format des messages



DNS : format des messages

Types d'enregistrements (RR) courants

- A / AAAA : nom → adresse IPv4 / IPv6
 - A : Nom=nom d'hôte (canonique), Value=@IPv4
 - AAAA : Nom=nom d'hôte, Value=@IPv6
- CNAME : alias (nom canonique)
 - Nom=un alias, Value=nom canonique (le vrai nom)
- MX : serveur de messagerie d'un domaine
 - Nom=un alias, Value=nom canonique d'un serveur de mail
- NS : serveur de noms ; PTR : adresse → nom ; SOA : paramètres de zone
 - NS : Nom=domaine, Value=nom du serveur de noms
 - PTR : Nom=un nom de la zone arpa, Value=nom canonique (valeur pointée)
 - SOA : Nom=nom d'une zone, Value=informations sur la zone

Types d'enregistrements (RR) courants

```
loic@Precichat:~$ host -a ens-lyon.fr
Trying "ens-lyon.fr"
Trying "ens-lyon.fr"
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 34911
;; flags: qr rd ra; QUERY: 1, ANSWER: 12, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;ens-lyon.fr.                IN      ANY

;; ANSWER SECTION:
ens-lyon.fr.                7200    IN      A       140.77.168.21
ens-lyon.fr.                7200    IN      MX      20 mx2.relay.renater.fr.
ens-lyon.fr.                7200    IN      MX      20 mx1.relay.renater.fr.
ens-lyon.fr.                7200    IN      TXT     "MS=F3E66FBE52E528A11B1F4DB8C7BFDDAFFD6931F1"
ens-lyon.fr.                7200    IN      TXT     "atlassian-domain-verification=xeE9GWBecoLzUAIWw3PlqtBIKzeYPQpT4Y1GqctRGb8AfZGDQwTFe8gYl20wVG5Z"
ens-lyon.fr.                7200    IN      TXT     "google-site-verification=b4X2I2_m4paDidixn4ULYuBxGcN0nPMZW1jycPZ5p4"
ens-lyon.fr.                7200    IN      TXT     "v=spf1 include:spf.ens-lyon.fr include:spf.relay.renater.fr +mx -all"
ens-lyon.fr.                7200    IN      SOA     dns.ens-lyon.fr. root.ens-lyon.fr. 2026090401 21600 3600 3600000 7200
ens-lyon.fr.                7200    IN      NS      dns2.ens-lyon.fr.
ens-lyon.fr.                7200    IN      NS      dns.ens-lyon.fr.
ens-lyon.fr.                7200    IN      NS      ccpnvx.in2p3.fr.
ens-lyon.fr.                7200    IN      NS      ccpntc3.in2p3.fr.
```

Répartition de charge : DNS round-robin

- Plusieurs adresses associées à un même nom
- Le serveur renvoie les adresses en alternance
- Répartit les clients entre plusieurs serveurs
- Simple, mais sans connaissance de la charge réelle
- Solution anycast (une seule @IP annoncée via BGP)
 - Load balacing via la couche réseau
 - Les datacenters Google annonce la même IP

```
loic@Precichat:~$ host -t a google.fr
google.fr has address 142.251.209.163
loic@Precichat:~$ host -t a debian.org
debian.org has address 151.101.2.132
debian.org has address 151.101.66.132
debian.org has address 151.101.130.132
debian.org has address 151.101.194.132
```

DNS : côté client et côté serveur

Côté client : le resolver

- Bibliothèque qui effectue les résolutions pour les applications
- Le resolver a en charge les résolutions de noms (inverse ou pas) chaque fois que cela est nécessaire -> `man resolver`
- Deux fichiers de configuration lui sont associés
 - `/etc/resolv.conf` permet de paramétrer les requêtes DNS effectuées (`man resolv.conf`)
 - `/etc/host.conf` permet de paramétrer le resolver (`man host.conf`), en particulier ordre de résolution
 - `order hosts,bind,nis`
 - `/etc/nsswitch.conf` est consulté en premier s'il existe
- Extrait de l'API du resolver pour les applications
 - `gethostbyname(name)`
 - `gethostbyaddr(addr)`

Côté serveur : named (BIND)

- BIND : Berkeley Internet Name Domain
 - <http://www.isc.org/sw/bind>
 - implantation d'un serveur DNS du domaine publique
- Le démon répondant aux requêtes DNS est named
 - fichier de configuration : named.conf
 - il faut y associer les fichiers décrivant les zones administrées (syntaxe master files : voir RFC 1035)
 - --> dans /etc/namedb ou /etc/bind
- Des utilitaires
 - rndc permet de contrôler à distance le fonctionnement de named (avec authentification)
 - named-checkconf et named-checkzone permettent de vérifier la syntaxe des fichiers de zones ou config.

LDAP : un annuaire fédérateur sécurisé

Problématique résolue par LDAP

- Fusionner de multiples bases (comptes, mail, téléphone...) en un annuaire unique
- Éviter la ressaisie et les incohérences entre applications
- Un référentiel commun, interrogeable par de nombreux services
- Accès en lecture optimisé, écriture plus rare

Le concept d'annuaire

- Annuaire informatique
 - Service permettant d'accéder à des informations relatives à des personnes, des machines (ou autres ressources) de manière organisée
 - Objectif : maintenir de façon cohérente et contrôlée une grande quantité de données
- Base d'information spécialisée : personnes, machines, services...
- Optimisé pour la lecture et la recherche (peu d'écritures)
- Organisation hiérarchique (arborescente)
- Différent d'une base de données relationnelle (SGBD)

Concept d'annuaire

- Différences annuaire/SGBD - dans un annuaire :
 - Pas de liens de dépendances entre les objets stockés
 - Les objets peuvent être distribués sur plusieurs annuaires pour assurer une meilleure disponibilité
 - Le schéma de stockage des données est standardisé pour assurer un partage des données
 - Les applications de l'annuaire n'ont pas besoin de connaître la structure interne des données stockées
 - Un annuaire est principalement consulté en lecture et optimisé pour cela

L'annuaire LDAP

- LDAP : Lightweight Directory Access Protocol
- Héritier de X.500 (allégé, sur TCP/IP)
- Standardisé par l'IETF (RFC 4510 et suivantes)
- Objectif : accès simple, fiable et contrôlé à des informations partagées
 - Fournir aux utilisateurs des informations fiables, facilement accessibles
 - Permettre aux utilisateurs de mettre à jour eux-mêmes leurs informations personnelles
 - Rendre les informations accessibles de façon contrôlée
 - Faciliter le nomadisme des utilisateurs
 - Eviter la redondance d'informations : un seul annuaire pour l'ensemble des services
 - Faciliter la gestion (administration) des postes de travail, des équipements réseau

