

FORMATUX - Support de cours GNU/Linux

Les services sur CentOS 6

Version 1

17-04-2017

Table des matières

Préface	ix
1. Crédits	x
2. Licence	x
3. Gestion des versions	xi
1. Network File System	1
1.1. Généralités	1
1.2. Installation	1
1.3. Configuration du serveur	2
1.3.1. Le fichier /etc/exports	2
1.3.2. Permissions sur les ressources	3
1.3.3. Cas concrètes	3
1.3.4. La commande exportfs	4
1.3.5. La commande showmount	4
1.4. Configuration du client	5
2. Serveur de noms DNS - Bind	7
2.1. Généralités	7
2.2. Installation du service	8
2.2.1. Cache DNS	10
2.2.2. Récursivité	10
2.2.3. Architecture DNS	11
2.3. Configuration du serveur	12
2.3.1. Le fichier /etc/named.conf	12
2.3.2. Les rôles	14
2.4. Fichiers de zone	15
2.4.1. Les types d'enregistrements	16
2.4.2. Fichier de zone directe	17
2.4.3. Le fichier de zone inverse	18
2.4.4. La commande nsupdate	19
2.4.5. La commande rndc	19
2.4.6. Le suivi des logs	20
2.5. Configuration du client	20
2.5.1. La commande dig	22
2.5.2. Mise en cache côté client	23
2.5.3. Mise à jour dynamique	23

2.6. Configuration du pare-feu serveur	23
3. Serveur de fichiers Samba	25
3.1. Le protocole SMB	26
3.2. Le protocole CIFS	26
3.3. Installation de Samba	26
3.4. Sécurité SELinux	28
3.5. La configuration de SAMBA	29
3.5.1. Les niveaux de sécurité	30
3.5.2. Les variables internes à Samba	30
3.5.3. La commande testparm	31
3.5.4. La section [global]	32
3.5.5. La section [homes]	33
3.5.6. La section [printers]	33
3.5.7. Partages personnalisés	34
3.5.8. La directive force group	36
3.6. Commandes d'administration	37
3.6.1. La commande pdbeedit	37
3.6.2. La commande smbpasswd	38
3.6.3. La commande smbclient	39
4. Serveur web Apache	41
4.1. Le protocole HTTP	43
4.1.1. Les URL	44
4.1.2. Les ports	45
4.2. Installation du serveur	45
4.2.1. Installation par rpm	46
4.2.2. Installation par yum	47
4.2.3. Les fichiers de configuration	48
4.2.4. Manuel d'utilisation	48
4.2.5. Lancement du serveur	48
4.2.6. Parefeu	49
4.3. Arborescence	49
4.4. Configuration du serveur	50
4.4.1. Section 1	51
4.4.2. Section 2	57
4.5. Configuration avancée du serveur	62
4.5.1. Le mod_status	62

4.5.2. Hébergement mutualisé (section 3)	64
4.6. Exemple de publication de sites	66
4.7. Redirection des logs vers syslog	67
5. Sécurisation du serveur web Apache	69
5.1. Introduction	69
5.2. Masquage de l'identité d'Apache	70
5.2.1. Directive ServerSignature	70
5.2.2. Directive ServerTokens	71
5.3. Gestion des authentifications	73
5.3.1. Authentification classique	74
5.3.2. Directive AuthType	75
5.3.3. Commande htpasswd	79
5.4. Utilisation du module SSL	80
5.4.1. Prérequis	81
5.4.2. Établissement d'une session TCP https (port 443)	81
5.4.3. Mise en place d'un site TLS	82
5.4.4. Le logiciel OpenSSL	83
6. Apache - haute disponibilité	89
6.1. Introduction	89
6.2. Serveur Applicatif	89
6.3. Reverse proxy	90
6.3.1. Le module mod_proxy	91
6.4. Simuler la charge	93
6.5. Répartir la charge	94
6.5.1. Le module mod_proxy_balancer	94
6.5.2. Le module mod_status	95
6.6. Tolérance aux pannes	96
7. Serveur de messagerie Postfix	97
7.1. Généralités	97
7.1.1. Les agents de messagerie	100
7.1.2. Les relais SMTP	102
7.1.3. Les formats de stockage	103
7.1.4. Synoptique	104
7.2. Installation du service	105
7.3. Arborescence et fichiers	106
7.4. Mise en oeuvre	106

7.4.1. Déroulé d'une session SMTP	107
7.4.2. La commande mailx	109
7.4.3. La commande swaks	113
7.5. Configuration du serveur	113
7.5.1. Les alias	114
7.5.2. Configurer un serveur relais	115
7.5.3. Prendre en compte un domaine	116
7.5.4. La directive mynetworks	117
7.5.5. Le format de stockage	118
7.5.6. La table de routage	119
7.6. Protocoles POP/IMAP	119
7.7. Architecture de postfix	120
7.7.1. Démons agents de courrier	123
7.7.2. Files d'attente	126
7.8. Boites aux lettres virtuelles	126
7.9. Suivi des messages à des fins légales	129
8. Serveur d'annuaire OpenLDAP	131
8.1. Généralités	131
8.1.1. La DIT	132
8.1.2. L'OLC	132
8.1.3. Le schéma	133
8.1.4. SASL	133
8.1.5. Le format LDIF	135
8.1.6. Les outils clients LDAP	136
8.2. Installation du serveur	136
8.3. Configuration du serveur	138
8.3.1. Le suffixe	139
8.3.2. Le RootDN et son mot de passe	139
8.3.3. Connexion avec le RootDN	140
8.3.4. La commande slapcat	141
8.3.5. La commande ldapmodify	141
8.3.6. La structure de la DIT	144
8.3.7. Activation des logs	145
8.3.8. Activation du TLS	146
9. Installation d'un serveur Shinken	151
9.1. Généralités	151

9.2. Prérequis	152
9.3. Installation	153
9.3.1. Installation des composants nécessaires à Shinken	153
9.3.2. Installation de shinken	154
9.3.3. Installation et configuration des modules	155
9.4. Démarrage de shinken	156
9.5. Références	157
10. Serveur proxy SQUID	159
10.1. Principes de fonctionnement	159
10.2. Le serveur SQUID	163
10.2.1. Dimensionnement	164
10.2.2. Installation	164
10.2.3. Arborescence et fichiers du serveur Squid	164
10.2.4. La commande squid	165
10.3. Configuration basique	166
10.4. Configurations avancées	169
10.4.1. Les Access Control List (ACL)	169
10.4.2. Les algorithmes de cache	170
10.5. Authentification des clients	171
10.6. Outils	171
10.6.1. La commande squidclient	171
10.6.2. Analyser les logs	171
10.6.3. La commande sarg	172
10.6.4. SquidGuard	173
11. Serveur de log Syslog	175
11.1. Généralités	175
11.1.1. Les catégories de messages	177
11.2. Client Syslog	179
11.2.1. La commande logwatch	180
11.3. Serveur Syslog	184
11.3.1. Stocker les logs dans des fichiers différenciés	184
11.4. Stockage en base de données	185
Glossaire	187
Index	189

Préface

Table des matières

1. Crédits	x
2. Licence	x
3. Gestion des versions	xi

GNU/Linux est un **système d'exploitation** libre fonctionnant sur la base d'un **noyau Linux**, également appelé **kernel Linux**.

Linux est une implémentation libre du système **UNIX** et respecte les spécifications **POSIX**.

