



# Services Réseau

*Licence Pro. CGIR*

IUT “A” de Lille, Département informatique

Michaël Hauspie

`Michael.Hauspie@univ-lille.fr`

`https://pro.univ-lille.fr/michael-hauspie`

PGP: 9A33 192E 0B7F 6AC4 DCDB EC11 E10C 62E9 75B4 06A8

# Organisation

## 👉 Organisation générale

- 12 semaines
- Environ 1 heure de cours et 3 heures de TP
- TP évalués tout au long du module

## 👉 Contenu

- Description des protocoles
- Pratique du protocole en TP
- Installation et configuration des services usuels implémentant les protocoles (sous linux)

Partie A

# Introduction



IUT A

Cours n° A.1

# L'administration système

# Les tâches de l'administrateur système

- ☞ Installation des stations de travail
- ☞ Gestion des comptes utilisateurs
- ☞ Installation des logiciels
- ☞ Maximiser l'utilisation des ressources
- ☞ Assurer la sécurité des données
- ☞ Architecturer le réseau :
  - \* pour une communication maximale
  - \* pour minimiser la congestion
  - \* pour sécuriser le réseau.
- ☞ Surveiller le fonctionnement :
  - \* du système et des services
  - \* du réseau et le trafic réseau

# En résumé

- 👉 Gérer un parc hétérogène ..... mais donner une interface standard
- 👉 Ne pas trop s'épuiser pour ..... rester calme face aux utilisateurs

# Les différents OS

## ☞ Systèmes propriétaires :

- ☞ Windows (NT, 2000, XP, 2003, Vista, 7, etc.)
- ☞ Netware
- ☞ UNIX (Solaris, AIX, SCO, HP-UX, IRIX, etc.)
- ☞ VMS
- ☞ MVS
- ☞ i5/OS (ex OS/400)

## ☞ Systèmes libres :

- ☞ UNIX (Linux, FreeBSD, OpenBSD, NetBSD, Hurd)

# La famille UNIX

Différents types d'UNIX :

☞ **BSD** originaire de l'Université de Berkeley :

- ☞ SunOS 4.x
- ☞ Linux (un peu *batard*)
- ☞ MacOS X

☞ **System V** originaire de l'industrie :

- ☞ Solaris (aka SunOS 5.x)
- ☞ HP-UX
- ☞ IRIX

☞ **POSIX** tentative de normalisation :

- ☞ OSF-1
- ☞ Linux

<http://www.levenez.com/unix>



# Documentations

L'administrateur système doit savoir **trouver** l'information :

- ① grâce à son savoir
- ② grâce aux documentations :
  - documentations en ligne
  - livres
- ③ via les gens qui savent :
  - collègues
  - forum de discussions
  - liste de diffusions de mails

RTFM = Read The ***Fantastic*** Manual

# Manuel UNIX

Le manuel d'UNIX est accessible en ligne grâce aux commandes :

`man`, `apropos` et `whatis`.

Un extrait sous Linux de la commande : `man man`

The table below shows the section numbers of the manual followed by the types of pages they contain.

- 1 Executable programs or shell commands
- 2 System calls (functions provided by the kernel)
- 3 Library calls (functions within system libraries)
- 4 Special files (usually found in /dev)
- 5 File formats and conventions eg /etc/passwd
- 6 Games
- 7 Macro packages and conventions eg `man(7)`, `groff(7)`.
- 8 System administration commands (usually only for root)
- 9 Kernel routines [Non standard]

# Autres documentations en lignes (1)

- ☞ documentation hypertexte des outils GNU
  - ➡ commande `info`
- ☞ documentations d'installation
  - ➡ répertoires de `/usr/share/doc` ou `/usr/doc`
- ☞ les fichiers HOWTO
  - ➡ <http://www.linuxdoc.org>
- ☞ les Frequently Asked Questions (FAQ)
  - ➡ <http://www.faqs.org>
- ☞ les fichiers de références (RFC, etc.)
  - ➡ <http://www.rfc-editor.org>

# Autres documentations en lignes (2)

## ☞ les sites webs dédiés à un système

- \* Systèmes SUN ..... <http://docs.sun.com>
- \* Systèmes Microsoft ..... <http://support.microsoft.com>

## ☞ les répertoires web

- ➡ <http://www.wikipedia.org>
- ➡ <http://directory.google.com>

## ☞ les moteurs de recherche web génériques

- ➡ <http://www.google.com>

# Des gens qui savent

- ☞ les listes de diffusions mails et leurs archives
  - ➡ `http://www.mail-archive.com`, etc.
- ☞ le réseau USENET (news)
  - ➡ interface web à `http://groups.google.com`

En **dernier** recours :

- ☞ les collègues (souvent plus anciens)
- ☞ les anciens binômes
- ☞ les anciens enseignants (souvent très occupés :-)

Partie B

# Gestion du réseau



IUT A

Cours n° B.1

# Accès à distance

# Accès à distance

Généralement, on ne configure pas un serveur en travaillant physiquement sur celui-ci

- Salle serveur climatisée, bruyante
- Data center délocalisé
- On travaille souvent avec plusieurs serveurs (IUT: une dizaine de serveurs)



# Ancienne solution, telnet

Telnet est un protocole de connexion distante à une machine qui utilise le port TCP 23

## Connexion telnet

```
bash:~$ telnet kwak.lifl.fr
Trying 132.206.11.14...
Connected to kwak.lifl.fr.
Escape character is '^]'.
Debian GNU/Linux lenny/sid
kwak login:
```

# Solution actuelle, SSH

## Telnet

- Communication non chiffrée
- Login/mot de passe peuvent être interceptés !

## SSH

- Connexion chiffrée
- Authentification de l'hôte via certificat
- SSH peut également servir de tunnel sécurisé
- Utilise le port 22

# Écoute d'un flux telnet

Wireshark interface showing a network capture. The main packet list displays a TCP Reset (RST) from 127.0.0.1 to 127.0.0.1 on port 23. The packet details pane shows the 'Transmission Control Protocol' section with 'Src Port: 35288' and 'Dst Port: telnet (23)'. The packet bytes pane shows the raw data of the RST segment.

A 'Follow TCP Stream' dialog is open, displaying the stream content in ASCII. The stream content shows a login attempt for 'hhaauussppiiee' with password 'tarata'.

Stream Content:

```
(.....38400,38400....#.kwak.lifl.fr:0....'.DISPLAY.kwak.lifl.fr:0.....xterm.....Debian GNU/Linux lenny/sid
kwak login: hhaauussppiiee
Password: tarata
```

# Écoute d'un flux SSH

Follow TCP Stream

Stream Content

```

.#...?4K..b.....l.../.;.....YJeQ.....ssh-rsa....m..yU..JLq&.f.q...X.Id... .Y*...(.u...
$.C.J#..[N.$Z...\ /7@.<. ....k<.?7.Z.F,....C.:&.0V>...b.....+
-.....zN.E.Nn.
..L.<.n.....@)u.S...#`D~.....rM.N.....G.} .59.....H...
z~..@...[.....8{..Q....}$Mf4.6.F.[D.PB.$...J..
Fp...S.
1.A.....
.....
.....7.E....4.W$Z..C?3.a6SGYW.v^..H..L...o.u./..
.D~{...b0...+J:ZYt...Z....@W.....'..$.){.\.rJ...X.....U.....a..) ... ^..v...ZIf.7:.nT...H....;
.lR..d.)
U...s.N^2..]C.D4.4"._..@.*.....z...[...zk.{..cE)S.....6.....9.a.J..eC.....H{.....K.B!
u...m.....fj@..M'...J.n...K...F
k1.S...E.t...n.....x.....8.k...>D|=..c@+...=.Xr...%i.z@.....,j...0..9.u.....]sh.8..!....A
(....6t.....A" .U.h.H...5=P....xK..H.e..nQ...a.?P...' .C....f#v.....
%..... .S...) .Z.7...b....] .E.....8.k...={./..t.?.....^.".....E
$K..TAV.....PHD..#j.ZQ.db.....q...f....|.j
z_...=.....M...n.B.u~....C9#.5..QePH.i..(k.;!...lq>y.....%...nM.cf
.....2. ..@.#'..|

```

Find Save As Print Entire conversation (3270 bytes) ASCII EBCDIC Hex Dump C Arrays Raw

Help Close Filter Out This Stream

# Authentification sans mot de passe

SSH permet une authentification par paire de clés

- Le mot de passe ne transite plus par le réseau du tout (même pas en chiffré)
- L'utilisateur possède une paire de clés
  - ▶ Clé publique disponible pour tous
  - ▶ Clé privée que seul l'utilisateur possède
  - ▶ Un mot chiffré avec la clé publique ne peut être déchiffré que grâce à la clé privée
  - ▶ Un mot chiffré avec la clé privée ne pourra être déchiffré que grâce à la clé publique
  - ▶ L'utilisateur donne sa clé publique au serveur une fois pour toutes
  - ▶ Lors de l'authentification, le serveur peut tester grâce à la clé publique si l'utilisateur possède bien la clé privée associée

# OpenSSH

## Commandes

- `ssh [utilisateur]@machine` → connexion,
- `ssh-keygen [-t <rsa|dsa>]` → génération des clés,
- `ssh-copy-id [utilisateur]@machine` → Copie de la clé publique.

## Configuration de l'utilisateur

- Fichiers du répertoire `$HOME/.ssh`
- `id_rsa` : clé privé
- `id_rsa.pub` : clé publique associée
- `known_hosts` : liste des certificats des machines sur lesquelles l'utilisateur s'est déjà connecté
- `authorized_keys` : liste des clés publiques dont la clé privées associée est autorisée à se connecter sans mot de passe

# OpenSSH - Configuration de la machine

- `/etc/ssh/ssh_config` : configuration du client ssh
- `/etc/ssh/sshd_config` : configuration du serveur ssh
- `/etc/ssh/ssh_host_(dsa|rsa)_key` : clé privé du serveur (rsa ou dsa)
- `/etc/ssh/ssh_host_(dsa|rsa)_key.pub` : clé publique du serveur

# OpenSSH - Quelques paramètres du serveur

- Port 22 : port d'écoute (22 par défaut)
- PermitRootLogin yes : autorise (ou non) le root à se connecter à distance via SSH
- PasswordAuthentication yes : autorise (ou non) les utilisateurs à se connecter à l'aide d'un mot de passe. Si non, l'utilisation de paire de clés est obligatoire
- ...