L'annuaire LDAP

- Un modèle d'information : type des informations contenues dans l'annuaire
- Un protocole d'accès : comment accéder aux informations contenues dans l'annuaire
- Un modèle de nommage : comment l'information est organisée et référencée
- Un modèle fonctionnel : une syntaxe des requêtes permettant l'interrogation de la base et la mise à jour des informations
- Un modèle de duplication : comment la base est répartie sur différents serveurs (tolérance aux pannes, répartition de la charge du serveur, ...)
- Un modèle de sécurité : comment contrôler l'accès aux données ainsi que leur transfert

Le protocole LDAP

- Définit les échanges Client/Serveur (connexion, opérations, résultats)
 - Commandes de connexion au service : bind, unbind, abandon (le client abandonne la requête en cours)
 - Commandes de mises à jour des entrées de l'annuaire : add, delete, modify, rename
 - Commandes d'interrogation : recherche (search) et comparaison (compare) d'entrées
 - La réplication (replication service), en cours de normalisation (LDUP : LDAP Duplication Protocol)
 - Créer des liens entre différents annuaires (referral service) - défini dans LDAPv3
- Transport TCP (port 389 ; 636 pour LDAPS)

Le protocole LDAP

- Définit les mécanismes de sécurité
 - méthodes d'authentification pour se connecter à l'annuaire (qui peut se connecter à l'annuaire et comment)
 - mécanismes de règles d'accès aux données (une fois connecté, à quoi peut-on accéder et avec quels droits)
 - mécanismes de chiffrement des transactions

Le protocole LDAP

- LDAPv3 est conçu pour être extensible sans avoir à modifier la norme
 - permet l'ajout d'opérations (en plus des 9 de base)
 - permet l'ajout de paramètres associés à une opération
 - les mécanismes de sécurité sont définis dans une couche séparé : permet des méthodes d'authentification externes

```
loic@Precichat:~$ cat /etc/services | grep ldap
ldap      389/tcp          # Lightweight Directory Access Protocol
ldap      389/udp
ldaps     636/tcp          # LDAP over SSL
ldaps     636/udp
loic@Precichat:~$
```

Se connecter à une base LDAP

- Outils en ligne de commande (ldapsearch, ldapadd...) ou navigateurs graphiques
- Connexion anonyme ou authentifiée (bind avec un DN)
- Chiffrement via TLS (LDAPS, STARTTLS)
- Base identifiée par un DN de base (ex. dc=univ-lyon1,dc=fr)

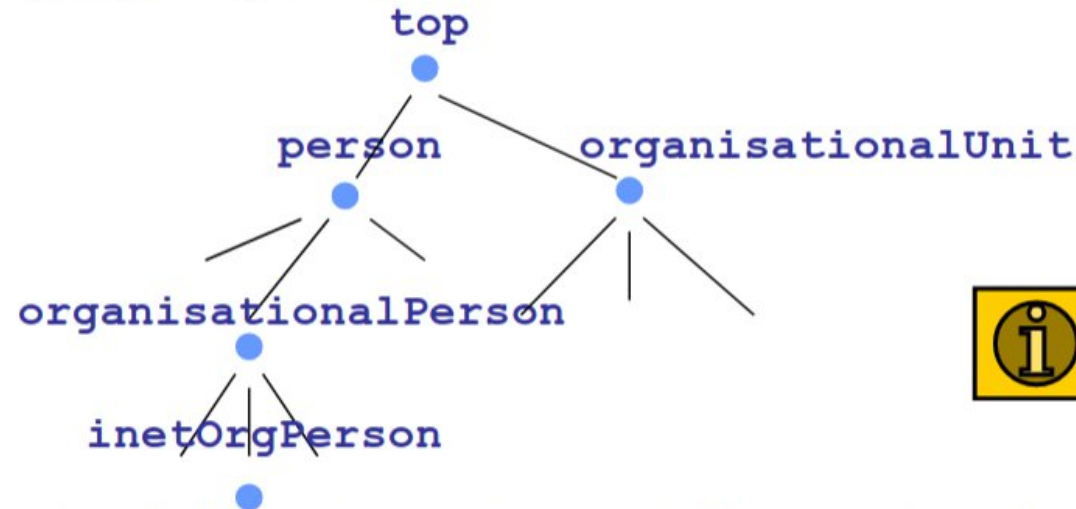
Les modèles LDAP

Le modèle d'information : entrées et attributs

- Un schéma LDAP est standardisé (IANA) : pour l'interopérabilité entre logiciels
- L'information est structurée en entrées (objets)
- Chaque entrée possède des attributs (nom, valeur)
 - un nom, un identifiant unique (OID), mono/multi-valué, une syntaxe et des règles de comparaison (matching rules), une valeur (format+taille limite), modifiable ou non
- Les classes d'objets définissent les attributs obligatoires / optionnels
 - des objets réels : un compte Unix (posixAccount), une organisation (o), un département (ou), un personnel (organizationPerson), une imprimante (device), ...
 - ou abstraits : l'objet père de tous les autres (top), ...
- Les classes forment une hiérarchie (héritage)

Modèle d'information : hiérarchie des classes d'objets

- Les classes d'objets forment une structure arborescente : tout en haut, l'objet `top`



- Chaque objet hérite des attributs de l'objet dont il est le fils
- Plus d'infos :
 - <http://www.it.ufl.edu/projects/directory/ldap-schema/>
 - <http://ldap.akbkhome.com/>

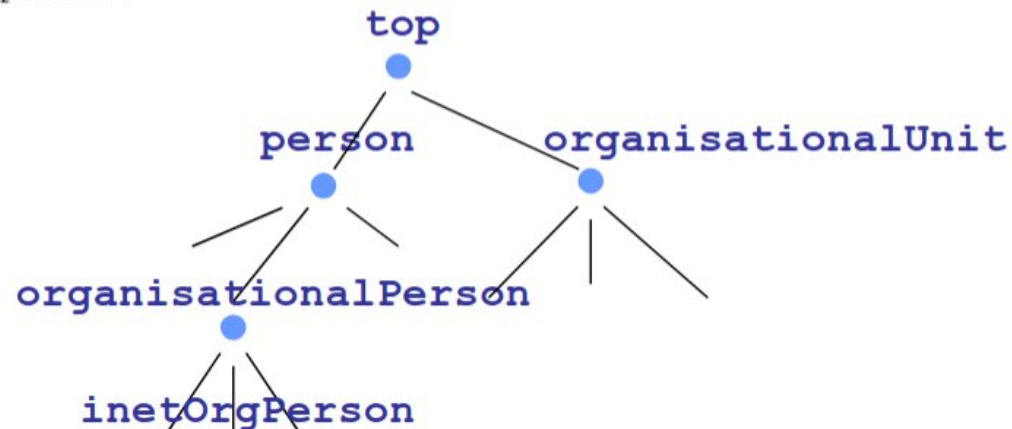
Modèle d'information : hiérarchie des classes d'objets

```
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
```