GNU/Linux est généralement distribué dans un ensemble cohérent de logiciels, assemblés autour du noyau Linux et prêt à être installé. Cet ensemble porte le nom de "**Distribution**".

- La plus ancienne des distributions est la distribution **Slackware**.
- Les plus connues et utilisées sont les distributions **Debian**, **Redhat** et **Arch**, et servent de base pour d'autres distributions comme **Ubuntu**, **CentOS**, **Fedora**, **Mageia** ou **Manjaro**.

Chaque distribution présente des particularités et peut être développée pour répondre à des besoins très précis :

- services d'infrastructure ;
- pare-feu ;
- serveur multimédia ;
- serveur de stockage ;
- etc.

La distribution présentée dans ces pages est la **CentOS**, qui est le pendant gratuit de la distribution **RedHat**. La distribution **CentOS** est particulièrement adaptée pour un usage sur des serveurs d'entreprises.

1. Crédits

Ce support de cours a été rédigé par les formateurs :

- Patrick Finet ;
- Antoine Le Morvan ;
- Xavier Sauvignon ;

2. Licence

Formatux propose des supports de cours Linux libres de droits à destination des formateurs ou des personnes désireuses d'apprendre à administrer un système Linux en autodidacte.

Les supports de Formatux sont publiés sous licence Creative Commons-BY-SA et sous licence Art Libre. Vous êtes ainsi libre de copier, de diffuser et de transformer librement les œuvres dans le respect des droits de l'auteur.

BY : Paternité. Vous devez citer le nom de l'auteur original.

SA : Partage des Conditions Initiales à l'Identique.

- Licence Creative Commons-BY-SA : <https://creativecommons.org/licenses/by-sa/3.0/fr/>
- Licence Art Libre : <http://artlibre.org/>

Les documents de formatux et leurs sources sont librements téléchargeables sur framagit :

- <https://framagit.org/alemorvan/formatux.fr-support/>

Vous y trouverez la dernière version de ce document.

A partir des sources, vous pouvez générer votre support de formation personnalisé. Nous vous recommandons le logiciel AsciidocFX téléchargeable ici : <http://asciidocfx.com/>

3. Gestion des versions

Tableau 1. Historique des versions du document

Version	Date	Observations
1.0	Avril 2017	Version initiale.

1

Network File System

NFS (Network File System) est un système de partage de fichiers via montage réseau.

1.1. Généralités

NFS est basé sur un fonctionnement client-serveur : le serveur met à disposition des ressources pour tout ou partie du réseau (clients).

Le dialogue entre les clients et le serveur se fait grâce aux services RPC (Remote Procedure Call ou procédure d'appel à distance).

Les fichiers distants sont montés dans un répertoire et apparaissent comme un système de fichiers local. Les utilisateurs clients accèdent en toute transparence aux fichiers partagés par le serveur, en parcourant les répertoires comme s'ils étaient locaux.

1.2. Installation

2 services sont nécessaires au fonctionnement de NFS :

- Le service network ;
- Le service rpcbind.

L'état des services peut être visualisé par les commandes :

```
service rpcbind status  
service network status
```

Si le paquet nfs-utils n'est pas installé :

```
yum install nfs-utils
```

Le paquet nfs-utils nécessite plusieurs dépendances dont nfs-utils-lib et rpcbind pour s'installer.

Le service nfs peut être démarré :

```
chkconfig nfs on  
service nfs start
```

L'installation du service NFS crée deux utilisateurs :

- nfsnobody : utilisé lors des connexions anonymes ;
- rpcuser : pour le fonctionnement du protocole RPC.

1.3. Configuration du serveur



Les droits du répertoire et les droits NFS doivent être cohérents.

1.3.1. Le fichier /etc(exports

Le paramétrage des partages de ressources s'effectue dans le fichier /etc/exports. A une ligne de ce fichier correspond un partage nfs.

Syntaxe du fichier /etc/exports.

```
/partage client1(permission) client2(permission)
```

- **/partage** : Chemin **absolu** du répertoire partagé ;
- **clients** : Machines autorisées à accéder aux ressources ;
- **(permissions)** : Permissions sur les ressources.

Les machines autorisées à accéder aux ressources peuvent être déclarées par :

- Adresse IP : 192.168.1.2
- Adresse réseau : 192.168.1.0/255.255.255.0 ou au format CIDR 192.168.1.0/24
- FQDN : client_*._formatux.lan : autorise les fqdn commençant par client_ du domaine formatux.lan ;
- * pour tout le monde.

Plusieurs clients peuvent être spécifiés sur la même ligne en les séparants par un espace.

1.3.2. Permissions sur les ressources

Il existe deux types de permissions :

- ro : lecture seule ;
- rw : modification.

Si aucun droit n'est précisé, alors le droit appliqué sera lecture seule.

Par défaut les UID et GID des utilisateurs du client sont conservés (excepté pour root).

Pour forcer l'utilisation d'un UID ou d'un GID différent de l'utilisateur qui écrit la ressource, il faudra spécifier les options anonuid=UID et anongid=GID ou donner un accès anonyme aux données avec l'option **all squash** (squash dans sens d'écraser).



Il existe un paramètre, no_root_squash, qui permet d'identifier l'utilisateur root du client comme étant celui du serveur. Ce paramètre peut être dangereux pour la sécurité du système.

Par défaut, c'est le paramètre root_squash qui est activé (même si non précisé), identifiant root comme utilisateur anonyme.

1.3.3. Cas concrêts

- /partage client(ro,all_squash)

Les utilisateurs du client n'ont accès qu'en lecture seule aux ressources et sont identifiés comme anonyme sur le serveur.

- /partage client(rw)

Les utilisateurs du client peuvent modifier les ressources et gardent leur UID sur le serveur. Seul root est identifié comme anonyme.

- /partage client1(rw) client2(ro)

Les utilisateurs du poste client1 peuvent modifier les ressources tandis que ceux du poste client2 n'ont qu'un accès en lecture seule.

Les UID sont gardés sur le serveur, seul root est identifié comme anonyme.

- /partage client(rw,all_squash,anonuid=1001,anongid=100)

Les utilisateurs du poste client1 peuvent modifier les ressources. Leurs UID sont transformés en 1001 et leur GID en 100 sur le serveur.

1.3.4. La commande exportfs

La commande exportfs (exported file systems) permet de gérer la table des fichiers locaux partagés avec les clients NFS.

Syntaxe de la commande exportfs.

```
exportfs [-a] [-r] [-u partage] [-v]
```

Tableau 1.1. Options de la commande exportfs

Option	Description
-a	Active les partages NFS
-r	Prend en compte les partages du fichier /etc(exports
-u partage	Désactive un partage donné
-v	Affiche la liste des partages

1.3.5. La commande showmount

La commande showmount permet de surveiller les clients.

Syntaxe de la commande showmount.

```
showmount [-a] [-e] [hôte]
```

Tableau 1.2. Options de la commande showmount

Option	Description
-e	Affiche les partages du serveur désigné
-a	Affiche tous les partages en cours sur le serveur

Cette commande permet aussi de savoir si le poste client a l'autorisation de monter les ressources partagées.



"showmount" trie et supprime les doublons dans les résultats (sort|uniq), il est donc impossible de déterminer si un client a fait plusieurs montages d'un même répertoire.

1.4. Configuration du client

L'accès aux ressources partagé d'un serveur NFS se fait par point de montage sur le client.