## SCP

- copie de fichiers par tunnel sécurisé
- `scp fichier_source fichier_destination`
- Les fichiers sont donnés sous la forme : `[[login]@machine:]chemin`

## Exemple

- `scp hauspie@kwak.lifl.fr:/home/hauspie/toto .`
- `scp titi hauspie@kwak.lifl.fr:/home/hauspie/`

## Tunnels

- option `-L` de `ssh`
- `ssh hauspie@demon.lifl.fr -L 8080:bruyere.lifl.fr:443`
  - ▶ Ouvre un tunnel du port local 8080 au port 443 de la machine `bruyere.lifl.fr` en passant par `demon.lifl.fr`
  - ▶ Se connecter sur la machine locale sur le port 8080 utilisera le tunnel et permettra une connexion à `bruyere.lifl.fr` sur le port 443 sécurisée entre la machine locale et la machine `demon.lifl.fr`

Cours n° B.2

# DNS

# Domain Name System

## Problématique

- Pour communiquer avec une machine, il faut connaître son adresse IP  
⇒ comment retenir plusieurs centaines de numéros IP ?
- Il faut un mécanisme qui associe un nom plus naturel à chaque machine
- Solution ⇒ Domain Name System (DNS)

# Fonction

- Le protocole DNS permet d'associer un nom à une adresse IP et inversement
  - ▶ Ex: `www.univ-lille1.fr`  $\Leftrightarrow$  `134.206.1.13`
- Le nom complet d'une machine est décomposé en trois parties :
  - ▶ Son nom proprement dit:
    - ★ `www.univ-lille1.fr`
    - ★ `www.framasoft.org`
  - ▶ Le domaine (ou zone) auquel la machine appartient :
    - ★ `www.univ-lille1.fr`
    - ★ `www.framasoft.org`
  - ▶ Le *TLD* ou Top Level Domain
    - ★ `www.univ-lille1.fr`
    - ★ `www.framasoft.org`

# Organisation

- L'organisation de l'espace de nommage est hiérarchique
  - ▶ Chaque domaine est géré par l'administrateur qui le possède
  - ▶ Pour pouvoir résoudre un nom, il faut connaître l'adresse du serveur qui gère le domaine
- DNS constitue une base de données d'enregistrements
- Chaque enregistrement a un type
  - ▶ A: une adresse IPv4
  - ▶ AAAA: une adresse IPv6
  - ▶ NS: Nom des serveurs faisant autorité
  - ▶ ...

# Serveurs racines

```
$ dig NS .  
[...]  
;; ANSWER SECTION:  
.      518157  IN  NS  a.root-servers.net.  
.      518157  IN  NS  b.root-servers.net.  
.      518157  IN  NS  c.root-servers.net.  
.      518157  IN  NS  d.root-servers.net.  
[...]  
;; ADDITIONAL SECTION:  
a.root-servers.net. 3599845  IN  A  198.41.0.4  
b.root-servers.net. 3599757  IN  A  192.228.79.201  
c.root-servers.net. 3599757  IN  A  192.33.4.12  
d.root-servers.net. 3599757  IN  A  199.7.91.13  
[...]
```

## Étape 1. TLD

Une requête DNS commence par l'interrogation des serveurs racines<sup>1</sup> afin de connaître le gestionnaire du *TLD* (Top Level Domain)

```
$ dig NS fr @198.41.0.4
```

```
; AUTHORITY SECTION:
```

```
fr.          172800  IN   NS    e.ext.nic.fr.
```

```
fr.          172800  IN   NS    f.ext.nic.fr.
```

```
fr.          172800  IN   NS    g.ext.nic.fr.
```

```
fr.          172800  IN   NS    d.nic.fr.
```

```
fr.          172800  IN   NS    d.ext.nic.fr.
```

```
;; ADDITIONAL SECTION:
```

```
e.ext.nic.fr. 172800  IN   A      193.176.144.22
```

```
e.ext.nic.fr. 172800  IN   AAAA   2a00:d78::102:193:176:144:2
```

```
f.ext.nic.fr. 172800  IN   A      194.146.106.46
```

```
f.ext.nic.fr. 172800  IN   AAAA   2001:67c:1010:11::53
```

```
g.ext.nic.fr. 172800  IN   A      194.0.36.1
```

```
g.ext.nic.fr. 172800  IN   AAAA   2001:678:4c::1
```



## Étape 2. Domaine

```
$ dig NS univ-lille1.fr @193.176.144.22
;; AUTHORITY SECTION:
univ-lille1.fr.      172800  IN   NS   reserv2.univ-lille1.fr.
univ-lille1.fr.      172800  IN   NS   reserv1.univ-lille1.fr.

;; ADDITIONAL SECTION:
reserv1.univ-lille1.fr. 172800  IN   A   193.49.225.15
reserv2.univ-lille1.fr. 172800  IN   A   193.49.225.90
```

## Étape 3. Requête finale

```
$ dig A www.univ-lille1.fr @193.49.225.15
```

```
;; ANSWER SECTION:
```

```
www.univ-lille1.fr.          21600 IN  CNAME  front-ig-slb.univ-lille1.fr.  
front-ig-slb.univ-lille1.fr. 21600 IN  A      193.49.225.174
```

# Détails techniques

- Deux types de serveurs
  - ▶ Les serveurs faisant autorité : possèdent les données d'une zone
  - ▶ Les serveurs résolveurs (ou récursifs) : utilisés par les utilisateurs finaux pour effectuer automatiquement les étapes précédentes
- Le protocole DNS utilise
  - ▶ Le port UDP 53 : utilisé pour les requêtes
  - ▶ Le port TCP 53 : utilisé pour les requêtes ou les transferts de zone
- il est décrit par les RFC
  - ▶ 1034
  - ▶ 1035
  - ▶ ...

# Implémentation

- L'implémentation la plus utilisée du protocole DNS est le serveur **bind9**
- Mais il en existe d'autres
  - ▶ DJBDNS
  - ▶ PowerDNS
  - ▶ ...
- Attention à bien faire la différence entre un **protocole** et **l'implémentation** de ce protocole

# Configuration

- Le server DNS bind se configure à l'aide de plusieurs fichiers
- `/etc/bind/named.conf`
  - ▶ Fichier de configuration principal
    - ★ Permet de définir les options globales
    - ★ Permet de définir les zones (nom, fichier de zone, options spécifiques, ...)
    - ★ Permet de définir les droits d'accès au serveur
  - ▶ Fichiers de base de données de zone
    - ★ Correspondance IP  $\Leftrightarrow$  nom de machine ou nom de machine  $\Leftrightarrow$  IP
    - ★ Paramètres de la zone (nom du serveur de nom, nom des serveurs de mail,...)

# Fichier named.conf

- Succession de déclaration de la forme

```
<déclaration> ["<nom-declaration>"] [<classe-declaration>]
{
    <option-1>;
    <option-2>;
    ...
    <option-N>;
};
```

- Possibilité d'utiliser des commentaires (//, /\* \*/, #)

# Fichier named.conf

## Déclarations courantes

- options permet de définir la configuration globale du serveur

```
options {  
    <option>;  
    [<option>; ...]  
};
```

- Nombre important d'options possibles  $\Rightarrow$  lire la documentation
- Les options les plus utilisées :
  - ▶ allow-query : autorise l'interrogation du serveur pour les zones gérées par celui-ci
  - ▶ allow-recursion : idem mais pour toutes les zones
  - ▶ blackhole : interdit l'accès du serveur pour un ensemble d'hôtes
  - ▶ directory : précise le répertoire dans lequel le serveur peut trouver les fichiers donnés par un chemin relatif

# Fichier named.conf

## Déclaration de zone

- La déclaration zone définit le comportement du serveur vis-à-vis d'une zone

```
zone <zone-name> <zone-class> {  
    <zone-options>;  
    [<zone-options>; ...]  
};
```

- Deux types de zones
  - ▶ Translation nom  $\Rightarrow$  IP
  - ▶ Translation IP  $\Rightarrow$  nom
- En plus des options de zone, on peut affiner certaines options globales telles que
  - ▶ allow-query
  - ▶ ...



## Exemple

```
// Nom -> IP
zone "toto.org" {
    type master; // serveur maître
    file "toto.org"; // fichier de description de zone
};

// IP -> nom (reverse)
// réseau à l'envers suivie de in-addr.arpa
zone "139.168.192.in-addr.arpa" {
    type master;
    file "toto.org.rev"; // fichier de description de zone
};

// Serveur secondaire
Zone "tata.org" {
    type slave; // serveur secondaire
    file "tata.org";
    masters { 134.206.10.18; }; // Adresse IP du serveur p
};
```

# named.conf

## Déclarations courantes

- Déclaration `acl`

```
acl <acl-name> {  
    <match-element>;  
    [<match-element>; ...]  
};
```

- Permet de définir une liste d'hôte qui pourra être utilisée comme alias dans les définitions de restriction d'accès
- Les `<match-element>` sont des adresses IP désignant une machine ou un réseau
- On peut aussi utiliser des alias prédéfinis
  - ▶ `any` : toutes les adresses IP
  - ▶ `localhost` : uniquement l'adresse IP du système local
  - ▶ `localnets` : uniquement le réseau local du système
  - ▶ `none` : aucune IP

## Exemple d'acl

```
acl ustl {  
    134.206.0.0/16;  
};  
acl machine-locale {  
    127.0.0.1;  
};  
acl reseau-local {  
    192.168.139.0/24;  
};  
options {  
    allow-query {  
        ustl;  
        reseau-local;  
        machine-locale; };  
    allow-recursion { machine-locale; };  
};
```

# Fichier de zone

- La syntaxe du fichier est totalement différentes
- Deux parties :
  - ▶ Les directives
    - ★ `$INCLUDE` permet d'inclure un autre fichier
    - ★ `$ORIGIN` redéfinit l'origine courante. Elle sera utilisée pour compléter les noms qui ne sont pas complètement qualifiés (qui ne se terminent pas par `.`).  
ex: `$ORIGIN toto.org.`  
L'origine courante peut être explicitement référencée par le caractère `@`
    - ★ `$TTL` donne la valeur par défaut pour la durée de vie des informations de la zone.  
ex: `$TTL 3600`
  - ▶ Les enregistrements. Toutes les informations que peut donner un serveur sur une zone
    - ★ translation nom  $\Rightarrow$  IP
    - ★ translation IP  $\Rightarrow$  nom
    - ★ nom du serveur mail
    - ★ nom du serveur DNS
    - ★ ...
- Les commentaires sont faits à l'aide du caractère ;

# Format des enregistrements

- A : translation nom  $\Rightarrow$  IP

- ▶ <host>      IN      A      <IP>
- ▶ toto          IN      A      192.168.139.128

- CNAME : alias (donner plusieurs noms à une seule IP)

- ▶ <alias>      IN      CNAME      <nom-reel>
- ▶ www          IN      CNAME      toto.toto.org.

- MX : nom du serveur recevant les mails @domaine

- ▶ IN      MX      <priorité>      <nom>
- ▶ IN      MX      10                      mail.toto.org

- NS : nom des serveurs faisant autorité pour la zone

- ▶ IN      NS      <nom>
- ▶ IN      NS      ns1.toto.org.
- ▶ IN      NS      ns2.toto.org.

À chaque enregistrement NS **doit** correspondre un enregistrement A !

- PTR : translation IP  $\Rightarrow$  nom (dans les zones inverse)

- ▶ <derniers-digits-ip>      IN      PTR      <nom>
- ▶ 128                              IN      PTR      toto.toto.org.