L'objet **person** a comme attributs : `commonName`, `surname`, `description`, `seeAlso`, `telephoneNumber`, `userPassword`

L'objet fils **organizationalPerson** ajoute des attributs comme : `organizationUnitName`, `title`, `postalAddress...`

L'objet petit-fils **inetOrgPerson** lui rajoute des attributs comme : `mail`, `labeledURI`, `uid` (`userID`), `photo...`

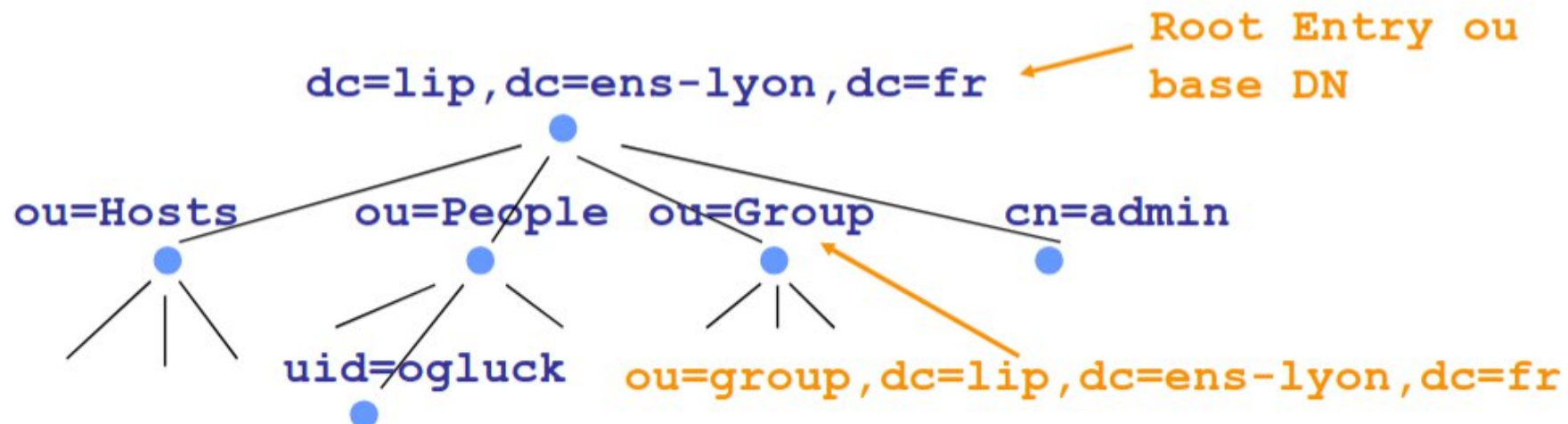


Le modèle de nommage

- Il définit comment sont organisées les entrées (=objets) de l'annuaire et comment elles sont référencées
- Structure arborescente contenant deux catégories d'objets
 - les conteneurs (une zone de rangement) : départ d'une nouvelle branche
 - peuvent contenir des conteneurs ou des feuilles
 - généralement, une sous-organisation de l'organisation (département, zone géographique, ...)
 - les feuilles (véritables données) : terminaison des branches (généralement les machines, les utilisateurs, ...)

Le modèle de nommage : le DIT

- Les entrées sont organisées en arbre : le DIT (Directory Information Tree)
- Chaque entrée a un nom unique : le DN (Distinguished Name)
- Composé de RDN (Relative Distinguished Name)
- Le DN reflète la position dans l'arborescence



Le format LDIF

- LDAP Data Interchange Format : représentation texte des entrées
- Sert à importer / exporter et à modifier l'annuaire
- Une entrée = un DN suivi de ses attributs
- Format lisible et automatisable

```
dn: uid=ogluck,ou=People,dc=lip,dc=ens-lyon,dc=fr
objectClass: top
objectClass: account
objectClass: posixAccount
objectClass: shadowAccount
uid: ogluck
uidNumber: 44132
gidNumber: 200
homeDirectory: /home/toto
cn: Olivier GLUCK
loginShell: /bin/bash
```

Le modèle fonctionnel : les opérations

- Interrogation : recherche (search), comparaison (compare)
- Mise à jour : ajout, modification, suppression
- Connexion / déconnexion : bind, unbind
- Une recherche = base + portée + filtre + attributs demandés
- Rappel des opérations de consultation/mise à jour
 - opérations de mise à jour des entrées de l'annuaire : add, delete, modify, rename
 - opérations d'interrogation : recherche (search) et comparaison (compare) d'entrées
 - --> il n'y a pas d'opération de lecture d'une entrée : pour connaître le contenu d'une entrée, il faut écrire une requête qui pointe sur cette entrée

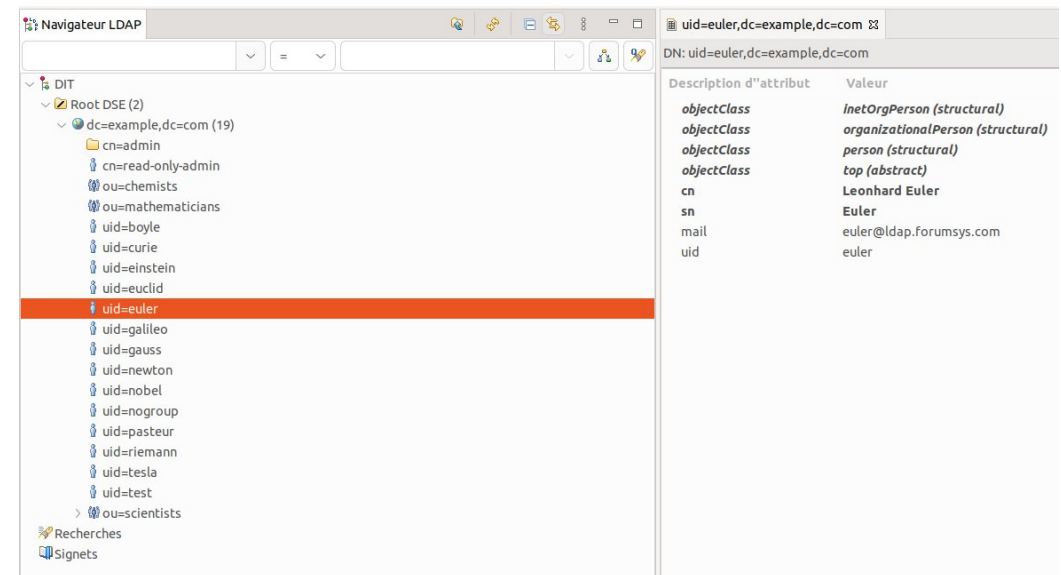
Une requête de recherche : les paramètres