Si besoin, créer le dossier local pour le montage :

```
[root]# mkdir /mnt/nfs
```

Lister les partages NFS disponibles du serveur :

```
[root]# showmount -e 172.16.69.237
/partage *
```

Monter le partage NFS du serveur :

```
[root]# mount -t nfs 172.16.69.237:/partage /mnt/nfs
```

Le montage peut aussi être automatisé au démarrage du système dans le fichier /etc/fstab :

```
[root]# vim /etc/fstab
```

Configuration du client

```
172.16.69.237:/partage /mnt/nfs nfs defaults 0 0
```

2

Serveur de noms DNS - Bind

Le DNS (Domain Name System) est un système de résolution de nom.

Il s'agit d'une base de données distribuée hiérarchique, dont la racine est symbolisée par un « . ».

L'interrogation de cette base se fait selon le principe de client-serveur.

L'URL www.formatux.lan est appelée FQDN (Fully Qualified Domain Name).

Une table de correspondances permet la traduction d'un FQDN en informations de plusieurs types qui y sont associées, comme par exemple son adresse IP.

2.1. Généralités

Il existe 13 serveurs racines répartis dans le monde, dont une majorité se situe en Amérique du Nord.

La traduction d'un FQDN en adresse IP est le cas que l'on rencontre le plus fréquemment mais DNS permet également de renseigner le système sur les serveurs de messagerie, sur les services disponibles, de faire des alias de noms de service, etc.

Par exemple, il est possible de proposer un FQDN `smtp.domain.tld` sans qu'il y ait réellement de serveur nommé SMTP.

La résolution est possible en IPv4 comme en IPv6.

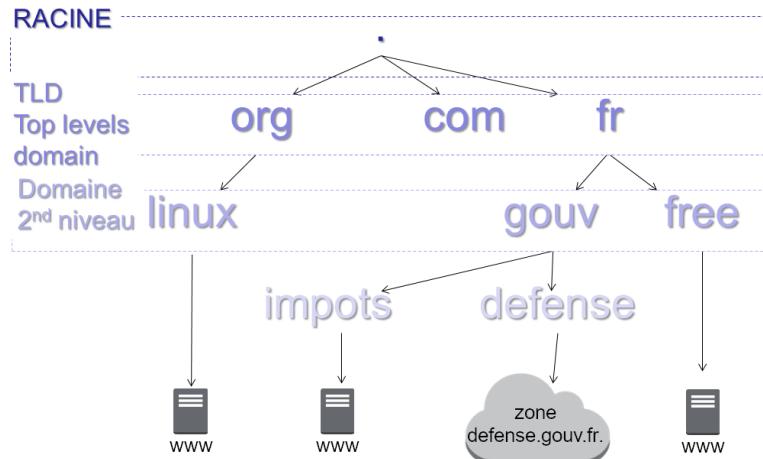


Figure 2.1. Principe de fonctionnement du DNS

Les Top Levels Domain TLD sont gérés par des organismes bien définis. Les Registrars se chargent pour les clients d'enregistrer les noms de domaine auprès de ces organismes.

Le dialogue se fait en général via le protocole UDP (User Datagram Protocol) mais parfois aussi en TCP sur le port 53.

BIND (Berkeley Internet Name Daemon) est un serveur DNS populaire tournant sur système UNIX / Linux.

Bind est disponible sur les serveurs RedHat :

- RHEL 5 : Version 9.3
- RHEL 6 : Version 9.8
- RHEL 7 : Version 9.9



Le protocole peut aussi être utilisé en TCP (connecté) dans certains cas : transferts vers le ou les serveurs secondaires, requêtes dont la réponse est supérieure à la taille d'un paquet UDP, etc.

2.2. Installation du service

Le serveur DNS BIND s'articule autour du service named.

Installation à partir d'un dépôt YUM :

```
[root]# yum install bind
```

L'installation du service BIND ajoute un utilisateur named. Cet utilisateur sera le propriétaire des fichiers de configuration.

```
[root]# grep named /etc/passwd
named:x:25:25:Named:/var/named:/sbin/nologin
```

BIND étant un service réseau, il faut le paramétrier pour un démarrage dans les niveaux 3 et 5, puis le démarrer :

```
[root]# chkconfig --level 35 named on
[root]# service named start
```



Il peut être utile d'installer le paquet bind-utils pour disposer d'outils de test DNS.

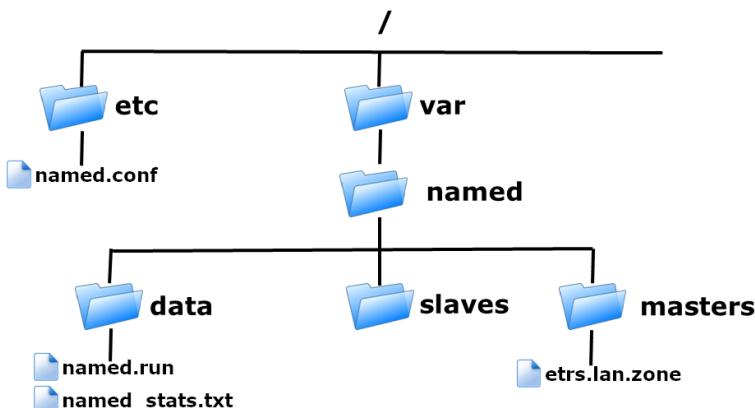


Figure 2.2. Arborescence du service Bind

Le dossier /var/named/masters n'est pas créé par défaut. Il faudra le créer et ne pas oublier d'y attribuer les droits à l'utilisateur named.

Le fichier de statistiques est généré par la commande rdnc (voir en fin de chapitre).

En environnement de production, les logs des requêtes clientes seront séparés des logs du service lui-même.

2.2.1. Cache DNS

Par défaut, Bind est configuré pour faire office de serveur de proxy-cache DNS (mandataire). Un serveur proxy, ou mandataire, effectue une requête réseau au nom d'un client.

Lorsqu'il résoud pour la première fois une requête DNS, la réponse est stockée dans son cache pour la durée de son TTL (Time To Live) restant. Si le même enregistrement est à nouveau demandé par un autre client DNS, le traitement de la requête sera plus rapide, puisque la réponse sera disponible depuis le cache du serveur.

Chaque enregistrement DNS dispose d'un TTL lorsqu'il est fourni par le serveur qui fait autorité pour la zone. Ce TTL est décrémenté jusqu'à la fin de sa validité. Il est ensuite supprimé des caches.



Lorsque l'enregistrement est mis en cache, le TTL qui lui est associé est le TTL restant.

2.2.2. Récursivité

Un serveur est configuré par défaut pour répondre de manière récursive aux requêtes de ses clients.

Il interroge tour à tour les serveurs DNS nécessaires à la résolution d'une requête jusqu'à obtenir la réponse.

Un serveur non-récuratif délèguera la résolution du nom DNS à un autre serveur DNS.



Dans quel cadre utiliser un serveur non-récuratif ?

Lorsque la latence entre le serveur DNS et le reste du réseau est trop forte, ou quand le débit est limité, il peut être intéressant de configurer un serveur DNS en non-récuratif.

Ce sera par exemple le cas pour un théâtre d'opération ou un élément mobile d'une force.

Le serveur DNS local fera autorité sur la zone locale, mais déléguera la résolution des FQDN de l'intradef à un serveur en métropole, qui lui, prendra la requête en charge de manière récursive.