# Enregistrements

- SOA : Start Of Authority. Donne les informations sur la zone

```
@    IN      SOA  <ns-primaire> <mail-admin-domain> (  
    <numero-de-serie>  
    <time-to-refresh>  
    <time-to-retry>  
    <time-to-expire>  
    <minimum-ttl>)
```

- <ns-primaire> : nom du serveur de nom maître pour la zone
- <numero-de-serie> : numéro de série du fichier de zone. DOIT être modifié à chaque modification du fichier
- <time-to-refresh> : Temps qu'un serveur esclave peut attendre avant de rafraîchir les informations sur la zone
- <time-to-retry> : Temps qu'un serveur esclave doit attendre avant de rafraîchir à nouveau la zone si un rafraîchissement précédent a échoué
- <time-to-expire> : Temps au bout duquel le serveur esclave se considère comme maître si le maître n'a pas répondu
- <minimum-ttl> : temps minimum de mise en cache des informations pour les autres serveurs de noms
- Toutes les durées sont exprimées en secondes mais on peut utiliser des abréviations pour les unités de temps (M, H, D...)
- Il ne peut en exister qu'un par zone

# Exemple

```
$TTL 604800
; Serveur de nom principal: ns.toto.org
; Mail de l'administrateur root@toto.org
@      IN      SOA      ns.toto.org. root.toto.org. (
                                1          ; Serial
                                1D         ; Refresh
                                2H         ; Retry
                                7D         ; Expire
                                1H )       ; Cache TTL

      IN  NS      ns.toto.org.

ns     IN  A       192.168.139.128
tata   IN  CNAME   ns.toto.org.
```

Cours n° B.3

# DHCP



## Attribution de paramètres IP

- Cas d'une gestion de parc informatique
  - ▶ Entrer les paramètres IP à la main pour chaque machine est une tâche lourde pour l'administrateur (ex: IUT ~330 machines)
- Cas d'un utilisateur nomade ou néophyte
  - ▶ Retenir tous les paramètres (IP, masque, DNS, routeur...) des réseaux que l'on utilise (maison, bureau)
  - ▶ Manipulation à chaque démarrage
  - ▶ Difficile pour le non informaticien

## DHCP

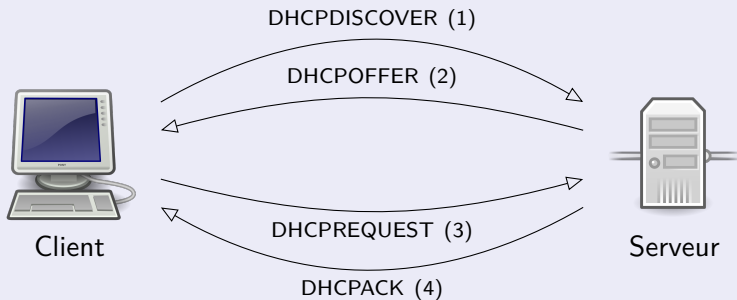
- Dynamic Host Configuration Protocol (RFC 2131)
- Permet de donner automatiquement les paramètres réseau à une machine au démarrage du système
  - ▶ Ne nécessite pas d'intervention de l'utilisateur
  - ▶ Ne nécessite pas de connaissance pour avoir accès au réseau

Le DHCP apporte aussi d'autres fonctionnalités

- Mise à jour automatique du DNS en fonction du nom de machine envoyé par le client
- Envoi d'une image de boot au client
  - ▶ Démarrage d'un système sans support physique (installation de système, poste client sans disque-dur...)

# Détails techniques

## Fonctionnement



À la réception du paquet DHCPACK, le client obtient la confirmation de son adresse IP ainsi que ses paramètres réseau

- Masque
- Adresse du routeur
- Adresse du serveur DNS
- ...

# DHCP

- Implémentation sous linux par `isc-dhcp-server`
- Un seul fichier de configuration
- Permet une mise à jour automatique du serveur DNS
- Permet de faire démarrer une machine en lui envoyant du code

- Liste de directives de configurations de deux types:
  - ▶ Les paramètres
  - ▶ Les déclarations
- Les paramètres précisent
  - ▶ comment faire quelque chose
  - ▶ s'il faut le faire
  - ▶ les information à fournir aux clients
- Les déclarations
  - ▶ décrivent la topologie du réseau
  - ▶ décrivent les clients
  - ▶ donnent les adresses pouvant être fournies au clients
  - ▶ ...

# Principaux paramètres

- `[not] authoritative`
  - ▶ Précise si le serveur fait autorité sur le réseau
- `ddns-update-style <interim|none>`
  - ▶ active/désactive la mise à jour automatique d'un serveur DNS

# Principales déclarations

## Définition d'un réseau

```
subnet <IP du réseau> netmask <masque> {  
    [paramètres spécifiques]  
    [déclarations spécifiques]  
}
```

## Définition d'un hôte

```
host <hostname> {  
    [paramètres spécifiques à un hôte]  
}
```

- ☞ permet d'affecter une IP fixe pour une même machine en fonction de son adresse MAC



## Exemple de déclaration d'un réseau

```
subnet 192.168.119.0 netmask 255.255.255.0 {  
    range 192.168.119.10 192.168.119.20;  
    option routers 192.168.119.2;  
    option subnet-mask 255.255.255.0;  
    option domain-name "licence-rt.org";  
    option domain-name-servers 192.168.119.131;  
  
    host "lprt" {  
        hardware ethernet 00:50:F2:5F:E6:0E;  
        fixed-address lprt.licence-rt.org;  
    }  
}
```

# Interaction entre DHCP3 et bind9

- Permet de contacter une machine à l'aide de son nom, même si son adresse IP change
- Se base sur une communication entre le serveur DHCP et le service DNS via un port de contrôle.
- L'outil RNDP permet également de piloter le serveur DNS.

## RNDP

- Outil d'administration d'un serveur bind9 en cours de fonctionnement
- Configuré grâce au fichier `/etc/bind/rndp.conf`.
  - ▶ définit une clé privée partagée entre le serveur DNS et RNDP
  - ▶ définit l'adresse du serveur DNS à piloter
- Si ce fichier est absent, rndp contacte le serveur à l'adresse `localhost` en utilisant la clé fournie dans le fichier `/etc/bind/rndp.key`.
- la clé doit être indiquée dans le fichier `named.conf`.

# Fichier rndc.conf

## Exemple

```
key "rndc-key" {  
    algorithm hmac-md5;  
    secret "rl5PKn38zEkMpL08A6KxDg==";  
};  
  
options {  
    default-key "rndc-key";  
    default-server 127.0.0.1;  
    default-port 953;  
};
```

- rndc-confgen peut être utilisé pour générer le fichier
- La fin du fichier généré par rndc-confgen contient les modifications à apporter au fichier named.conf

# Modifications du fichier named.conf

## Exemple

```
key "rndc-key" {
    algorithm hmac-md5;
    secret "rl5PKn38zEkMpL08A6KxDg==";
};

key "dhcp-update-key" {
    algorithm hmac-md5;
    secret "8uurKn63JdnUHUkakjUUA==";
};};

controls {
    inet 127.0.0.1 port 953
    allow { 127.0.0.1; } keys { "rndc-key"; };

    inet 192.168.119.131 port 953
    allow {192.168.119.111; } keys {"dhcp-update-key";};
};
```

# Commande rndc

Permet d'administrer le serveur DNS

- `rndc reconfig` : recharge le fichier `named.conf` (mais pas les zones)
- `rndc reload` : recharge le fichier `named.conf` **ET** les zones
- `rndc freeze <zone>` : gèle la mise à jour automatique d'une zone. Indispensable avant d'éditer un fichier décrivant une zone pouvant être mise à jour dynamiquement
- `rndc thaw <zone>` : Recharge la zone depuis le fichier de zone et réactive la mise à jour dynamique.

## Attention !

Une fois `rndc` configuré, le script `/etc/init.d/bind9` utilisera `rndc` pour stopper le serveur. Si la configuration est mal faite, il faudra tuer le processus du serveur à *la main* (`killall named`) avant de pouvoir le relancer pour recharger la configuration corrigée.

# Mise à jour de zone via DHCP

Si le serveur DNS est correctement configuré, on peut configurer DHCP pour qu'il effectue une mise à jour automatique de zone

## Les paramètres

- `ddns-update-style interim;`
- définir la clé privée (même syntaxe que pour `rndc`  
`key <nom> {...};`)
- définir les zones que DHCP devra mettre à jour

# Modification du fichier `dhcpcd.conf`

Ici, l'identifiant doit être le même que celui utilisé dans la configuration du serveur DNS.

```
key <identifiant> {  
    algorithm hmac-md5;  
    secret "secret commun au serveur";  
}  
  
zone <zone dns> {  
    primary <ip du serveur dns>;  
    key <identifiant de la clé>;  
}
```

# Boot par réseau

- Permet de facilement installer des postes sans support physique (CD, clé USB...)
- Permet de fabriquer des terminaux sans disque-dur
- Se base sur deux mécanismes :
  - ▶ DHCP pour donner les informations sur le serveur TFTP et sur le nom du fichier image à charger
  - ▶ TFTP (Tiny FTP) pour transférer l'image
- Nécessite une carte réseau prenant en compte la norme PXE



# TFTP

Service de transfert de fichiers minimaliste

- Pas d'authentification, accès public
- implémenté par `tftpd-hpa`, `tfptd`, ...

# Configuration du DHCP

Deux options à ajouter dans la définition du réseau

- `filename "fichier";` : définit le nom du fichier image à télécharger
- `next-server <ip-serveur-tftp>;` : donne l'ip du serveur TFTP à contacter pour télécharger le fichier image

# DHCP - Exemple complet 1/2

```
# Paramètres du serveur
server-identifier 192.168.119.111;
ddns-update-style interim;
authoritative;
# Paramètres de mise à jour DNS
key "dhcp-update-key" {
    algorithm hmac-md5;
    secret "8uurKn63JdnUHUkakjUUA==" ;
};
zone licence-rt.org. {
    primary 192.168.119.131;
    key "dhcp-update-key";
}
zone 100.168.192.in-addr.arpa. {
    primary 192.168.119.131;
    key "dhcp-update-key";
}
```

## DHCP - Exemple complet 2/2

# Définition du réseau

```
subnet 192.168.119.0 netmask 255.255.255.0 {  
    # Configuration du réseau  
    option routers 192.168.119.2;  
    option subnet-mask 255.255.255.0;  
    option domain-name "licence-rt.org";  
    option domain-name-servers 192.168.119.131;  
  
    # Plage d'attribution des Ips  
    range dynamic-bootp 192.168.119.10 192.168.119.30;  
  
    # Paramètres de boot PXE  
    filename "pxelinux.0";  
    next-server 192.168.119.112;  
}
```

## Partie C

# Gestion d'utilisateurs



IUT A

Cours n° C.1

# Annuaire, LDAP

# Les annuaires

D'un point de vue informatique, un annuaire est un moyen de stocker et consulter de l'information

- ☞ Coordonnées
- ☞ Identifiants (login/mot de passe)
- ☞ ...