- Une requête est composée de 8 paramètres

<code>base object</code>	l'endroit de l'arbre où doit commencer la recherche
<code>scope</code>	la profondeur de la recherche
<code>derefAliases</code>	si on suit les liens ou pas
<code>size limit</code>	nombre de réponses limite
<code>time limit</code>	temps maxi alloué pour la recherche
<code>attrOnly</code>	renvoie ou pas la valeur des attributs en plus de leur type
<code>search filter</code>	le filtre de recherche
<code>list of attributes</code>	la liste des attributs que l'on souhaite connaître

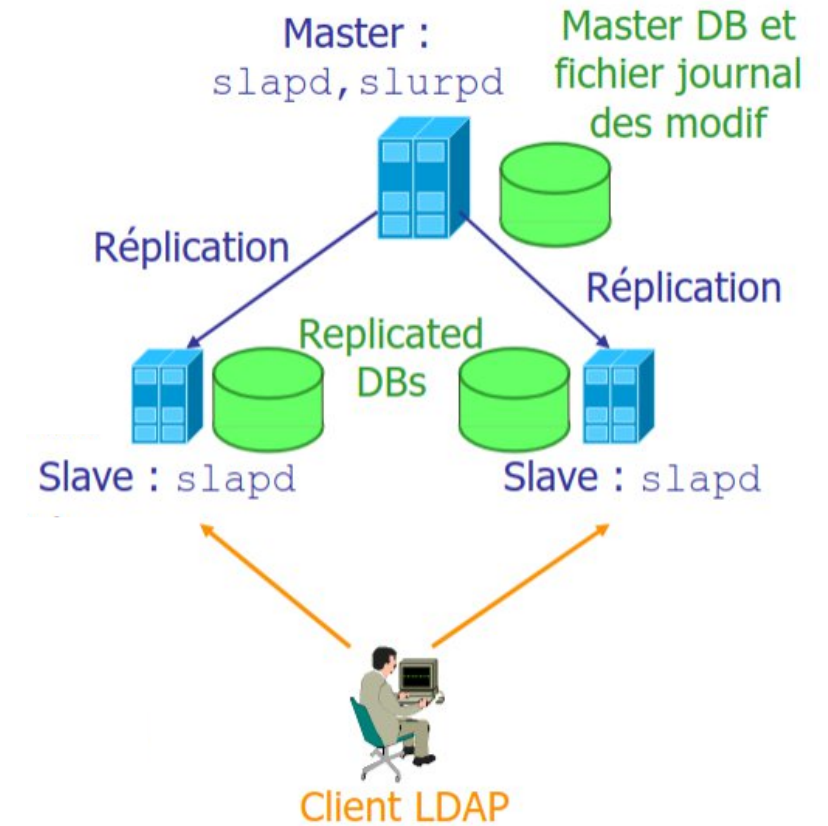
Les URL LDAP (RFC 4516)

- Désignent une base et une requête sous forme d'URL
- `ldap://serveur/base?attributs?portée?filtre`
- Permettent aux applications de pointer directement une entrée
- Utilisées dans la configuration des services



Le modèle de duplication (réplication)

- Répliquer l'annuaire sur plusieurs serveurs
- Améliore la disponibilité et le temps de réponse
- Un maître et des réplicas (ou multi-maîtres)
- Réplication des mises à jour du maître vers les réplicas



Le modèle de sécurité

- Authentification au bind : anonyme, simple, ou SASL (Kerberos par exemple)
 - Anonymous authentication, Root DN/passwd
 - Authentification (administrateur), User DN/passwd
 - DN : cn=jean.dupont,ou=users,dc=entreprise,dc=fr
 - Mot de passe : *****
- Chiffrement des échanges : TLS
- Contrôle d'accès par entrée et par attribut (Access Control List)
- Recommandé : chiffrement + authentification forte (Kerberos)

Mettre en place un annuaire LDAP

Questions sur la mise en place

- Il faut bien choisir les schémas
 - Quelles informations veut on stocker dans l'annuaire ?
 - --> choix des objets contenant les attributs désirés
 - Quelles sont les applications qui vont utiliser l'annuaire ?
 - Authentification des utilisateurs sous Unix, sous Windows (samba), gestion des groupes d'utilisateurs, listes de mail dynamiques (sympa), carnets d'adresses Netscape, ... ?
- Il faut réfléchir à l'organisation du DIT
 - impacts sur la performance, les droits d'accès, ...
- Puis dans un second temps
 - gestion centralisée sur un seul serveur ?
 - nombre de serveurs redondants ? Emplacement ?

OpenLDAP

- Implémentation libre de LDAP (domaine public)
- Démon serveur : slapd
- Outils : ldapsearch, ldapadd, ldapmodify, ldapdelete...
- Configuration moderne : slapd.d (dynamique)

Configuration et alimentation : /etc/ldap/slapd.conf

- Déclarer les schémas, la base (suffixe) et les ACL
- Alimenter l'annuaire avec des fichiers LDIF (ldapadd)
- Interroger avec ldapsearch (filtres)
- Sauvegarder / exporter en LDIF

Configuration et alimentation : /etc/ldap/slapd.conf

- Permet de configurer le démon slapd
 - définition des schémas utilisés : **include /etc/ldap/schema/inetorgperson.schema**
 - définition du backend (moteur de base de données utilisé pour ranger les données) : database ldbm (ldbm par défaut, sinon sql, ...)
- définition de la base de l'annuaire et de l'administrateur
 - le suffixe : racine de l'arbre : **suffix "dc=lip,dc=ens-lyon,dc=fr"**
 - l'administrateur et son mot de passe : **rootdn cn=admin,dc=lip,dc=ens-lyon,dc=fr / rootpw toto**
- le répertoire où la base est stockée
directory "/var/lib/ldap"

Configuration et alimentation : /etc/ldap/slapd.conf

■ définitions des ACLs (man slapd.access)

Format d'un ACL :

```
access to <what> [ by <who> <access> [ <control> ] ]+
```

<what>: *, un dn, un filtre LDAP, une liste d'attributs (attrs=...)

<who> : *, dn, anonymous, users (quelqu'un authentifié), self (le proprio), ...

<access> : none, auth, compare, search, read, write, ...

<control> : stop, continue, break (imbrication des règles...)