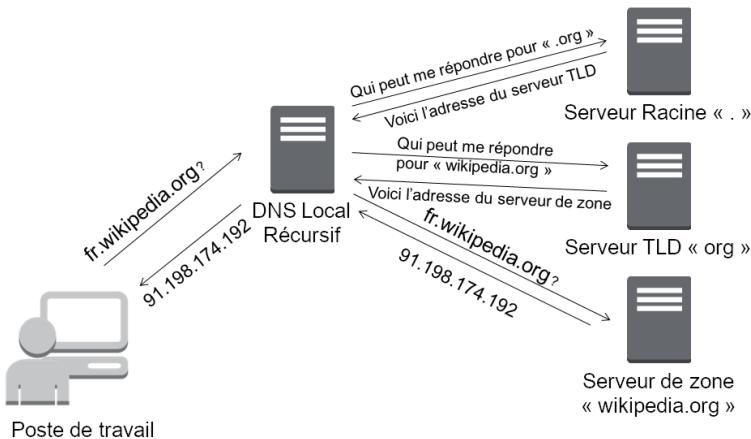


Figure 2.3. Le mode récursif

2.2.3. Architecture DNS

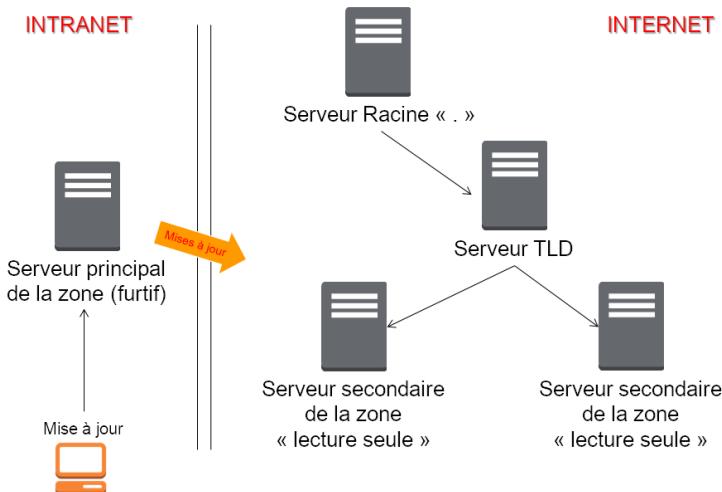
Un serveur DNS principal dispose d'une copie en écriture de la zone.

Il peut être intéressant de ne le rendre accessible que depuis le domaine local, et ne permettre l'interrogation des DNS depuis l'extérieur que vers les serveurs secondaires. D'un point de vue architectural cela lui évite ainsi les attaques ou les surcharges.

Le serveur est alors appelé serveur furtif.

Un serveur DNS n'est pas nécessairement le serveur maître de toutes les zones pour lesquels il fait autorité.

Un serveur DNS peut très bien être configurer pour être maître d'une zone et esclave d'une autre.

**Figure 2.4. Architecture avec serveur furtif**

La mise à jour du serveur se fait depuis le réseau local. Le serveur principal n'est pas accessible depuis l'extérieur.

La mise à jour se propage vers les serveurs secondaires.

L'interrogation par les clients internet ne se fait que sur les serveurs secondaires, ce qui protège le serveur principal des attaques par déni de service.

Pour un réseau complexe, il existe de nombreuses solutions architecturales de l'infrastructure DNS qu'il est important de bien étudier.

2.3. Configuration du serveur

2.3.1. Le fichier */etc/named.conf*

Ce fichier contient les paramètres de configuration du service DNS.

```
[root]# less /etc/named.conf
options {
    listen-on port 53 { 192.168.1.200; };
    directory "/var/named";
    allow-query { 192.168.1.0/24; };
};
```



Chaque ligne du fichier /etc/named.conf (même à l'intérieur des accolades) se termine par un point-virgule.

L'oubli de ce ";" est l'erreur la plus fréquente dans la configuration d'un serveur Bind.



Les noms, sous la forme FQDN (Fully Qualified Domain Name) doivent se terminer par ". ". En l'absence de ce ". ", Bind suffixera automatiquement avec le nom de domaine l'enregistrement.

Eg : www.formatux.lan → www.formatux.lan.formatux.lan.

La rubrique options contient la configuration générale du serveur BIND via différentes directives :

- listen-on : Définit l'interface, l'adresse et le port du service sur le serveur.
- directory : Définit le répertoire de travail de BIND, contenant les fichiers de zone.
- allow-query : Définit les hôtes autorisés à faire des requêtes sur le serveur. Par adresse IP ou réseau.

Permettre qu'un serveur DNS résolve les requêtes de n'importe quel client est une très mauvaise idée. Il faut au moins restreindre les droits aux réseaux locaux.

Il est possible, pour cela, de créer des ACL pour simplifier l'administration du fichier de configuration.

Le fichier de configuration contient également les informations relatives aux fichiers de zone.

```
[root]# less /etc/named.conf
zone "formatux.lan" IN {
    type master;
    file "masters/formatux.lan.direct";
    allow-update { 192.168.1.0/24; };
};

zone "1.168.192.in-addr.arpa" IN {
    type master;
```

```
file "masters/formatux.lan.inverse";
};
```

Les rubriques **zone** contiennent les configurations des zones de résolution de nom, inverses ou directes :

- type : Définit le type de serveur pour cette zone :
 - maître : possède la base de données en écriture
 - esclave : possède la base en lecture seule
 - forwarder : fait office de proxy-cache pour cette zone.
- file : Définit le chemin du fichier de zone.
- allow-update : Définit les hôtes ayant l'autorisation de mettre à jour les enregistrements DNS.

Les fichiers de zone inverse sont nommés en prenant l'adresse réseau de la zone (en inversant les octets) suivi du domaine in-addr.arpa.

2.3.2. Les rôles

Un serveur peut avoir le rôle de maître pour la zone, c'est-à-dire qu'il possède la zone en écriture.

```
[root]# less /etc/named.conf
zone "formatux.lan" IN {
    type master;
    file "masters/formatux.lan.direct";
    allow-update { 192.168.1.0/24; };
};
```

Seuls les clients figurant dans la variable allow-update pourront mettre à jour la base de données du DNS.

Un serveur peut également être un serveur secondaire (slave) pour la zone, c'est-à-dire qu'il possède la zone en lecture.

```
[root]# less /etc/named.conf
zone "formatux.lan" IN {
```

```
type slave;
file "slaves/formatux.lan.direct";
};
```

Un serveur peut enfin être expéditeur (forwarder) pour la zone, c'est-à-dire qu'il a connaissance de cette zone, et relaie les informations pour celle-ci.

```
[root]# less /etc/named.conf
zone "unautredomaine.fr" IN {
    type forwarder;
    forwarders {221.10.12.1};
};
```



Vous retrouverez cette notion sous Windows en tant que « redirecteur ».

2.4. Fichiers de zone



Les fichiers présents dans le répertoire /var/named doivent appartenir à l'utilisateur système named.

SELinux ne permettra pas l'enregistrement des fichiers de zone en dehors de ce répertoire.

Ces fichiers contiennent des enregistrements (RR : Resource Records) DNS de différents types. Ils permettent la résolution directe de noms (du nom vers l'adresse IP), ou la résolution inverse (de l'adresse IP vers le nom).

En plus de contenir les adresses IP et les noms des machines, les fichiers contiennent les paramètres de durée de vie des enregistrements (Time To Live, TTL).

Lorsqu'un enregistrement DNS est mis en cache, le temps restant sur son TTL est également conservé. À la fin du TTL, l'enregistrement est supprimé des caches.

- Un TTL plus long réduit les échanges DNS.
- Un TTL plus court permet une reconfiguration du réseau plus rapide.