# Annuaire et bases de données

Les annuaires sont différents des bases de données

- ➡ Les opérations de lecture sont beaucoup plus fréquentes avec un annuaire
- ➡ Les informations sont souvent diffusées plus largement



# Annuaire

- ☞ Retenir les informations sur les employés d'une société
- ☞ Et les partager avec ces même employés
- ☞ Applicables à toutes les ressources
- ☞ Exemple d'annuaire : DNS

En particulier, dans le cas d'un réseau

- ☞ Stocker les identifiants et les paramètres de connexions des utilisateurs
- ☞ Leur UID, GID
- ☞ Leur chemin de leur répertoire home
- ☞ Centraliser ces données pour avoir les mêmes sur toutes les machines du parc

# Mauvaise utilisation des annuaires

Les annuaires ne sont pas destinés à :

- ➡ Effectuer des modification très fréquentes des données
- ➡ Manipuler des données volumineuses

# Annuaire électronique, exemples

- ☞ Bases d'utilisateurs Unix (/etc/passwd)
- ☞ Annuaire d'applications
  - ☞ Carnet d'adresse client mail
- ☞ Annuaire pour la gestion de réseaux
  - ☞ NIS/NIS+
  - ☞ Microsoft Active Directory
- ☞ Annuaire génériques
  - ☞ X.500
  - ☞ LDAP
  - ☞ ...

# Historique

Dérivé de X.500 :

- ☞ Standard conçu pour interconnecter des annuaires téléphoniques
- ☞ X.500 définit les règles de nommages des objets, le protocole d'accès aux données,...

## Avantages

- ☞ Bon passage à l'échelle
- ☞ Permet d'effectuer des recherches évoluées
- ☞ Possibilité de distribuer l'information et l'administration

## Inconvénient

- ☞ Implémentations lourdes
- ☞ Peu interopérables

- ☞ Créé à partir du nettoyage de X500
- ☞ LDAPv1 : RFC 1487
- ☞ LDAPv2 : RFC 1777
- ☞ LDAPv3 : RFC 2251
- ☞ LDAP est une standardisation de l'accès à une base de données d'information → ne définit pas la façon de stocker les bases

## LDAP définit

- ☞ Le protocole d'accès/de mise à jour de l'information
- ☞ Les types des informations disponibles
- ☞ La façon dont elle sont organisées/référencées

# Opérations de base

- ☞ interrogation : search, compare
- ☞ mise à jour : add, delete, modify, rename
- ☞ connexion : bind, unbind, abandon



Le type des données que l'annuaire peut stocker est défini par les schémas

- ☞ Un schéma définit un ensemble de types (de classe) d'objets que connaît le serveur
- ☞ La classe d'un objet définit le nom et les propriétés des attributs
- ☞ Un objet possède des attributs obligatoires ou facultatifs
- ☞ L'héritage de classe d'objets est possible
- ☞ Les schéma standards peuvent être consultés sur <http://oav.net/mirrors/LDAP-ObjectClasses.html> <sup>2</sup>

# Attributs

Un attribut est défini par :

- ☞ Un nom
- ☞ Un Objet IDentifier (OID, numéro unique au monde qui identifie l'attribut)
- ☞ S'il peut posséder une ou plusieurs valeurs (par exemple, on peut avoir plusieurs numéro de téléphone)
- ☞ Comment on peut comparer les valeurs de cet attribut

Deux catégories d'attributs :

- ☞ Attributs normaux, manipulés par les utilisateurs (`givenname`, `telephoneNumber`...)
- ☞ Attributs systèmes, manipulés par le serveur (`modifiersname`...)

# Classe d'objets

Une classe d'objet permet de décrire une entité (une personne par exemple) par une liste d'attribut. Elle est définie par

- ☞ un nom
- ☞ un OID
- ☞ des attributs obligatoires
- ☞ des attributs optionnels
- ☞ un type (structurelle, auxiliaire ou abstraite)

# Exemple de classe

- Une organisation (o)
- Ses départements (ou, `organizationalUnit`)
- Ses employés (`organizationalPerson`)
- ...

# Type de classes

- ☞ Classe structurelle : description d'objets de l'annuaire (personnes, organisation...).
- ☞ Classe auxiliaire : permet de rajouter des informations à des objets structurels
- ☞ Classe abstraite : objets basiques du schéma (top, alias)

# Hiérarchie des classes

Les classes forment une hiérarchie

- ☞ Le sommet est l'objet `top`
- ☞ Chaque classe hérite des attributs dont elle est fille
- ☞ L'attribut `objectClass` permet de préciser la classe d'un objet

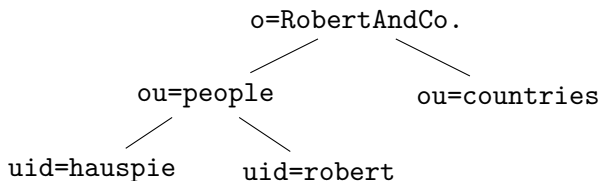
- `top`
  - ▶ `person`
    - ★ `organizationalPerson`
  - ▶ `organizationalUnit`

# Nommage des entrées

- ☞ Définit comment les objets sont nommés et comment ils sont organisés
- ☞ L'organisation des objets est faite suivant une structure hiérarchiques  
⇒ Directory Information Tree (DIT)
- ☞ L'identification d'un objet se fait par un nom, le Distinguished Name (dn)

Au sommet de l'arbre, on trouve le suffixe de la base LDAP (BaseDN)

- ☞ Le suffixe définit l'espace de nommage géré par le serveur
- ☞ Le Distinguished Name d'un objet le référence de manière unique
- ☞ Constitué de la suite des noms des entrées, en commençant par l'entrée et en remontant vers le suffixe, séparé par des ' , '.



Le dn de hauspie est `uid=hauspie,ou=people,o=RobertAndCo.`



# Recherche

Pour obtenir une information, il faut effectuer une opération de recherche.  
Une recherche comporte 8 paramètres

- ➡ `base object` : Où on commence la recherche
- ➡ `scope` : La profondeur de la recherche
- ➡ `derefAliases` : suit-on les liens inter-serveurs ?
- ➡ `size limit` : nombre maximum de réponses
- ➡ `time limit` : temps maximum alloué pour la recherche
- ➡ `attrOnly` : Précise si l'on veut uniquement le type des attributs ou également leur valeur
- ➡ `search filter` : le filtre de recherche
- ➡ `list of attributes` : les attributs que l'on veut connaître

# La profondeur de recherche

- ☞ `search scope = base` : uniquement l'entité d'où commence la recherche
- ☞ `search scope = onelevel` : recherche parmi les *frères* de l'entité où l'on commence la recherche (même profondeur de l'arbre)
- ☞ `search scope = subtree` : on recherche dans le sous arbre donc l'entité de début de recherche est la racine

# Filtre de recherche

(opérateur(recherche1)(recherche2)...) )

(cn=Toto)	égalité
(cn=*to*)	sous-chaîne
(cn~=dupond)	approximation
(employeeNumber>=10)	relation d'ordre
(sn=*)	existence
(&(sn=hauspie)(l=lille))	ET
( (sn=hauspie)(sn=robert))	OU
(!(tel=*))	NON

# LDAP Data Interchange Format (LDIF)

Pour décrire les données contenues dans un annuaire LDAP, on utilise le format de description LDIF

Forme générale :

```
dn: <distinguished name>
objectClass: <classe d'objet>
objectClass: <classe d'objet>
[...]
attribute type:<valeur>
attribute type:<valeur>
[...]
```

# LDIF

## Exemple

```
dn: cn=Michael Hauspie,ou=2XS,dc=univ-lille1,dc=fr
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
cn: Michael Hauspie
sn: hauspie
givenName: Michael
mail: Michael.Hauspie@univ-lille1.fr
userPassword: {SSHA}t59IhKr5WSHtM1jPi/33XuQWzwoWUHQ
```

# LDIF – modification d'entrée

Forme générale :

```
dn: distinguished name
changetype: identifiant
opérateur de modification
liste d'attributs
-
opérateur de modification
liste d'attributs
```

Les opérations possible sont :

- ✎ `changetype: add` : créer une nouvelle entrée
- ✎ `changetype: delete` : supprimer une entrée
- ✎ `changetype: modifyrdn` : renommer une entrée
- ✎ `changetype: modify` : modifier une entrée

## LDIF – Modification d'attributs

Dans le cas de l'opération `modify`, il faut préciser un opérateur de modification parmi

- ➡ `add` : ajoute un attribut
- ➡ `replace` : modifie un attribut
- ➡ `delete` : supprime un attribut

Si on veut effectuer plusieurs modification sur les attributs, on les sépare par `' - '`

Exemple, ajouter un numéro de téléphone et une adresse mail

```
dn: cn=Michael Hauspie,ou=2XS,dc=univ-lille1,dc=fr
changetype: modify
add: telephonenumber
telephonenumber: +33 (0)3.59.63.22.33
-
add: mail
mail: Michael.Hauspie@lifl.fr
```

# Conception du modèle de nommage

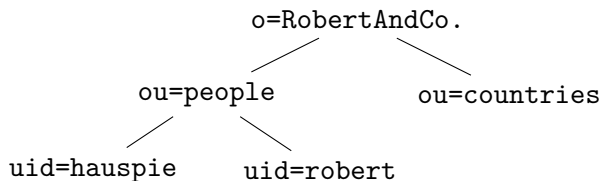
Le modèle de nommage définit la façon dont les données sont organisées

- ☞ Influe sur la performance des recherches et sur les facilités d'administration des données
- ☞ Arbre profond : les entités sont fortement classées (par organisation, département etc...)
- ☞ Arbre peu profond : toutes les entités au même niveau



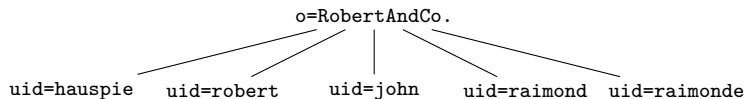
# Conception du modèle de nommage

*Abre profond*



# Conception du modèle de nommage

*Arbre peu profond*



# Conception du modèle de nommage

Que choisir ?