Par défaut :

```
access to attrs=userPassword
```

```
    by dn="" write # l'admin
```

```
    by anonymous auth # droit de lecture uniquement lors du bind
```

```
    by self write # le propriétaire
```

```
    by * none
```

The admin dn has full write access

```
access to *
```

```
    by dn="" write
```

```
    by * read # nécessaire d'avoir read pour le bind
```

Configuration et alimentation : /etc/ldap/slapd.conf

- définition des réplicats

- sur le serveur maître

- # fichier dans lequel slapd stocke les modifications pour slurpd

- replogfile /var/lib/ldap/replog

- # définition d'un réplicat

- replica host=ldap.ens-lyon.fr:389 bindmethod=... ..

- sur un esclave

- # le dn autorisé à faire la mise à jour

- updatedn "souvent slurpd"

- # URL du maître

- updateref ldap://master-ldap.ens-lyon.fr:389

Liens avec les applications

- Authentification centralisée des utilisateurs
- Unix / Linux : PAM + nsswitch (getent) via LDAP
- Windows : Active Directory est un annuaire LDAP
- Samba : comptes stockés dans LDAP

Authentication Unix via LDAP

- PAM (Pluggable Authentication Modules) délègue à LDAP
 - pour authentication via LDAP, rajouter la ligne auth sufficient pam_ldap.so dans le fichier /etc/pam.d/login qui signifie l'authentication via LDAP est suffisante
- nsswitch.conf : source LDAP pour comptes et groupes
- Les mots de passe sont vérifiés côté annuaire
- Souvent couplé à Kerberos pour la sécurité

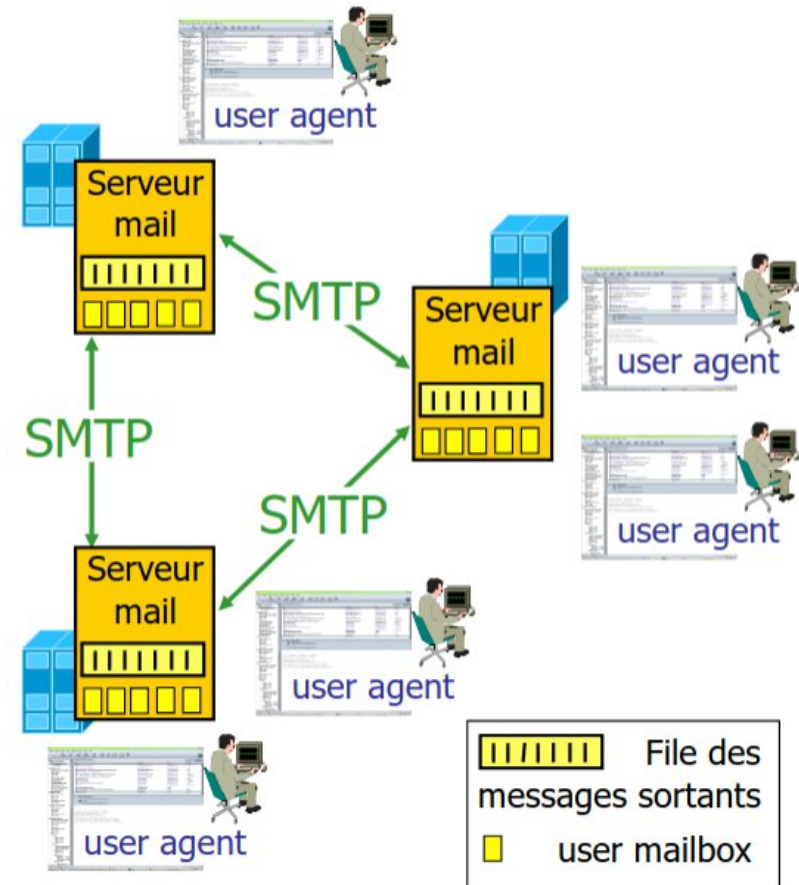
Plan de la partie 2

- Introduction / Rappel
- Applications de connexion à distance (SSH, X)
- Transfert de fichiers (FTP, TFTP, NFS, SMB)
- Annuaire : NIS, DNS, LDAP
- **Messagerie électronique (SMTP, IMAP, MIME)**

La messagerie électronique

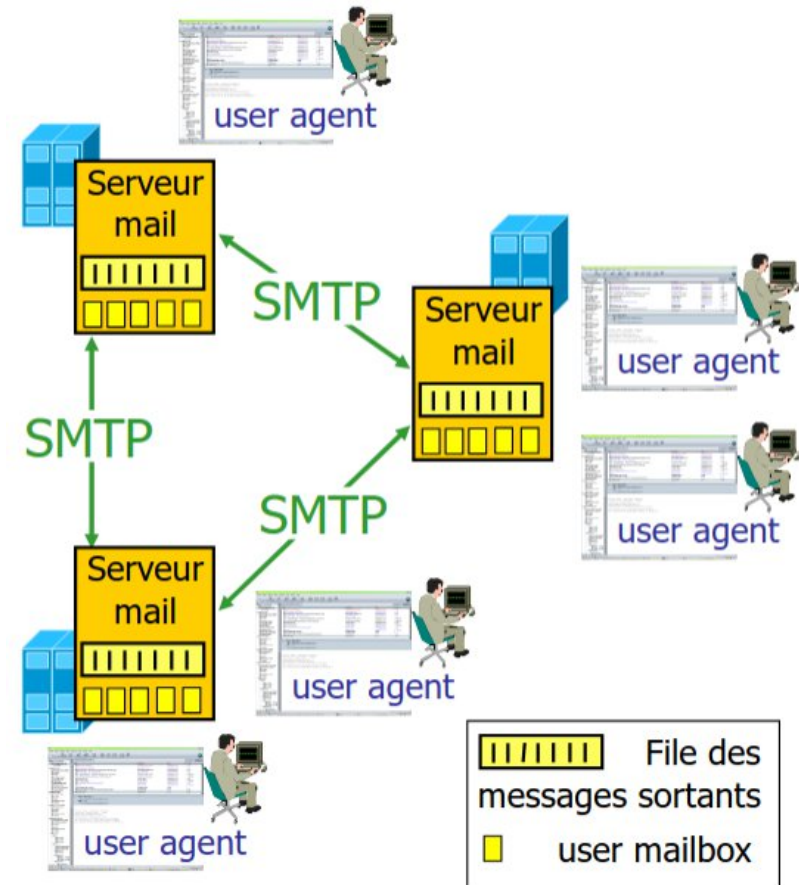
Le courrier électronique : les composants

- 4 composants principaux :
 - des agents utilisateurs
 - des serveurs de mail
 - un protocole de transfert de mail : Simple Mail Transfer Protocol (SMTP)
 - un protocole d'accès à la boîte aux lettres (POP, IMAP, ...)