2.4.1. Les types d'enregistrements

Tableau 2.1. Les types d'enregistrements

Type	Description
A	Nom attribué à une adresse de type IP V4
AAAA	Nom attribué à une adresse de type IP V6
CNAME	Alias d'un enregistrement A déjà défini Éviter de faire un alias vers un alias
MX	Serveur de messagerie destinataire pour la zone concernée
NS	Le ou les serveurs de noms de la zone (type A)
PTR	Enregistrement pointeur pour une zone inverse
SOA	Démarre la configuration (cf: diapos suivantes)
SVR	Service (protocole jabber,...)
TXT	Informations

- Champ MX : Le numéro précise la priorité, la plus faible étant la plus prioritaire. Ceci permet de définir un ou plusieurs serveurs de secours qui stockeront les mails en attendant le retour du serveur principal.
- Champ de type A : Enregistrement standard. Attribue un nom à une adresse IP.

Plusieurs enregistrements identiques de type A vers des adresses différentes permet de faire de l'équilibrage de charge par round-robin (RR).

Exemple :

```
mail A 192.168.1.10
      A 192.168.1.11
```

- Champ AAAA : On utilise quatre A pour symboliser IPv6 car une adresse IPv6 est codée sur 16 octets, soit 4 fois plus qu'une adresse IPv4.
- CNAME : Permet d'attribuer un ou plusieurs alias à un enregistrement A déjà défini. Plusieurs enregistrements du même alias permettent également de faire de l'équilibrage de charge type RR.



On trouvera des enregistrements typiques, comme autoconfig, qui permet le mécanisme de configuration automatique d'un client de messagerie.

2.4.2. Fichier de zone directe

Ce fichier est nécessaire au fonctionnement du système DNS. C'est par lui que se fait la résolution d'un nom en adresse IP.

```
[root]# less /var/named/formatux.lan.direct
$ORIGIN .
$TTL 3600
formatux.lan. SOA inf1-formatux.formatux.lan. contact.formatux.lan.
(123; 14400; 3600; 604800; 3600; )

@ IN NS inf1-formatux.formatux.lan.
poste1 IN A 192.168.1.10
inf1-formatux IN A 192.168.1.200
formatux.lan. MX 10 192.168.1.201
inf3-formatux IN A 192.168.1.202
www IN CNAME inf1-formatux.formatux.lan.
```

- \$ORIGIN : Définit la valeur par défaut du domaine courant pour les renseignements du fichier. Un . signifie la racine.
- \$TTL : Durée de vie par défaut des enregistrements de la zone dans le cache, exprimée en secondes. Le TTL peut également être précisé enregistrement par enregistrement.
- SOA : Start Of Authority. La ligne démarre la configuration d'une zone. Définit :
 - le nom du serveur maître principal,
 - l'email de l'administrateur de la zone (un . remplace le @ de l'adresse mail).
 - Entre parenthèses, le numéro de série du fichier (incrémenté à chaque mise à jour) et les délais de mise à jour ou de rafraîchissement, exprimés en secondes.
 - Numéro de zone : Numéro incrémental (voir le paragraphe suivant)
 - Rafraîchissement : Durée en secondes avant une tentative de synchronisation avec le serveur maître

- Réitération : Intervalle de temps avant réitération si l'essai précédent n'a pas fonctionné
- Expiration : Durée en secondes avant l'expiration car le serveur maître est injoignable
- Cache négatif (TTL) : Durée de vie en secondes des enregistrements



Le @ a une signification particulière pour Bind. Il se représente lui même, raison pour laquelle le @ de l'adresse courriel d'administration est remplacée par un .

Le numéro de la zone sert à identifier la dernière modification du DNS maître. Tous les serveurs secondaires utilisent ce numéro pour savoir s'ils doivent se synchroniser.

Il existe deux méthodes d'incrémentation du numéro de zone :

- Incrémentationale : 1, puis 2, puis 3 (pourquoi pas ?)
- Basée sur la date : AAAAMMMJJXX, qui nous donne par exemple, pour la première modification du jour : 2017210101 (méthode à privilégier)

2.4.3. Le fichier de zone inverse

Bien que non obligatoire, ce fichier est fortement conseillé pour un fonctionnement optimal du système DNS. C'est par lui que se fait la résolution d'une adresse IP en nom.



Des services comme SSH s'appuie sur la résolution inverse.

```
[root]# more /var/named/formatux.lan.inverse
$ORIGIN 1.168.192.in-addr.arpa.
$TTL 259200
@ SOA inf1-formatux.formatux.lan. contact.formatux.lan.
( 123; 14400; 3600; 604800; 3600; )
@ NS inf1-formatux.formatux.lan.
1 0 PTR poste1.formatux.lan.
200 PTR inf1-formatux.formatux.lan.
```

2.4.4. La commande *nsupdate*



L'usage de la commande nsupdate est exclusive. Il ne faut plus modifier les fichiers de zone manuellement, sous peine de pertes d'informations.

Syntaxe :

```
nsupdate
```

Exemple:

```
[root]# nsupdate
> server 192.168.1.200
> zone formatux.lan
> update add poste2.formatux.lan 3600 A 192.168.1.11
> update delete poste1
> send
```

La commande nsupdate est interactive.

À la saisie, elle ouvre un prompt dans lequel il faut saisir les requêtes de mise à jour du fichier de zone.

Ces requêtes peuvent être :

- **server** : Précise le serveur BIND pour lequel les requêtes seront envoyées.
- **zone** : Précise la zone de résolution pour laquelle les requêtes seront envoyées.
- **prereq yxdomain nom** : L'existence de l'enregistrement nom est une condition de mise à jour.
- **update add nom TTL type @IP** : Ajoute l'enregistrement nom, en précisant son type, son adresse IP et son TTL.
- **update delete nom** : Supprime l'enregistrement nom.
- **send** : Valide et envoie les requêtes.

2.4.5. La commande *rndc*

La commande **rndc** permet de manipuler le serveur DNS à chaud.

Syntaxe de la commande rndc.

```
rndc reload  
rndc querylog on|off
```

- **reload** : Prend en compte les modifications apportées sans devoir relancer le service
- **querylog** : Active ou non la journalisation

Après modification d'un fichier de zone, il est nécessaire de faire prendre en compte les modifications au service. Les fichiers étant lus au démarrage du service, cela permet de prendre en compte les modifications, mais résulte en la perte des statistiques. Il faut donc privilégier la méthode reload.

2.4.6. Le suivi des logs

La fonction d'enregistrement des fichiers journaux est activée ou désactivée par la commande rndc.

Le fichier de logs est par défaut /var/named/data/named.run

Exemple :

```
[root]# rndc querylog on  
[root]# tail -f /var/named/data/named.run
```

Bind propose dans son fichier de configuration des options pour journaliser les informations :

- de transferts,
- de requêtes clients,
- d'erreurs,
- ...

2.5. Configuration du client

NetworkManager est un outil de gestion du réseau. Sur un serveur dont le réseau est défini par cet outil, la configuration cliente de Bind est décrite dans le fichier de l'interface.

```
[root]#less /etc/sysconfig/network-scripts/ifcfg-ethX
DOMAIN="formatux.lan"
DNS1=192.168.1.200
DNS2=192.168.1.201
```

NetworkManager modifiera lui-même le fichier `/etc/resolv.conf` à chaque relance du service réseau.