	Arbre profond	Arbre peu profond
Les plus	<p>Reflète plus la réalité</p> <p>Plus simple d'avoir des DN uniques</p> <p>Facilite la répartition sur plusieurs serveurs</p> <p>Mise à jour plus rapide (dans le cas d'OpenLDAP)</p> <p>Recherche plus rapide si elle est effectuée plus précisément</p>	<p>Moins de travail de maintenance si l'organisation varie beaucoup</p> <p>DN courts</p>
Les moins	<p>DN long</p> <p>Problème si l'organisation change</p>	<p>Difficile d'avoir des DN uniques</p> <p>Répartition sur plusieurs serveurs difficile</p>

# Conception du modèle de nommage

## Choix du suffixe

Le suffixe est l'identifiant de la base

- ☞ Si possible, choisir un identifiant unique au monde
- ☞ l'IETF préconise l'utilisation du nom de domaine DNS comme suffixe

Deux façon de l'écrire :

- ☞ En utilisant une entrée organisation : `o=licence-rt.org`
- ☞ En utilisant une entrée domaine (Domain component) :  
`dc=licence-rt,dc=org`

# OpenLDAP

- Implémentation libre d'un serveur LDAP
- Sous debian, package slapd

# OpenLDAP

- 👉 La configuration du serveur OpenLDAP se fait maintenant grâce
- 👉 L'administration se fait donc **en ligne** à l'aide des outils:  
ldapsearch, ldapadd, ldapmodify...
- 👉 Ex: affichage de la configuration complète :  

```
# ldapsearch -Y EXTERNAL -H ldapi:/// -b cn=config
```
- 👉 `-Y EXTERNAL` → utilisation du mécanisme d'authentification SASL EXTERNAL (basée sur l'uid du processus). Si l'utilisateur qui lance la commande est root, il pourra accéder et modifier la configuration.
- 👉 `-H ldapi:///` → connexion via la socket unix locale.
- 👉 `-b cn=config` → suffixe de la base de configuration

# OpenLDAP: exemples de changement de configuration

## Changement du niveau de log

```
# cat > /tmp/loglevel.ldif <<EOF
dn: cn=config
changetype: modify
replace: olcLogLevel
olcLogLevel: 256
EOF

# ldapmodify -Y EXTERNAL -H ldapi:/// -f /tmp/loglevel.ldif
```

## Ajout de schéma

```
# ldapadd -c -Y EXTERNAL -H ldapi:/// -f monschema.ldif
# zcat /usr/share/doc/samba-doc/examples/LDAP/samba.ldif.gz |
    ldapadd -c -Y EXTERNAL -H ldapi:///
```

# OpenLDAP: exemples de changement de configuration

## Ajout d'un administrateur de configuration

```
# cat > /tmp/access.ldif << EOF
dn: olcDatabase={0}config,cn=config
changetype: modify
add: olcAccess
olcAccess: to * by dn="cn=admin,dc=licence-rt,dc=org" write
EOF

# ldapmodify -c -Y EXTERNAL -H ldapi:/// -f /tmp/access.ldif
```

## Ajout d'un index sur le champ uid

Les index sont des structures de données accélérant la recherche dans l'arbre

```
# cat > /tmp/uid_index.ldif <<EOF
dn: olcDatabase={1}mdb,cn=config
changetype: modify
add: olcDbIndex
olcDbIndex: uid eq
EOF

# ldapmodify -Y EXTERNAL -H ldapi:/// -f /tmp/uid_index.ldif
```



# Gestion hors ligne

- Outils de gestion hors ligne: `slapcat`, `slapadd`
- Test de la configuration configuration: `slaptest`. Peut également convertir une ancienne configuration au nouveau format (options `-f` et `-F`)
- Génération de l'index de recherche: `slapindex`
- Générer le hash d'un mot de passe: `slappasswd`
- Le serveur doit être **arrêté** pour que ces outils fonctionnent
- Il est préférable d'utiliser les outils en ligne avec  
`-Y EXTERNAL -H ldapi:///`

# Gestion en ligne

- `ldapsearch` : effectuer une recherche
- `ldapadd` : ajouter une entrée
- `ldapmodify` : modifier une entrée
- `ldapdelete` : supprimer une entrée
- `ldapmodrdn` : déplacer une entrée

# Création initiale de la base

## Création d'une nouvelle base

```
# cat > /tmp/newdb.ldif << EOF
dn: olcDatabase=mdb,cn=config
objectClass: olcDatabaseConfig
objectClass: olcMdbConfig
olcDatabase: mdb
olcDbDirectory: /srv/ldap/licence-rt
olcSuffix: dc=licence-rt,dc=org
olcRootDN: cn=admin,dc=licence-rt,dc=org
olcRootPW: {SSHA}RDMT4VAf8R7T46a3kyNiF1JmnTPploDd
EOF

# ldapadd -c -Y EXTERNAL -H ldapi:/// -f /tmp/newdb.ldif
```

# Création initiale de la base, suite

## Ajout de l'objet racine de la base

```
# cat > /tmp/newdbroot.ldif <<EOF
dn: dc=licence-rt,dc=org
objectClass: dcObject
objectClass: organization
dc: licence-rt
o: licence-rt.org
EOF
# ldapadd -Y EXTERNAL -H ldapi:/// -f /tmp/newdbroot.ldif
```

# Création initiale de la base

## Ajout de l'utilisateur administrateur

```
# slappasswd
New password:
Re-enter new password:
{SSHA}m+sPI3JWRW1v7Rrne2m7fcKJzLeBFn7B
# cat > /tmp/admin << EOF
dn: cn=admin,dc=licence-rt,dc=org
objectClass: simpleSecurityObject
objectClass: organizationalRole
cn: admin
description: Administrator
userPassword: {SSHA}m+sPI3JWRW1v7Rrne2m7fcKJzLeBFn7
EOF

# ldapadd -c -Y EXTERNAL -H ldapi:/// -f /tmp/admin.ldif
```

# Exemple de recherche

```
bash$ ldapsearch -x -h localhost -b dc=licence-rt,dc=org
# extended LDIF
#
# LDAPv3
# base <dc=licence-rt,dc=org> with scope subtree
# filter: (objectclass=*)
# requesting: ALL
#
# licence-rt.org
dn: dc=licence-rt,dc=org
objectClass: top
objectClass: dcObject
objectClass: organization
o: licence-rt.org
dc: licence-rt

# admin, licence-rt.org
dn: cn=admin,dc=licence-rt,dc=org
objectClass: simpleSecurityObject
objectClass: organizationalRole
cn: admin
description: Administrator

# search result
search: 2
result: 0 Success

# numResponses: 3
# numEntries: 2
bash$
```

# Analyse de la commande de recherche

- `-x` mécanisme d'authentification simple (non SASL)
- `-h localhost` adresse du serveur
- `-b dc=licence-rt,dc=org` début de la recherche → on cherche à partir de la racine de notre base

Par défaut, ici, l'interrogation de l'annuaire est **anonyme**. Certains attributs sont donc non visibles (mot de passe).

Pour y avoir accès, il faut s'authentifier avec un compte autorisé à voir ces attributs. Dans notre cas, le compte administrateur.

# Accès authentifié

L'accès authentifié passe par une liaison au serveur (**bind**). On va donner au serveur le dn d'un objet de l'annuaire et s'authentifier grâce à l'attribut **userPassword** de cet objet.

```
bash$ ldapsearch -x -h localhost -b dc=licence-rt,dc=org -D cn=admin,dc=licence-rt,dc=org \
-W description=Administrator
Enter LDAP Password:
# extended LDIF
#
# LDAPv3
# base <dc=licence-rt,dc=org> with scope subtree
# filter: description=Administrator
# requesting: ALL
#
# admin, licence-rt.org
dn: cn=admin,dc=licence-rt,dc=org
objectClass: simpleSecurityObject
objectClass: organizationalRole
cn: admin
description: Administrator
userPassword:: e1NTSEF9QmNZWXgxUGhZRGZrVXIvVOxOYmhWL3VSaytGb3Z1b3U=

# search result
search: 2
result: 0 Success

# numResponses: 2
# numEntries: 1
bash$
```



Cours n° C.2

# Authentification sous unix

# Authentification locale

Dans le cas d'un système local, l'authentification fait appel à deux fichiers :

- `/etc/passwd` : donne la liste des utilisateurs et les attributs de ces utilisateurs (uid, gid, répertoire, shell...)
- `/etc/shadow` : contient une version hachée du mot de passe des utilisateurs.

# Mécanisme de contrôle

Le processus d'authentification est contrôlé par deux mécanismes :

- Name Service Switch (nss) : permet de spécifier les sources des différentes base de données du système (uid, gid, host...)
- Pluggable Authentication Module (pam) : permet de gérer le processus d'authentification, d'accès et de configuration de la session d'un utilisateur.

# Name Service Switch

Pour stocker les informations du système tels que la liaison entre uid et login, on peut utiliser plusieurs sources :

- Les fichiers (/etc/passwd...);
- Un service NIS (yp);
- Un annuaire LDAP;
- Une base de donnée relationnelle;
- ...

La sélection des sources à utiliser ainsi que leur priorité est définie dans le fichier `/etc/nsswitch.conf`

## Exemple de fichier nsswitch.conf

```
passwd:      files
group:       files
shadow:      files

hosts:       files dns
networks:    files

protocols:   db files
services:    db files
ethers:      db files
rpc:         db files

netgroup:    nis
```

# Utilisation de LDAP avec NSS

Pour utiliser LDAP avec NSS, il faut :

- installer une library NSS qui fournit une source de données LDAP (`libnss-ldap`, `libnss-ldapd`, `sssd`,...)
- la configurer via son fichier de configuration ;
- modifier le fichier `nsswitch.conf` de façon à utiliser `ldap` comme source pour
  - ▶ `passwd`
  - ▶ `group`
- ajouter des objets LDAP pour représenter les utilisateurs et les groupes :
  - ▶ Les utilisateurs ont le type `posixAccount`,
  - ▶ Les groupes ont le type `posixGroup`,

# Vérification de l'installation

Une fois les étapes précédentes réalisées, vous pouvez interroger la base NSS à l'aide de l'outil `getent`

```
bash$ getent passwd
beaufils:*:1000:1000:Bruno.BEAUFILS:/home/infoens/beaufils:/bin/bash
hauspiem:*:1348:1000:Michael.HAUSPIE:/home/infoens/hauspiem:/bin/bash
bash$ getent group
infoens:*:1000:beaufils,hauspiem
```

# Authentication

- Configurer NSS ne permet au système que d'obtenir les uid et gid des utilisateurs par le LDAP ;
- Pour qu'un utilisateur puisse s'authentifier, il faut configurer PAM de façon à ce que le système puisse vérifier le mot de passe en effectuant une liaison LDAP.



# Pluggable Authentication Module (PAM)

- PAM contrôle l'authentification, le contrôle d'accès à la machine et la configuration de la session de l'utilisateur.
- Configuré via les fichiers situés dans le répertoire `/etc/pam.d`
- Chacun des fichiers présents configure pam pour un service donné :
  - ▶ `/etc/pam.d/su` : contrôle le comportement de la commande `su` ;
  - ▶ `/etc/pam.d/sudo` : contrôle le comportement de la commande `sudo` ;
  - ▶ `/etc/pam.d/login` : contrôle l'utilisation de pam dans le cas d'une connexion ;
  - ▶ `/etc/pam.d/other` : utilisé si le service demandé n'est pas présent. Il est important que ce fichier soit présent dans le système.

# Pluggable Authentication Module (PAM)

Sous debian, ces fichiers incluent en général :

- `/etc/pam.d/common-auth` : authentification (vérification du mot de passe) ;
- `/etc/pam.d/common-account` : une fois l'utilisateur authentifié, on vérifie également qu'il est autorisé à se connecter ;
- `/etc/pam.d/common-password` : concerne le changement de mot de passe (permet de vérifier la complexité d'un mot de passe par exemple) ;
- `/etc/pam.d/common-session` : règle les paramètres de session (affecte les variables d'environnement, crée le répertoire home si nécessaire...).