Le courrier électronique : les composants

- Les messages entrants et sortants sont stockés sur le serveur
- La boîte aux lettres de chaque utilisateur contient les messages entrants (à lire)
- File d'attente de messages mail sortants (à envoyer)
- Protocole SMTP entre les serveurs de mail pour l'envoi des messages
 - modèle C/S : Client (serveur de mail émetteur) - Serveur (serveur de mail récepteur)
 - le client se connecte sur le port 25/TCP du serveur pour transférer son message



Le courrier électronique : les composants

- MUA (Mail User Agent) : le logiciel de l'utilisateur (lecture, rédaction)
- MTA (Mail Transfer Agent) : achemine le courrier entre serveurs (SMTP)
- MDA : dépose le message dans la boîte aux lettres
- Boîte aux lettres + file d'attente des messages sortants

Le courrier électronique : les composants

- Les protocoles d'accès : consultation de sa boîte aux lettres (après authentification)
 - POP3 : Post Office Protocol v3 [RFC 1939]
 - autorisation (agent <--> server) et téléchargement
 - IMAP4 : Internet Message Access Protocol v4 [RFC 3501]
 - plus de caractéristiques, plus complexe, plus récent
 - manipulation de messages stockés sur le serveur
- HTTP (Webmail) : Hotmail , Yahoo! Mail, ...

Analogie avec le courrier papier

- L'expéditeur dépose sa lettre (MUA → serveur d'envoi)
- Les centres de tri acheminent (MTA de proche en proche)
- La lettre arrive dans la boîte du destinataire (MDA)
- Le destinataire relève sa boîte (POP / IMAP)

Configuration d'un agent utilisateur

- Serveur d'envoi : SMTP (avec authentification, port 587)
- Serveur de réception : POP ou IMAP
- Identité (adresse) et chiffrement (TLS)
- Webmail : tout se fait via le navigateur

Serveur sortant (SMTP) : ldesgeor - smtp.ens-lyon.fr ▼ [Modifier le serveur SMTP...](#)

[Gérer les identités...](#)

Paramètres du serveur

Type de serveur : Serveur de courrier IMAP

Nom du serveur : imaps.ens-lyon.fr Port : 993  Défaut : 993

Nom d'utilisateur : ldesgeor

SMTP, codage et types MIME

Le protocole SMTP

- Achemine le courrier entre serveurs (port 25 ; 587 pour la soumission)
- Mode texte : dialogue commandes / réponses
- Trois phases : établissement, transfert, fermeture
 - handshaking (établissement de la connexion)
 - transfert d'un ou plusieurs messages
 - fermeture de la connexion
- Les connexions sont persistentes
 - si plusieurs messages à destination du même serveur sont en attente sur le serveur émetteur, ils transiteront tous sur la même connexion TCP
- Historiquement en clair → aujourd'hui STARTTLS

Le protocole SMTP

- Un message est composé d'un en-tête et d'un corps (RFC 822)
 - les champs de l'en-tête peuvent être positionnés soit par l'agent utilisateur émetteur, soit par le serveur entrant, soit par le serveur sortant
 - un champ d'en-tête est de la forme nom_champ: valeur<CRLF>
 - l'en-tête contient au minimum les champs From et To, très souvent le champ Subject

Le protocole SMTP

- Une succession de Commande/Réponse
 - Commande SMTP : texte ASCII
 - Réponse SMTP : code d'état (status) + phrase
- Un message peut contenir plusieurs objets ; ils sont alors envoyés dans un message "multipart" (contrairement à HTTP : 1 objet = 1 réponse)
- Le serveur SMTP utilise CRLF.CRLF pour reconnaître la fin d'un message
- Les messages (en-tête ET corps) sont transférés en ASCII 7 bits (US-ASCII)

Le codage des messages

- SMTP est prévu pour transférer des caractères US-ASCII sur 7 bits --> problème de la représentation des caractères accentués, du transfert des octets (images...)
- Pour transférer une image ou du texte accentué, l'agent utilisateur émetteur/récepteur doit encoder/décoder le contenu du message
- Encodage quoted-printable :
 - généralement utilisé pour transférer du texte
 - permet le transfert des caractères ASCII étendus (codés sur 8 bits >128) comme les caractères accentués :
 - ils sont codés par les 3 caractères US-ASCII suivants : =xx où xx est le code hexadécimal du caractère à encoder
 - du coup, il faut coder le caractère = différemment : =20

Le codage des messages

- Encodage base64 :
 - généralement utilisé pour transférer des flux d'octets
 - permet le transfert des images ou autre série d'octets en tant que caractères ASCII NVT :
 - 3 octets (24 bits) sont transférés en tant que 4 caractères ASCII NVT : les 3 octets sont découpés en 4 fois 6 bits
 - bourrage avec le caractère = si pas aligné sur 4 caractères
 - permet de ne pas transférer plus de bits que le contenu initial (excepté le bourrage)
- ESMTP [RFC 1425] : une évolution de SMTP qui permet le transfert des messages sans passer au format ASCII NVT
 - transfert de blocs de données sur 8 bits (flux d'octets)
 - spécifié par Content-Transfer-Encoding: 8bit ou Binary dans l'en-tête