Avant de lancer une recherche DNS, le logiciel client vérifiera si la requête porte sur un FQDN ou non. Si le nom n'est pas pleinement qualifié, le client suffixera la requête avec le premier suffixe DNS fourni.

Si la résolution est impossible, le client émettra une nouvelle requête avec le suffixe suivant, ainsi de suite jusqu'à l'obtention d'une réponse.

Par exemple, il est possible de fournir deux suffixes :

```
formatux.fr
formatux.lan
```

Lors d'une requête DNS portant sur, par exemple, portail, une première requête `portail.formatux.fr` sera faite. En l'absence de réponse positive, une seconde requête sera effectuée portant sur `portail.formatux.lan`. Pour éviter ce phénomène d'interrogation multiple, il est préférable de fournir une adresse pleinement qualifiée.



Veiller à spécifier au moins deux serveurs DNS pour assurer une redondance en cas de panne du serveur principal.

Sans outil de gestion du réseau, la configuration cliente de Bind est décrite dans le fichier `/etc/resolv.conf`.

```
[root]# less /etc/resolv.conf
search "formatux.lan"
nameserver 192.168.1.200
nameserver 192.168.1.201
```



NetworkManager, si actif, écrasera les valeurs entrées manuellement dans ce fichier.

L'utilitaire **system-config-network-tui** (**nmtui** sous CentOS 7) permet une configuration graphique correcte du réseau par une interface ncurses.

2.5.1. La commande dig

La commande dig (Domain Information Groper) permet d'interroger des serveurs DNS.



Dig doit être privilégié par rapport à NSLookup qui n'est plus maintenue.

Syntaxe de la commande dig.

```
dig [name] [type] [options]
```

Exemple :

```
[root]# dig centos65.formatux.lan A
...
;; QUESTION SECTION:
; centos65.formatux.lan. IN A

;; ANSWER SECTION:
centos65.formatux.lan. 86400 IN A 192.168.253.131

;; AUTHORITY SECTION:
formatux.lan. 86400 IN NS centos65.formatux.lan.
...
```

```
[root]# dig -t MX linux.fr
```

```
[root]# dig linux.fr MX +short
```

2.5.2. **Mise en cache côté client**

Le service NSCD est responsable de la mise en cache des requêtes réseaux type LDAP ou DNS.

Pour profiter de la mise en cache local, il faudra veiller à ce que le service NSCD soit démarré.

```
[root]# service nscd start  
[root]# chkconfig nscd on
```

Nscd n'est pas installé par défaut sur les RHEL 6.

2.5.3. **Mise à jour dynamique**

Les clients peuvent s'enregistrer dynamiquement sur le serveur DNS, ce qui est intéressant dans le cadre d'une attribution de l'adressage IP dynamique avec DHCP.

2.6. Configuration du pare-feu serveur

Les règles iptables à configurer en tcp et udp sont les suivantes :

```
[root]# vi /etc/sysconfig/iptables  
# Autoriser DNS  
iptables -t filter -A INPUT -p tcp -dport 53 -j ACCEPT  
iptables -t filter -A INPUT -p udp -dport 53 -j ACCEPT
```



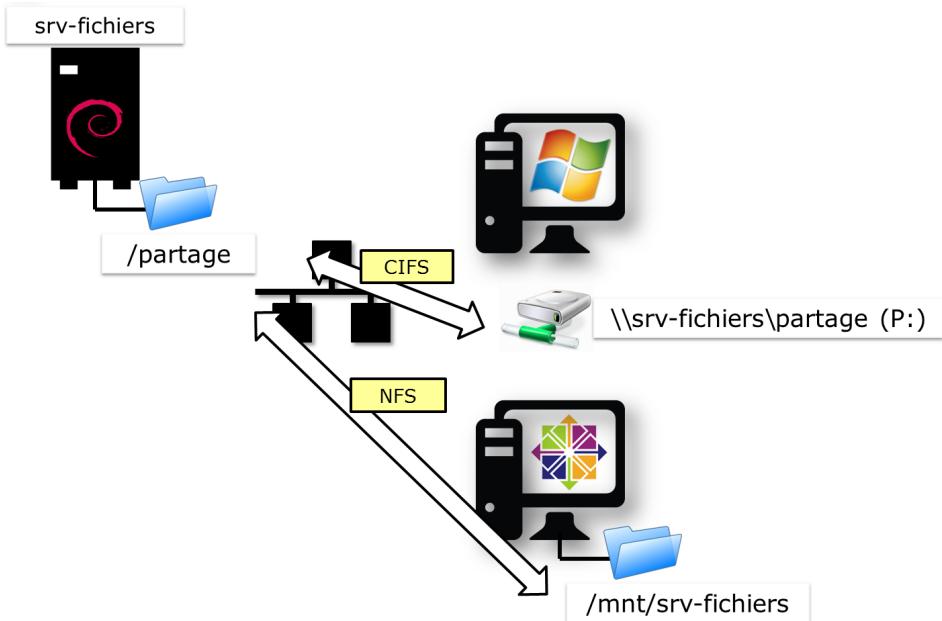
system-config-firewall-tui est l'outil graphique permettant de configurer le pare-feu.

3

Serveur de fichiers Samba

Samba est un serveur de fichiers permettant l'interopérabilité entre divers systèmes, notamment les systèmes Linux et Microsoft. Il permet à des systèmes Linux de créer des partages utilisables par des machines Windows et vice-versa.

Le projet Samba a été initié dès 1992 sous licence GPL (et donc gratuit).



Un même dossier partagé par NFS et par Samba est ainsi accessible depuis toutes les plateformes clientes.

3.1. Le protocole SMB

Le protocole SMB (Server Message Block) était une extension de Microsoft pour permettre la redirection des entrées/sorties vers NetBIOS (Network Basic Input/Output System).

SMB permettait :

- le transfert de données entre machines Windows : partages de fichiers, impressions et messagerie électronique ;
- le parcours du voisinage réseau (browsing) ;
- la résolution de noms NetBIOS en adresses IP (Windows Internet Name Server) ;
- l'authentification centralisée (notion de domaine).

Le protocole NetBIOS était une interface permettant la mise en place de noms de machines, de groupes de travail, de domaines, etc. Il faisait fonctionner le voisinage réseau jusqu'à Windows 2000, mais son mode de fonctionnement induisait une charge réseau importante.

NetBIOS était le système de noms des réseaux SMB comme l'est aujourd'hui le service DNS.

Les diffusions NetBIOS ne passant pas les routeurs, la notion de voisinage réseau désigne l'ensemble des stations de travail utilisant le protocole NetBIOS sur un même segment de réseau IP. Le **maître explorateur** (master browser) est le poste client ou serveur tenant à jour la liste des ordinateurs utilisés par le service de voisinage réseau.

3.2. Le protocole CIFS

Les améliorations apportées au protocole SMB ont permis de faire aboutir la suite de protocoles clients/serveurs CIFS (Common Internet File System) que le service Samba implémente.

3.3. Installation de Samba

```
[root]# yum install samba smbclient
```

La partie serveur de Samba est basée essentiellement sur 2 démons :

- Le démon **smb** est le serveur SMB : il répond aux requêtes des clients lorsque ceux-ci accèdent aux partages définis et il est le seul à accéder aux systèmes de fichiers Linux.
- Le démon **nmb** est le serveur de noms NetBIOS essentiel au fonctionnement de SMB. Il peut être configuré comme serveur WINS.
- Le fichier de configuration principal est **smb.conf**. C'est dans ce fichier que sont définis les paramètres de fonctionnement des 2 démons, ainsi que la définition des exports de ressources.