# Exemple de fichier common-auth

```
auth    [success=2 default=ignore]    pam_unix.so nullok_secure
auth    [success=1 default=ignore]    pam_ldap.so minimum_uid=1000 use_first_pass
auth    requisite                     pam_deny.so
auth    required                      pam_permit.so
```

# Exemple de fichier common-account

```
account [success=1 default=ignore] pam_unix.so
account requisite pam_deny.so

account required pam_permit.so

account [success=ok] pam_ldap.so minimum_uid=1000
```

- Un module PAM gérant LDAP doit être installé (`libpam-ldap`, `libpam-ldapd`, `sssd`, ...)
- Et configuré.

# Plus d'informations sur PAM

## Attention

La configuration de PAM est critique pour la sécurité de votre système! Prenez le temps de lire la documentation officielle pour comprendre le fonctionnement de PAM.

## Documentation officielle

[http://linux-pam.org/Linux-PAM-html/Linux-PAM\\_SAG.html](http://linux-pam.org/Linux-PAM-html/Linux-PAM_SAG.html)

## Un tutoriel intéressant

<http://wpollock.com/AUnix2/PAM-Help.htm>

## Cours n° C.3

# Préambule au partage de fichiers: Rappels Unix - Système de fichiers

# Système de fichier unix, point de montage

- Dans les systèmes unix/linux, une seule arborescence existe
- Pas de notion de volume par exemple comme sous Windows
- La racine du système de fichier est /
- Tout répertoire du système de fichier peut être un point de montage vers un périphérique, un système de fichier distant, etc...
- En particulier, / est souvent le point de montage de la partition principale du disque principal
- /home est souvent un point de montage vers une autre partition ou un autre disque



# Montage

La commande `mount` permet de monter un système de fichier sur un point de montage particulier

- ➡ Par défaut, seul `root` peut monter un système de fichier
- ➡ Le fichier `/etc/fstab` permet de définir des points de montage standards

# Fichier /etc/fstab

- ☞ Chaque ligne représente un point de montage
- ☞ Le format d'une ligne est le suivant  
`systeme_de_fichier /point/de/montage type_sf options dump passnum`
- ☞ `systeme_de_fichier` : le système de fichier à monter (`/dev/hda1` par exemple)
- ☞ `/point/de/montage` : quel répertoire de l'arborescence va être utilisé comme point de montage (`/`, `/home...`)
- ☞ `type_sf` : le type du système de fichier à monter (`ext2`, `ext3`, `nfs`, `smbfs...`)
- ☞ `options` : les options à passer à la commande `mount` (*c.f. `man mount`*)
- ☞ `dump` : 1 si le système de fichier doit être sauvegardé lors d'un appel à la commande `dump`
- ☞ `passnum` : un numéro définissant l'ordre dans lequel les systèmes de fichiers doivent être vérifiés

# Intérêt du /etc/fstab

- ➡ Permet de monter des système de fichier automatiquement au boot du système (option auto)
- ➡ Permet de simplifier l'appel à la commande mount.

➡ `mount /point/de/montage`

au lieu de



`mount -t ext3 -o ro,uid=robert /dev/sda3 /point/de/mon`

- ➡ En utilisant l'option `user`, autorise un utilisateur normal à monter un système de fichier

# Le partage de fichiers

Dans un contexte de réseau local, il est vite nécessaire de permettre à tous les postes clients d'accéder à une même zone de stockage

- ➡ Permettre à tous les utilisateurs d'accéder aux données d'un projet
- ➡ Permettre à tous les utilisateurs d'utiliser n'importe quelle machine tout en retrouvant son espace personnel
- ➡ ...

# Deux protocoles principaux

Les deux protocoles les plus utilisés pour effectuer du partage de fichiers sont :

- 👉 NFS (Network File System)
- 👉 SMB (Server Message Block) dans les premières versions de windows, CIFS (Common Internet File System)

Le premier est principalement utilisé dans le monde Unix/Linux et le deuxième est utilisé par Windows

Cours n° C.4

# NFS - Network File System

# Network File System

- Développé à l'origine par Sun Microsystems en 1984
- Basé sur le protocole d'accès aux procédures distantes Open Network Computing Remote Procedure Call (ONC RPC)
- Définit par les RFCs 1094 (Version 2), 1813 (Version 3, la plus courante) et 3530 (Version 4)

# Remote Procedure Call

- Un protocole d'accès aux procédures distantes permet de demander à une machine (locale ou distante) d'exécuter du code
- Permet de rendre transparent tous les aspects réseaux d'un tel mécanisme
- Les programmes qui proposent des mécanismes RPC sont standardisés et un numéro leur est associé
- NFS est implémenté grâce à ces mécanismes
  - Le serveur attend des appels de procédure distante
  - Le client émet des appels de procédure distante



# Remote Procedure Call

- ☞ Si le numéro de programme est standard, le port (TCP ou UDP) permettant d'accéder à un programme donné ne l'est pas forcément
- ☞ L'association entre numéro de programme et numéro de port est faite par un service, le portmapper qui écoute sur le port 111 en TCP et en UDP
  - ☞ Sous linux, l'implémentation du portmapper est faite par portmap
- ☞ Quelques numéros de programmes classiques et le port associé
  - ☞ 100000 : portmap, port 111
  - ☞ 100003 : nfs, port 2049
  - ☞ 100004 : ypserv, pas d'association standard

# Configuration du serveur NFS

- ☞ Comme NFS se base sur les RPC, il faut impérativement que `portmap` fonctionne
- ☞ Un seul fichier de configuration : `/etc/exports`
- ☞ Chaque entrée de ce fichier correspond à un répertoire à exporter (*i.e.* partager)

# Le fichier /etc/exports

- ➡ Un export par ligne
- ➡ Un export est donné dans le format suivant :  
`/chemin/du/repertoire <liste de clients autorisés>`
- ➡ La liste de clients est constitué d'un ensemble d'adresses et d'options
- ➡ Le format d'un client est le suivant :  
`client(options)`

# Format de définitions des clients

Un client peut être

- ☞ Un simple nom d'hôte (nom DNS ou adresse IP)
- ☞ Un groupe réseau (si NIS est utilisé)
- ☞ Un domaine avec *wildcards* (\*.licence-rt.org)
- ☞ Un sous-réseaux IP (192.168.119.0/24)

# Options d'exportation

De nombreuses options sont disponibles pour contrôler la façon dont le répertoire est exporté<sup>3</sup>

- ☞ `rw/ro*` : Précise si le dossier est partagé en lecture/écriture ou en lecture seule
- ☞ `async/sync*` : Avec `async`, le client n'attend pas la réponse du serveur pour effectuer d'autres opérations. Augmente les performances mais si le serveur plante, des données peuvent être corrompues/perdus. Avec `sync`, le client attend toujours que sa requête ait été satisfaite.

# Contrôle d'accès

L'accès aux fichiers distant se fait par l'UID de l'utilisateur.

- ☞ Le client et le serveur doivent utiliser les même UID
- ☞ Ceci induit un gros problème de sécurité
  - ⇒ l'UID 0 sur le client n'est pas forcément le même utilisateur que sur le serveur.
  - ⇒ En effet, il est possible qu'un utilisateur soit root sur sa machine.
  - ⇒ Généralement, on souhaite que le root du client ne puisse pas être root vis-à-vis du système de fichiers partagé
  - ⇒ L'option `root_squash` permet de *mapper* l'utilisateur root vers l'utilisateur `nobody` du serveur afin d'empêcher n'importe qui de manipuler les fichiers en tant que root. Ceci est l'option par défaut.
  - ⇒ L'option `no_root_squash` permet d'autoriser le root du client à agir en tant que root sur les fichiers du serveur. Ceci sert principalement pour les machines sans-disque

# Contrôle d'accès

On peut également *remapper* tous les utilisateur vers un utilisateur donné (partage d'un répertoire public par exemple)

- ☞ `all_squash` : *map* tous les utilisateurs du client vers l'utilisateur `anonymous` du serveur
- ☞ `anonuid=xxx` : utilise l'utilisateur d'uid `xxx` comme utilisateur `anonymous`
- ☞ `anongid=xxx` : utilise le groupe de gid `xxx` comme groupe `anonymous`

Pour une liste complète des options : `man exports`

# Quelques exemples

```
# sample /etc/exports file

# / utilisable par la machine master et trusty.
# Sur trusty le root peut agir comme root sur le répertoire partagé
/
    master(rw) trusty(rw,no_root_squash)

# /projects peut être monté par toutes les machines
# dont le nom commence par proj du domaine local.domain en lecture/écriture
/projects
    proj*.local.domain(rw)

# Toutes les machines du domaine local.domain peuvent
# accéder à /usr en lecture seule
/usr
    *.local.domain(ro)

# pc001 peut accéder à /home/joe en lecture écriture en forçant
# l'utilisateur 150/100
/home/joe
    pc001(rw,all_squash,anonuid=150,anongid=100)

# /pub, montable par tout le monde en utilisant l'utilisateur anonymous
/pub
    (ro,all_squash)
```



# Client NFS

La machine client monte généralement un partage `nfs` avec la commande `mount`

```
mount [-t nfs] [-o options] serveur:/chemin/vers/repertoire /point/de/montage/local
```

Les options classiques :

- 👉 `rw/ro`

- 👉 `nosuid` : empêche le changement effectif d'utilisateur quand on lance un script dont le `suid` bit est affecté. Important si on ne fait pas forcément confiance au serveur

Pour une liste complète des les options : `man mount`

On peut évidemment ajouter une entrée dans `/etc/fstab` pour faciliter le montage

# Inconvénient de NFS

- ➡ L'inconvénient principal de NFS (au moins jusqu'à sa version 3) est son manque de sécurité
- ➡ En effet, l'accès aux fichiers est contrôlé par l'uid et le gid uniquement !
- ➡ Il suffit d'être root sur une machine client pour pouvoir utiliser l'uid et le gid que l'on désire

Cours n° C.5

# Common Internet File System (CIFS *a.k.a.* SMB)

# Common Internet File System

- ☞ Système de partage utilisé par Windows
- ☞ À l'origine, *Server Message Block* (SMB), développé par IBM
- ☞ Repris et modifié en 1990 par Microsoft et 3Com pour lancer **LAN Manager** qui sera ensuite intégré dans **Windows for Workgroups** en 1992, puis dans tous les Windows
- ☞ Utilise NetBIOS ou TCP/IP comme protocole de transport (Uniquement NetBIOS à l'origine, TCP/IP n'est venu que plus tard)

# Common Internet File System

- Partage de fichiers mais aussi d'imprimantes
- Permet de mettre en place une authentification pour le contrôle d'accès !