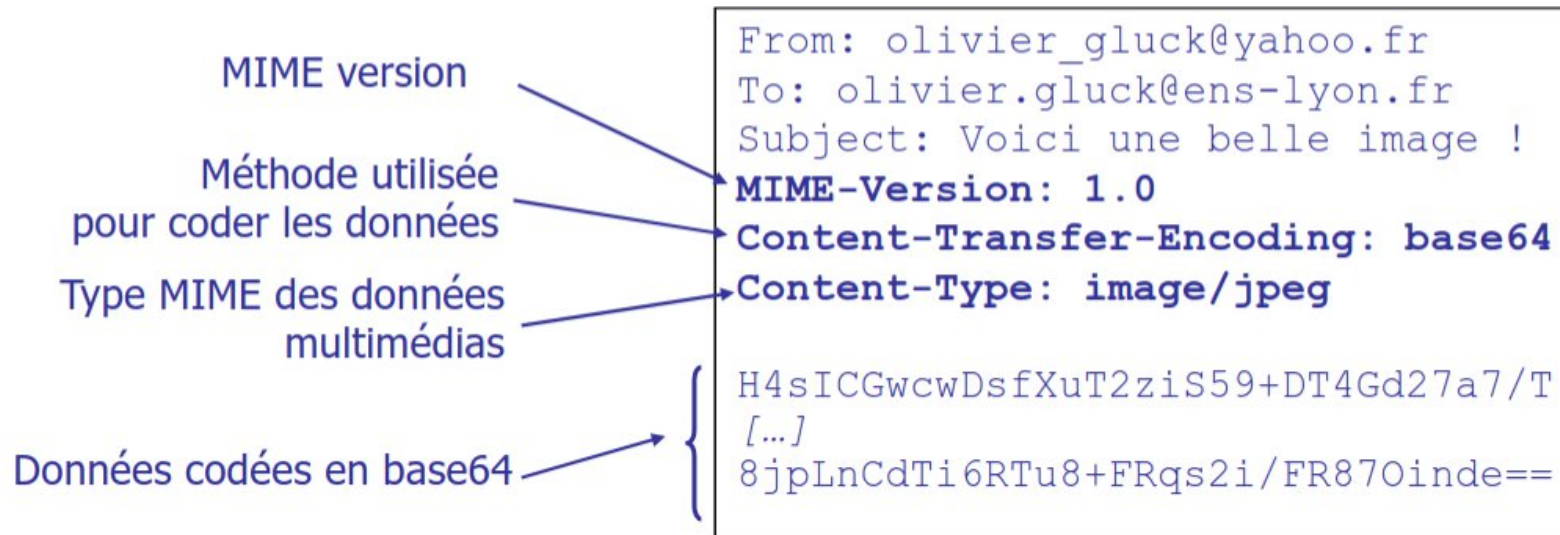
Les types MIME

- MIME : Multi-purpose Internet Mail Extensions
- Permet l'échange de fichiers multimédias entre machines quelconques en spécifiant dans l'en-tête
 - le type du fichier en vue d'un traitement par l'agent utilisateur destinataire
 - le codage des données du fichier
- Les commandes MIME ont été intégrées dans HTTP1.0
- Un type MIME est composé
 - d'un type général (text, image, audio, video, application...)
 - et d'un sous-type (image/gif, image/jpeg, application/pdf, application/rtf, application/msword, text/plain, text/html)
- En perpétuelle évolution
- La machine cliente doit ensuite associer l'exécution d'une application à chaque type MIME

Les types MIME

Content-Type: type/subtype; parameters

- Lignes supplémentaires dans l'en-tête du message pour déclarer un type MIME et un encodage
- Content-type est généralement positionné à partir de l'extension du document demandé (/etc/mime.types)



Les types MIME

```
From: olivier_gluck@yahoo.fr
To: olivier.gluck@ens-lyon.fr
Subject: Voici une belle image mais avec du texte !
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary=98766789

--98766789
Content-Transfer-Encoding: quoted-printable
Content-Type: text/plain

Cher Olivier,
Voici une photo de nos dernieres vacances !
--98766789
Content-Transfer-Encoding: base64
Content-Type: image/jpeg

H4sICGYRMTQAA3NsaWRlcy5wcwDsfXuT2ziS59+DT4Gd275a
56o7LlgSJbFNiWpSqsfw6rvLxPgSxIlVnk64i54ftRKi67/T
[...]
8jpLnCdTi6RTu8+FRqs2i/RTuy56plYbYVsalfdvUjHrtV6g
RTf4/hy67fgIIVDfeR+rtYuNFR87Oinde==
--98766789--
```

Les commandes SMTP

- HELO / EHLO : identification du client
- MAIL FROM / RCPT TO : expéditeur et destinataires
- DATA : corps du message (terminé par une ligne « . »)
- QUIT : fin de session

```
Return-Path: francesco.bronzino@ens-lyon.fr
X-MS-Exchange-Organization-Network-Message-Id: 6647c93c-3e57-4d36-c389-08df0e763cc8
X-MS-Exchange-Organization-PRD: ens-lyon.fr
X-MS-Exchange-Organization-SenderIdResult: Fail
Received-SPF: Fail (SAMBXE01.univ-lyon1.fr: domain of
francesco.bronzino@ens-lyon.fr does not designate 134.214.126.37 as permitted
sender) receiver=SAMBXE01.univ-lyon1.fr; client-ip=134.214.126.37;
helo=mx1.relay.renater.fr;
X-MS-Exchange-Organization-AVStamp-Enterprise: 1.0
X-MS-Exchange-Organization-AuthSource: SAMBXE01.univ-lyon1.fr
X-MS-Exchange-Organization-AuthAs: Anonymous
X-MS-Exchange-Transport-EndToEndLatency: 00:00:00.2238060
X-MS-Exchange-Processed-By-BccFoldering: 15.02.2562.042
MIME-Version: 1.0

Lundi a 14h c=E2=80=99est parfait pour moi, merci !

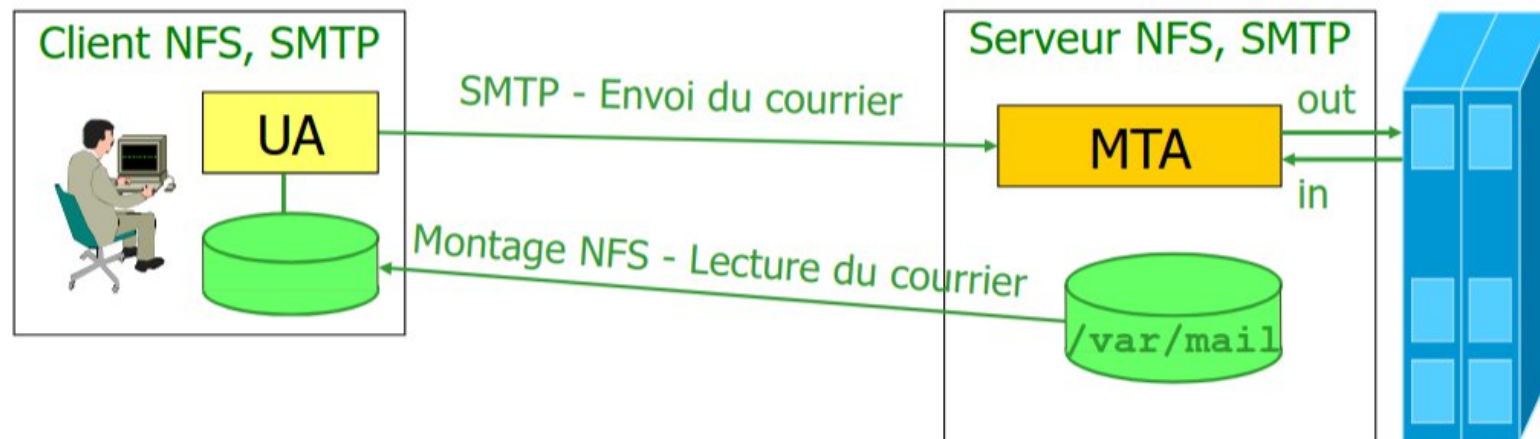
- Francesco

> On Sep 9, 2026, at 15:23, Augustin Laouar <Augustin.Laouar@ens-lyon.fr> w=
rote:
>=20
> =EF=BB=BFSalut,
>=20
> Paul, Lo=C3=Afc et moi m=C3=Ame nous sommes mis d=E2=80=99accord pour d=
=C3=A9placer la r=C3=A9union le lundi apr=C3=A8s-midi =C3=A0 14h.
>=20
> Laisse nous savoir si ce cr=C3=A9neau te convient.
>=20
> Augustin
>=20
>> On 9 Sep 2026, at 13:52, Francesco Bronzino <francesco.bronzino@ens-lyon=
.fr> wrote:
```