La partie cliente de samba (samba-client) contient les outils qui permettent le montage et le parcours des ressources Samba.

Le paquet **samba-client** fournit les binaires :

- **findsmb** : afficher de nombreuses informations sur les stations d'un sous-réseau qui répondent aux requêtes de noms SMB.
- **nmblookup** : interroger le protocole NetBIOS et associe les noms Netbios à des adresses IP.
- **sharesec** : manipuler les droits de partages de fichiers
- **smbcacls** : manipuler les ACL NT sur les partages de fichiers
- **smbclient** : offrir une interface FTP Like pour accéder à des partages de fichiers
- **smbget** : télécharger des fichiers depuis un partage windows
- **smbspool** : envoyer une impression à un serveur d'impression
- **smbtree** : fournir un explorateur de fichier en mode texte similaire au "Voisinage réseau" des ordinateurs Windows. Il affiche un arbre des domaines connus, desserveurs et des partages accessibles depuis les serveurs.
- ...

Le paquet **samba-common** fournit les binaires :

- **net** : offrir les mêmes fonctionnalités que la commande net du monde Windows.

- **pdbedit** : gérer de la base SAM
- **smbcquotas** : manipuler les quotas NT d'un partage de fichiers
- **smbpasswd** : changer le mot de passe des utilisateurs
- **testparm** : tester la syntaxe d'un fichier de configuration smb.conf
- ...

```
[root]# chkconfig smb on
[root]# chkconfig nmb on
[root]# service smb start
[root]# service nmb start
```

3.4. Sécurité SELinux

Par défaut, la sécurité SELinux est active. Pour vérifier le contexte de sécurité en place sur des fichiers, il faut utiliser la commande :

```
[root]# ls -Zd /export/
```

Le contexte de sécurité *samba_share_t* doit être positionné sur les dossiers partagés :

```
[root]# chcon -R -t samba_share_t /export/
```

Plus d'information sur la sécurité SELinux et Samba [sur le site de RedHat¹](https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/6/html/Managing_Confined_Services/sect-Managing_Confined_Services-Samba-Booleans.html).

Pensez à stopper le parefeu ou à le configurer dans le fichier */etc/sysconfig/iptables* :

```
-A INPUT -m state --state NEW -m udp -p udp --dport 137 -j ACCEPT
-A INPUT -m state --state NEW -m tcp -p tcp --dport 137 -j ACCEPT
-A INPUT -m state --state NEW -m udp -p udp --dport 138 -j ACCEPT
-A INPUT -m state --state NEW -m udp -p udp --dport 139 -j ACCEPT
-A INPUT -m state --state NEW -m udp -p udp --dport 445 -j ACCEPT
-A INPUT -m state --state NEW -m tcp -p tcp --dport 445 -j ACCEPT
```

¹ https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/6/html/Managing_Confined_Services/sect-Managing_Confined_Services-Samba-Booleans.html

Pour pouvoir utiliser Samba comme contrôleur de domaine et utiliser les commandes useradd et groupadd, il faudra mettre le booléen **samba_domain_controller** à on.

```
[root]# setsebool -P samba_domain_controller on
```

Pour rappel, il est possible d'obtenir tous les booléens SELinux concernant Samba avec la commande suivante :

```
[root]# getsebool -a | grep "samba"
samba_create_home_dirs --> off
samba_domain_controller --> off
samba_enable_home_dirs --> off
samba_export_all_ro --> off
samba_export_all_rw --> off
samba_portmapper --> off
samba_run_unconfined --> off
samba_share_fusefs --> off
samba_share_nfs --> off
sanlock_use_samba --> off
use_samba_home_dirs --> off
virt_use_samba --> off
```

Pour autoriser les partages via Samba des répertoires de connexions des utilisateurs, il faudra positionner le booléen **samba_enable_home_dirs** également à on.

```
[root]# setsebool -P samba_enable_home_dirs on
```

3.5. La configuration de SAMBA

La partie serveur de Samba est basée sur 1 seul fichier de configuration **/etc/samba/smb.conf** qui définit les paramètres de fonctionnement des 2 démons et des exports de ressources.

Chaque paramètre fait l'objet d'une ligne au format :

```
nom = valeur
```

Les commentaires commencent par un ';' ou un '#' et se termine à la fin de la ligne.

Le caractère '\' permet de scinder une ligne logique sur plusieurs lignes physiques.

Ce fichier est constitué de 3 sections spéciales :

- **[global]** : paramètres généraux ;
- **[homes]** : paramètres des répertoires utilisateurs ;
- **[printers]** : paramètres des imprimantes.

Les autres sections, déclarées entre '[]' sont des déclarations de partage.

3.5.1. Les niveaux de sécurité

Il existe cinq niveaux de sécurité (option security), mais un seul peut être appliqué par serveur :

- **share** : le niveau dit de partage, lié à une ressource. Un mot de passe est associé à chaque partage (Déprécié : ne plus utiliser) ;
- **user** : le niveau de sécurité de l'utilisateur, lié à son authentification. C'est le niveau recommandé et par défaut ;
- **server** : l'authentification est réalisée par un autre serveur (Déprécié : ne plus utiliser) ;
- **domain** : l'authentification est réalisée par un autre serveur, mais le serveur Samba doit être membre du domaine ;
- **ads** : l'authentification est réalisée par un serveur Active Directory.

3.5.2. Les variables internes à Samba

Tableau 3.1. Les variables internes à Samba

Variable	Observation
%a	Architecture du client
%l	Adresse IP du client
%M	Nom dns du client
%m	Nom NetBios du client

Variable	Observation
%u	Identité de l'utilisateur pour le partage concerné
%U	Identité souhaitée par l'utilisateur du partage
%H	Répertoire de connexion de l'utilisateur
%u%g ou %G	Groupe principal de l'utilisateur
%u ou %U	Nom du partage
%S	
%P	Répertoire racine du partage concerné
%d	PID du processus courant
%h	Nom DNS du serveur Samba
%L	Nom NetBIOS du serveur Samba
%v	Version de samba
%T	Date et heure système
%%\$var	valeur de la variable d'environnement var

3.5.3. La commande testparm

La commande **testparm** teste la validité du fichier /etc/samba/smb.conf.

L'option -v affiche tous les paramètres applicables.

```
[root]# testparm
Load smb config files from /etc/samba/smb.conf
...
Loaded services file OK.
Server role : ROLE_STANDALONE
...
```

Sans option, cette commande renvoie la configuration du serveur samba sans les commentaires et omet les paramètres positionnés à leur valeur par défaut, ce qui facilite sa lecture.

```
[root]# testparm
```

3.5.4. La section [global]

Tableau 3.2. Les directives de la section [global]

Directive	Exemple	Explication
workgroup	workgroup = FORMATUX	Définir le groupe de travail ou le nom de domaine NetBIOS. (À mettre en majuscule).
netbios name	netbios name = inf1-formatux	Nom NetBIOS de la station Maximum 15 caractères Pas de rapport direct avec le nom de la machine Linux
server string	server string = Samba version %v	Description du serveur apparaissant dans l'explorateur Windows
hosts allow	hosts allow = 127.172.16.1.	Permet de restreindre les clients du serveur aux seuls réseaux mentionnés. Notez la présence d'un point à la fin de l'adresse et l'absence du 0.
log file	log file = /var/log/samba/log.%m	Enregistrer les évènements dans un fichier %m représente le nom NetBIOS du client
security	security = user	Modèle de sécurité du serveur
passdb backend	passdb backend = tdbsam	Stockage des utilisateurs et des mots de passe. Le format tdbsam (Trivial

Directive	Exemple	Explication
		Database) est le format par défaut, limité en performance à 250 utilisateurs. Au delà, il faudra passer au format ldapsam et stocker les utilisateurs et les groupes dans une base LDAP.