# Nommage

La correspondance entre le nom du serveur et l'adresse de celui-ci peut être obtenue de trois façon :

- ✎ par NetBIOS : dans ce cas, NetBIOS sera utilisé comme protocole de transport
  - ✎ Un broadcast est effectué pour demander l'adresse de la machine
  - ✎ Ou un serveur WINS est contacté
- ✎ par une résolution DNS : dans ce cas, TCP/IP sera utilisé comme protocole de transport
- ✎ résolution directe si le serveur est donné par son adresse IP : TCP/IP sera alors utilisé pour le transport

# Implémentation sous linux

- ☞ L'implémentation sous linux est fournie par le logiciel Samba<sup>4</sup>
- ☞ Implémente :
  - 🗄 le système de fichier CIFS
  - 🗄 la gestion de domaine windows (samba peut être un contrôleur de domaine)

---

4. <http://www.samba.org>

# Samba – Configuration

Le fichier de configuration de samba est en général  
`/etc/samba/smb.conf`

- ☞ Format similaire aux fichiers `.ini` windows
- ☞ Contient une série de sections et de variables
- ☞ Une section commence par `[nom_de_section]` et se termine à la prochaine définition de section
- ☞ On peut affecter une valeur à une variable → `variable = valeur`
- ☞ Une seule variable par ligne



# Configuration – Les sections

- ☞ Une section détermine le nom d'un partage ainsi que ses paramètres propres
- ☞ Ex : [toto] définit un partage qui sera vu comme un répertoire nommé toto par les postes clients
- ☞ Il existe des sections spéciales

# Configuration – Les sections

Les sections spéciales sont :

- ☞ `[global]` : contient les paramètres de configuration du serveur (tout ce qui ne concerne pas un partage en particulier)
- ☞ `[homes]` : Cette section définit un partage automatique des répertoires home des utilisateurs unix. Le partage sera vu comme un répertoire portant le nom de l'utilisateur
- ☞ `[printers]` : Définit les paramètres de partage des imprimantes du serveur

# Configuration – La section global

C'est elle qui définit les paramètres du serveur :

- ☞ Son nom : `netbios name = monserver`
- ☞ Son groupe de travail : `workgroup = LICENCE_RT`
- ☞ Sa description : `server string = mon beau server`
- ☞ L'interface réseau sur lequel il écoute :  
`interfaces = eth1 192.168.119.0/24`
- ☞ Les paramètres de sécurité
  - ☞ Authentification par utilisateur local (unix par exemple) :  
`security = user`
  - ☞ Accès libre : `security = share`
  - ☞ Authentification par domaine : `security = domain`
- ☞ ...

# Exemple de configuration

## Serveur public et lecture seule

- ☞ On souhaite mettre en place un serveur de fichiers public (accessible sans authentification)
- ☞ Dont l'accès se fait uniquement en lecture seule

```
[global]
    workgroup = LICENCE_RT
    netbios name = PUBLIC
    security = SHARE

[Public]
    comment = Les fichiers publics
    path = /home/fichiers_publics
    read only = Yes
    guest ok = Yes
```

# Exemple de configuration

Serveur public, lecture seule et imprimante

On suppose qu'un système d'impression est disponible et configuré sur le serveur et que les imprimantes y sont configurées

```
[global]
    workgroup = LICENCE_RT
    netbios name = PUBLIC
    security = share
    printcap name = /etc/printcap
    map to guest = Bad User

[printers]
    path = /var/spool/samba
    printable = Yes
    guest ok = yes
    browseable = yes
```

# Exemple de configuration

## Serveur privé avec un partage public

[global]

```
workgroup = LICENCE_RT
netbios name = SEMI-PUBLIC
security = user
map to guest = Bad User
guest account = sambaguest
invalid user = root
```

[Private]

```
read only = Yes
browseable = Yes
path = /var/samba/private
guest ok = No
```

[Public]

```
read only = No
force create mode = 0660
force directory mode = 2770
browseable = Yes
guest ok = Yes
path = /var/samba/public
```

Cours n° C.6

# Contrôleur de domaine Windows avec Samba et LDAP

# Rôle du contrôleur de domaine

Gérer :

- l'authentification ;
- l'espace disque utilisateur ;
- les profils itinérants ;
- les machines.



# Le SID (Security ID)

Le SID est l'identifiant de sécurité de chaque "objet" dans le domaine :

- Domaine ;
- Machine ;
- Utilisateur ;
- Groupe ;

Dans le cas des objets autres que le domaine, le SID est constitué à partir du SID du domaine et d'un RID (Relative IDentifier).

```
bash# net getdomainsid
SID for domain IUT_INFO_ENS is: S-1-5-21-2184002573-2487086478-3822586455
bash# pdbedit -L -v hauspiem
[...]
User SID:                S-1-5-21-2184002573-2487086478-3822586455-3696
Primary Group SID:       S-1-5-21-2184002573-2487086478-3822586455-1000
[...]
```

# Calcul du RID

Pour les groupes :

$$RID = GID$$

Pour les utilisateurs :

$$RID = (UID \times 2) + 1000$$

# SID particuliers

Well-Known Entity	RID	Type	Essential
Domain Administrator	500	User	No
Domain Guest	501	User	No
Domain KRBTGT	502	User	No
Domain Admins	512	Group	Yes
Domain Users	513	Group	Yes
Domain Guests	514	Group	Yes
Domain Computers	515	Group	No
Domain Controllers	516	Group	No
Domain Certificate Admins	517	Group	No
Domain Schema Admins	518	Group	No
Domain Enterprise Admins	519	Group	No
Domain Policy Admins	520	Group	No
Builtin Admins	544	Alias	No
Builtin users	545	Alias	No
Builtin Guests	546	Alias	No
Builtin Power Users	547	Alias	No
Builtin Account Operators	548	Alias	No
Builtin System Operators	549	Alias	No
Builtin Print Operators	550	Alias	No
Builtin Backup Operators	551	Alias	No
Builtin Replicator	552	Alias	No
Builtin RAS Servers	553	Alias	No

# Intégration dans le LDAP

- Le LDAP doit utiliser le schéma `samba.schema` fournit dans le package `samba-doc` ;
- Un objet doit être créé, de type `sambaDomain`, qui contient le SID du domaine ;
- Les utilisateurs doivent être de type `sambaSamAccount` en plus du type `posixAccount` pour associer les utilisateurs samba et les utilisateurs unix ;
- Les groupes doivent être de type `sambaGroupMapping` en plus du type `posixGroup` pour associer les groupes samba et les groupes unix.

# Modification de la configuration LDAP

Il faut :

- charger le schéma : `include /etc/ldap/schema/samba.schema`
- ajouter des index sur les champs samba :
  - ▶ `sambaSID ;`
  - ▶ `sambaPrimaryGroupSID ;`
  - ▶ `sambaDomainName ;`
  - ▶ `sambaGroupType ;`
  - ▶ `sambaSIDList.`

# Modification de la configuration samba

```
security = domain  
passdb backend = ldapsam:ldap://ldap.rt-cgir.org  
ldap suffix = dc=rt-cgir,dc=org  
ldap machine suffix = ou=computers  
ldap user suffix = ou=people  
ldap admin dn = "cn=admin,dc=rt-cgir,dc=org"  
ldap delete dn = no
```

Le mot de passe de l'administrateur LDAP doit être stocké dans la base de mot de passe de samba à l'aide de la commande :

```
smbpasswd -w motdepasse
```

# Ajout de poste client au domaine

Pour qu'une machine cliente (windows) puisse intégrer le domaine, elle doit avoir un compte.

Le compte d'une machine est le nom de cette machine suivi du caractère \$

# Gestion des comptes

Le paquet `smbldap-tools` contient des scripts qui permette de gérer les comptes utilisateurs facilement :

- Modifie directement la base LDAP ;
- Permet de créer/modifier les comptes en modifiant à la fois les attributs posix ET samba ;
- Configuré dans `/etc/smbldap-tools/smbldap.conf`
- Création à partir d'un LDAP vide: `smbldap-populate`



# Exemple de LDIF d'un compte utilisateur

```
dn: uid=hauspiem,ou=people,dc=rt-cgir,dc=org
uid: hauspiem
cn: Michael.HAUSPIE
uidNumber: 1348
gidNumber: 1000
homeDirectory: /home/hauspiem
sn: HAUSPIE
givenName: Michael
loginShell: /bin/bash
sambaSID: S-1-5-21-2184002573-2487086478-3822586455-3696
sambaPrimaryGroupSID: S-1-5-21-2184002573-2487086478-3822586455-1000
sambaAcctFlags: [UX          ]
sambaPwdMustChange: 9999999999
sambaPwdLastSet: 1
objectClass: top
objectClass: person
objectClass: posixAccount
objectClass: sambaSamAccount
```

## Partie D

# Courrier électronique



IUT A

Cours n° D.1

# Principes de base

# Fonctionnement du mail

## ☞ Système distribué

- ✎ L'administrateur d'un domaine est chargé de fournir un serveur de mail
- ✎ L'adresse de ce serveur mail est fournie par le serveur DNS du domaine grâce à un enregistrement MX

☞ Quand on cherche à écrire un message à utilisateur@domaine, c'est ce serveur mail qui est contacté

# Les protocoles de gestion des mails

On distingue deux types de protocoles

- ➡ Les protocoles d'émission
- ➡ Les protocoles de consultation

- ➡ Pour l'émission, un seul protocole est utilisé : SMTP
- ➡ Pour la consultation, les deux principaux protocoles sont POP et IMAP

Cours n° D.2

# Simple Mail Transfert Protocol (SMTP)

Le but du protocole SMTP est d'acheminer les e-mails à leur(s) destinataire(s)

Quand on envoie un e-mail, les étapes suivantes sont généralement suivies :

- ① L'utilisateur écrit son mail avec un client mail
- ② Le client mail contacte un serveur SMTP **relais**
- ③ Le serveur relais vérifie que l'utilisateur est bien autorisé à l'utiliser
- ④ Le serveur relais demande au serveur DNS gérant le domaine du destinataire l'adresse du serveur SMTP gérant les adresses concernées
- ⑤ Le serveur relais contacte ce serveur et y dépose le mail si ce dernier l'y autorise

# Protocole SMTP

- ☞ Décrit par la RFC 821
- ☞ Protocole texte (utilisable avec telnet ou nc)
- ☞ Utilise le port TCP 25
- ☞ Trois phases principales
  - ① L'ouverture de la session
  - ② Le dépôt de messages
  - ③ La fermeture de la session



# Protocole SMTP - Ouverture de session

- ☞ Le client se présente à l'aide de la commande HELO en donnant son domaine

✎ HELO lifl.fr

- ☞ Le serveur répond en donnant son domaine

✎ 250 mx1.univ-lille1.fr Hello hauspie@kwak.lifl.fr  
[134.206.11.14], pleased to meet you

- ☞ Permet aux serveurs de vérifier avec qui ils sont en train de parler

# Protocole SMTP - Dépôt de message

Le dépôt de message s'effectue en plusieurs étapes

- ① On indique au serveur l'expéditeur grâce à la commande MAIL FROM:  
✎ MAIL FROM: hauspie@lifl.fr
- ② Le serveur vérifie (ou non, en fonction de sa configuration) l'adresse de l'expéditeur et accepte (ou non)  
✎ 250 2.1.0 hauspie@lifl.fr... Sender ok
- ③ Le client donne ensuite la liste des destinataires grâce à la commande RCPT TO: (utilisée éventuellement plusieurs fois)  
✎ RCPT TO: Michael.Hauspie@lifl.fr
- ④ Le serveur accepte ou non le destinataire  
✎ 250 2.1.5 hauspie@lifl.fr... Recipient ok