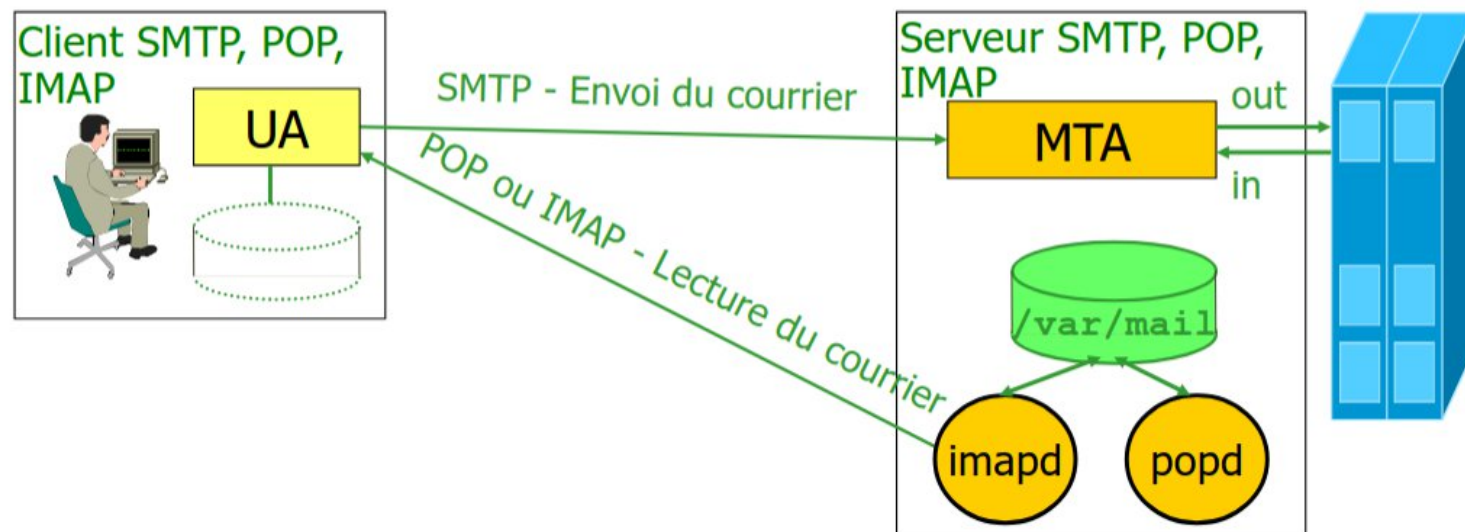
Accès à la boîte aux lettres

Accès à sa BAL

- Par lecture directe du fichier



- Via un protocole spécifique



Relever son courrier : POP et IMAP

- Accès direct au fichier (local ou via NFS) : peu pratique à distance
- POP3 (RFC 1939) : téléchargement et effacement (client nomade simple)
- IMAP (RFC 3501) : garde les messages sur le serveur (multi-appareils)
- Aujourd'hui : IMAP (ou webmail) est la norme

POP3

- Phase d'autorisation
 - Commandes client :
 - user: déclare username
 - pass: password
 - Deux réponses possible du serveur :
 - +OK
 - -ERR
- Phase de transaction
 - list: liste les numéros de messages et leur taille
 - retr: rappatrie un message à partir de son numéro
 - dele: efface un message
 - quit

```
S: +OK POP3 server ready
C: user alice
S: +OK
C: pass hungry
S: +OK user successfully logged on
```

```
C: list
S: 1 498
S: 2 912
S: .
C: retr 1
S: <contenu du message 1>
S: .
C: dele 1
C: retr 2
S: <contenu du message 2>
S: .
C: dele 2
C: quit
S: +OK POP3 server signing off
```

POP3

- POP3 est extrêmement simple
 - permet uniquement de télécharger des messages depuis le serveur en laissant éventuellement une copie de ceux-ci dans la BAL de l'utilisateur
 - pas adapté aux utilisateurs nomades
 - impossible de gérer des répertoires sur le serveur
 - impossible de gérer les messages en les laissant sur le serveur
- IMAP répond à cette problématique au prix d'un protocole beaucoup plus complexe

IMAP

- IMAP permet la gestion distante des messages
 - associe un message à un répertoire distant sur le serveur
 - permet à l'utilisateur de faire une recherche dans les messages sur le serveur
 - permet de ne consulter que des extraits de messages (par exemple que l'en-tête ou que la partie texte d'un message multipart...)
 - contrairement à POP3, IMAP conserve des informations d'état sur chaque utilisateur (noms des répertoires, listes des messages qu'ils contiennent...)

POP3 vs IMAP

- POP3 : simple, messages rapatriés puis souvent supprimés du serveur
- IMAP : gestion des dossiers côté serveur, synchronisation multi-clients
- IMAP conserve l'état (dossiers, drapeaux lus / non lus)
- IMAP plus riche mais plus complexe

L'accès Webmail

- Pas de protocole d'accès spécifique
 - l'utilisateur utilise un navigateur Web comme agent utilisateur pour consulter/envoyer ses courriers
- Utilise le protocole HTTP (ou HTTPS) pour communiquer avec les serveurs SMTP/IMAP
 - le serveur HTTP exécute des scripts qui utilisent
 - le protocole IMAP pour communiquer avec le serveur IMAP et ainsi manipuler les messages distants de l'utilisateur
 - le protocole SMTP pour traduire une demande d'envoi d'un message de la part de l'utilisateur
- Interface web (HTTP / HTTPS) devant le serveur de mail
- Rien à installer ni configurer côté client
- Le serveur web dialogue en SMTP / IMAP pour l'utilisateur
- Accessible partout, avec un simple navigateur

Les alias

- Rediriger une adresse vers une autre (ou une liste)
- Utile pour les rôles (postmaster, webmaster...) et les listes
- Évite de dépendre du login réel de l'utilisateur

Une bonne architecture de serveur mail

- Séparer soumission (SMTP + AUTH + STARTTLS) et transport
- Stockage des boîtes (Maildir) et accès IMAP (IMAPS)
- Webmail en HTTPS devant l'IMAP
- Filtrage antispam / antivirus, chiffrement partout