# Protocole SMTP - Dépôt de message

- ⑤ On indique le début du corps du message grâce à la commande DATA
  - ✎ DATA
- ⑥ Le serveur répond en spécifiant comment le client doit indiquer la fin des données
  - ✎ 354 Enter mail, end with "." on a line by itself
- ⑦ On envoie le corps du message proprement dit
  - ✎ Subject: Bonjour  
Bonjour  
.
- ⑧ Le serveur accepte le dépôt
  - ✎ 250 2.0.0 19H9S5j3006336 Message accepted for delivery

# Protocole SMTP - Fin de la session

- ➡ La session se termine simplement par la commande QUIT
- ➡ Le serveur ferme la connexion après un message
  - ✎ `221 2.0.0 mx1.univ-lille1.fr closing connection`

# Protocole SMTP

```
$ telnet smtp.univ-lille1.fr 25
Trying 193.49.225.20...
Connected to smtp.univ-lille1.fr.
Escape character is '^]'.
220 mx2.univ-lille1.fr ESMTP Sendmail 8.14.0/8.14.0; Wed, 17 Oct 2007 14:38:16 +0200
HELO lifl.fr
250 mx2.univ-lille1.fr Hello hauspie@kwak.lifl.fr [134.206.11.14], pleased to meet you
MAIL FROM: Michael.Hauspie@lifl.fr
250 2.1.0 Michael.Hauspie@lifl.fr... Sender ok
RCPT TO: Michael.Hauspie@lifl.fr
250 2.1.5 Michael.Hauspie@lifl.fr... Recipient ok
DATA
354 Enter mail, end with "." on a line by itself
Subject: Mail de test
Envoi d'un mail de test à moi même
.
250 2.0.0 19HCcGnF004137 Message accepted for delivery
quit
221 2.0.0 mx2.univ-lille1.fr closing connection
Connection closed by foreign host.
```

# Implémentation SMTP sous linux

Les principaux serveurs SMTP présents sous linux :

## ① Sendmail

- ☞ Populaire car disponible sur toutes les plateformes Unix (ou presque)
- ☞ Mais critiqué pour sa lenteur et surtout sa difficulté d'installation et de configuration

## ② Exim

## ③ Postfix

- ☞ Fiable et souple
- ☞ Permet de déléguer l'analyse des mails à un programme externe (très utile pour le spam)

- La configuration d'un serveur postfix se fait principalement grâce au fichier `/etc/postfix/main.cf`
- Comme pour beaucoup de services, il y a beaucoup de paramètres configurables
- Les principaux vont concerner
  - Nom d'hôte du serveur et nom du domaine qu'il gère
  - Les paramètres réseau
  - Les destinations du serveur
  - Les paramètres de relais
  - Le format des adresses mails des utilisateurs

# Format général du fichier de configuration

- ➡ Les commentaires sont des commentaires ligne, précédés du caractère #
- ➡ Tous les paramètres de configuration sont un couple variable/valeur de la forme
  - ✎ `variable = valeur`
- ➡ `man 5 postconf` permet d'en savoir plus sur le fichier de configuration



# Hôte, domaine

👉 myhostname : Nom qualifié complet du serveur

📝 myhostname = mail.licence-rt.org

👉 mydomain : Domaine géré par le serveur

📝 mydomain = licence-rt.org

👉 myorigin : Domaine utilisé par les mails envoyés par les utilisateurs locaux

📝 myorigin = \$mydomain

☞ `inet_interfaces` : permet de définir sur quelle interfaces le serveur va écouter

☞ `inet_interfaces = all`

☞ `inet_interfaces = loopback-only`

☞ `inet_interfaces = $myhostname, localhost`

☞ `proxy_interfaces` : précise l'adresse IP du serveur tel qu'il est vu par le reste du monde dans le cas où celui-ci est situé derrière une passerelle qui effectue de la translation d'adresse

☞ `proxy_interfaces = 88.176.128.213`

# Postfix

## Destinations et relais

Lorsqu'un message destiné à `utilisateur@domaine` doit être déposé, deux cas de figures se présentent :

- ① Le domaine est considéré comme local par le serveur
  - ☞ Si l'utilisateur existe, le message est déposé dans sa boîte
  - ☞ Sinon, un message d'erreur est retourné à l'expéditeur
  - ☞ Le paramètre `mydestination` permet de configurer les domaines considérés comme locaux
    - ➡ `mydestination = $myhostname, $mydomain`
- ② Sinon, le serveur tente de relayer le message.

# Postfix

## Règles de relais

Si le serveur doit relayer un message, il ne le fait que si sa configuration l'y autorise :

- ☞ Le but principal est d'éviter qu'un serveur SMTP soit utilisé de manière publique pour relayer des spams
- ☞ `mynetworks` : Permet de définir quels sont les réseaux de confiance, *i.e.* les réseaux pour lesquels le serveur veut bien relayer les messages
  - 📎 `mynetworks = 192.168.119.0/24, 127.0.0.0/8`

Si le serveur doit relayer un message, il peut le faire de deux façons :

- ① Il contacte directement le serveur chargé de gérer le domaine destination
  - ☞ Il contacte le serveur DNS du domaine pour obtenir l'enregistrement MX puis, il contacte le serveur correspondant
- ② Il contacte un autre serveur SMTP qui se chargera du relais
  - ☞ C'est souvent le SMTP du fournisseur d'accès à internet qui est utilisé dans ce cas
  - ☞ La variable `relay_host` permet de définir l'adresse du serveur relais
  - ☞ `relayhost = smtp.univ-lille1.fr`

# Postfix

## Alias

Par défaut, les utilisateurs du serveur sont uniquement les utilisateurs unix de la machine.

On peut définir des alias pour ajouter de *faux* utilisateurs

Par exemple :

- ✎ `root@licence-rt.org → lpirt@licence-rt.org`

- ✎ Tous les messages envoyés à `root@licence-rt.org` seront en fait reçus par `lpirt@licence-rt.org`

- ✎ `lpirt@licence-rt.org → Michael.Hauspie@lifl.fr`

- ✎ Tous les messages envoyés à `lpirt@licence-rt.org` seront en fait reçus par `Michaël.Hauspie@lifl.fr`

- ☞ La variable `alias_maps` permet de définir une base d'alias (`man aliases`)
- ☞ La variable `canonical_maps` permet de définir des règles de réécriture d'adresses pour les mails entrant et sortant (`man postmap`). Attention, pour les mails entrant, les adresses sont modifiées par les règles mais cela ne permet pas de faire arriver un mail à un utilisateur inconnu (il faut utiliser `alias_maps` pour cela)
  - ☞ Très utilisé pour effectuer une transformation  
`login` → `Prenom.Nom@domaine`

# Postfix

## Format de stockage de la boîte mail

Le format de stockage de la boîte mail est défini par la variable `home_mailbox`. Les possibilités sont :

- ① Le format `mailbox` : tous les messages sont stockés dans un seul fichier.
  - 👉 `home_mailbox = chemin_du_fichier_mbox` ou rien
  - 👉 Le nom du fichier `mbox` est relatif au répertoire de l'utilisateur
  - 👉 Si rien n'est indiqué le fichier est par défaut  
`/var/spool/mail/login`
- ② Le format `Maildir` : les messages sont stockés dans des fichiers différents maintenant une arborescence. C'est ce format qu'il faut utiliser si on veut pouvoir utiliser un serveur IMAP
  - 👉 `home_mailbox = RepertoireMaildir/`
  - 👉 Le répertoire est relatif au répertoire `home` de l'utilisateur. Le `'/'` est **important**



Pendant les étapes de configuration il est conseillé de régler l'option  
`soft_bounce = yes`

Cela aura pour effet de ne pas effectuer de rebond en cas d'erreur de mail

Cours n° D.3

# La consultation d'emails

# La consultation d'emails

On peut consulter ses mails de deux façons :

- ➡ Être connecté sur la machine où se situe le dépôt de mail
- ➡ Utiliser des services de consultation distante

# La consultation locale

En fonction de la configuration du serveur SMTP, les mails locaux sont disponibles :

- ➡ Dans un fichier au format mailbox
- ➡ Dans un répertoire au format Maildir

On peut alors les consulter directement (cat, less, etc...) ou en utilisant des commandes comme mail, mutt...

# La consultation distante

La consultation locale n'est pas la plus pratique

- ➡ Il faut être connecté sur la machine
- ➡ À distance, il faut se connecter en SSH
- ➡ Et surtout, on a pas forcément (et même rarement) accès à un serveur mail avec un shell

# La consultation distante

Pour palier à ces inconvénients, il a été proposé des protocoles de consultation d'emails distants

- Le protocole POP (Post Office Protocol)
- Le protocole IMAP (Internet Message Access Protocol)

# Le protocole POP

Le protocole POP (Post Office Protocol) est un des premiers protocole de consultation d'emails à distance

- 👉 Version d'origine : RFC 918 (1984)
- 👉 Actuellement en version 3 : POP3, RFC 1939 (1996)
- 👉 Conçu pour un usage simple
  - ➡ Récupérer les emails sur la machine client
  - ➡ Les supprimer du serveur

# Le protocole POP

- ☞ Utilise le port TCP 110
- ☞ Existe en version TLS sur le port 995



# Implémentation

Comme pour beaucoup de services, il existe nombre d'implémentations différentes du protocole POP sous linux :

- ☞ Cyrus
- ☞ Courier
- ☞ QPopper
- ☞ ...

# Inconvénients de POP

L'inconvénient principal du protocole POP réside dans sa simplicité

- ☞ les seules opérations disponibles sont la récupération et la suppression d'emails
- ☞ pas pratique si on utilise plusieurs machines différentes

La solution, le successeur de POP, **IMAP**

# Le protocole IMAP

Le protocole IMAP (Internet Message Access Protocol) est apparu peu après POP

- ➡ IMAP Version 2, RFC 1064 (1988)
- ➡ Actuellement, IMAP Version 4 , RFC 3501 (2003)

Le but principal d'IMAP est de permettre de manipuler la boîte mail directement sur le serveur

- ➡ Permet d'avoir toujours une synchronisation des différentes machines avec le véritable contenu de la boîte
- ➡ Gère plusieurs boîtes (ou dossiers)
- ➡ Permet des recherches et des manipulations de messages sans avoir à les télécharger

# Le protocole IMAP

Le protocole IMAP utilise une connexion TCP en mode texte

- ☞ en clair sur le port 143
- ☞ il existe aussi une version TLS sur le port 993

# Implémentation

Comme pour POP, il existe beaucoup d'implémentations différentes sous linux

- ☞ La plupart des implémentations fournissent à la fois POP et IMAP
- ☞ Attention, beaucoup d'implémentations récentes utilisent le format Maildir !
- ☞ On peut citer
  - ✎ Courrier
  - ✎ Cyrus
  - ✎ ...