3.5.5. La section [homes]

La section [homes] contient la configuration des partages utilisateurs.

C'est une section réservée par Samba, qui lui applique un fonctionnement très particulier. Ce nom de partage ne doit pas être utilisé pour un autre partage ni modifié.

```
[homes]
comment = Home Directories
browseable = no
writable = yes
```

Tous les utilisateurs verront le même partage "homes" mais le contenu sera personnalisé pour chacun.

Attention à bien configurer les booléens SELinux pour autoriser le partage des dossiers personnels.

3.5.6. La section [printers]

La section [printers] contient la configuration du serveur d'impression.

```
[printers]
comment = All Printers
browseable = no
writable = yes
guest ok = no
```

```
printable = yes
```

Samba peut ainsi faire office de serveur d'impressions, ce qui est une fonctionnalité intéressante (ne nécessite pas l'acquisition de licences clientes).

3.5.7. Partages personnalisés

Avant de paramétriser une nouvelle section du fichier smb.conf qui correspondra à un nouveau partage, il convient de se poser quelques questions :

- Quel est le chemin du partage ?
- Qui peut modifier le contenu ?
- Le partage doit-il être visible sur le réseau ou au contraire sera-t-il masqué ?
- Y aura-t-il un accès anonyme ?

Un nouveau partage est représenté par une section [nomdupartage] dans le fichier smb.conf. En voici un exemple :

```
[partage]
comment = Partage
browseable = yes
writable = yes
path = /export/data
valid users = @users
read list = georges
write list = bob, alice
invalid users = maurice
create mask = 0664
directory mask = 0775
force group = users
```

De nombreuses directives sont disponibles pour configurer les partages :

Directive	Exemple	Explication
comment	comment = Exemple de partage	Affiche un commentaire dans l'explorateur de fichiers.
browsable	browsable = yes	Affiche le partage dans le voisinage réseau.

Directive	Exemple	Explication
writeable	writeable = yes	Le partage est en lecture seule ou en écriture.
path	path = /export/data	Le chemin absolu à partager sur le réseau. Attention au contexte SELinux de ce dossier.
valid users	valid users = @users	Liste les utilisateurs ou les groupes autorisés à accéder au partage.
invalid users	invalid users = alice	Liste les utilisateurs ou les groupes qui ne sont pas autorisés à accéder au partage.
read list	read list = bob	Liste les utilisateurs ou les groupes autorisés à accéder au partage en lecture.
write list	write list = patrick, alain	Liste les utilisateurs ou les groupes autorisés à accéder au partage en écriture.
create mask	create mask = 0664	Les fichiers créés prendront les droits spécifiés.
directory mask	directory mask = 0775	Les dossiers créés prendront les droits spécifiés.
force group	force group = users	Les nouveaux fichiers et dossiers appartiendront au groupe spécifié. Dans ce cas, il n'y a pas d'@ devant le groupe !

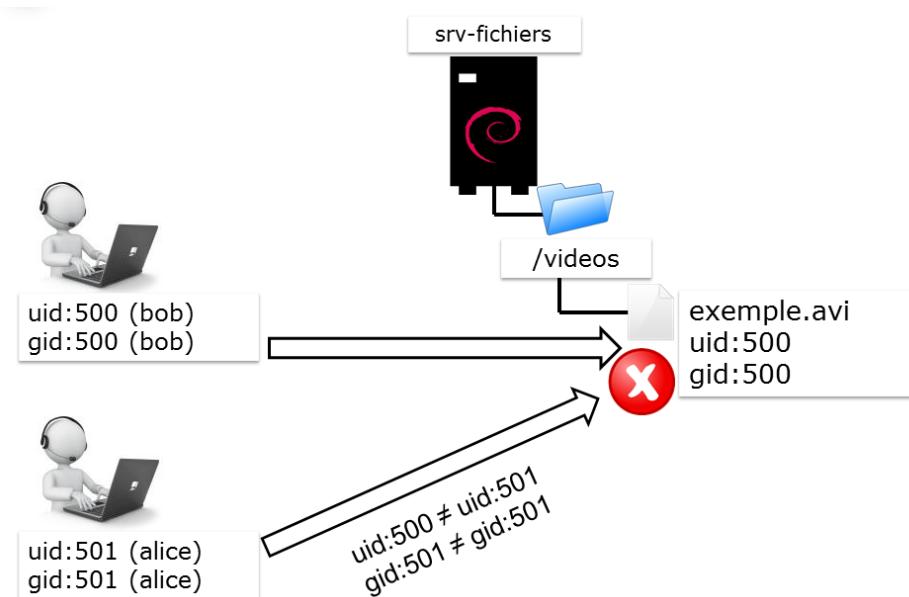
3.5.8. La directive **force group**

La directive **force group** permet de forcer l'appartenance d'un fichier créé à un groupe spécifique.

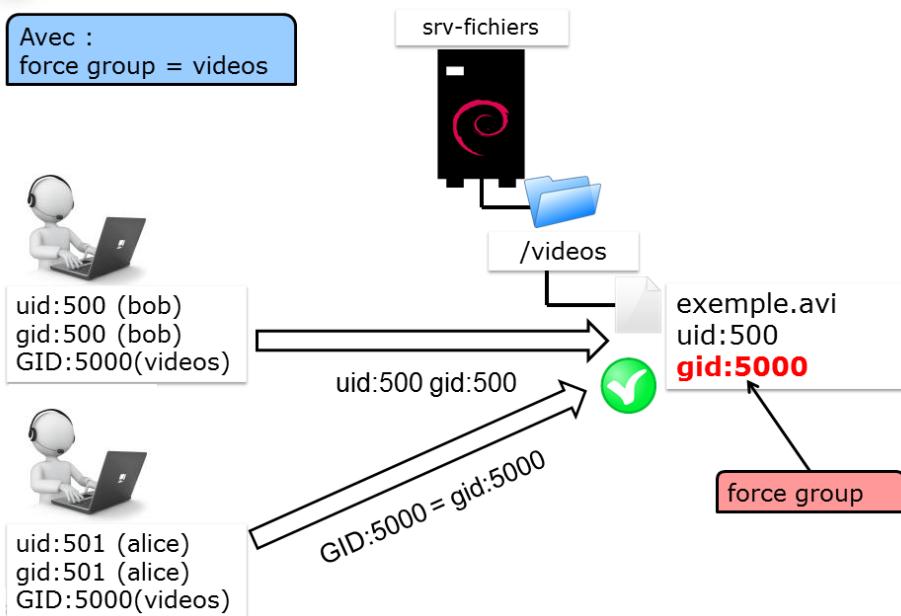
Cette directive est essentielle dans le fonctionnement de Samba, puisqu'elle assure que, quelque soit le groupe principal d'un utilisateur, celui-ci sera autorisé à accéder à un fichier sur le partage s'il fait parti, en tant qu'invité, du groupe spécifié dans la directive.

Les deux exemples ci-dessous mettent en avant ce mécanisme :

Utilisation sans le mécanisme **force group** :



Utilisation avec le mécanisme **force group** :



3.6. Commandes d'administration

3.6.1. La commande pdbedit

La commande pdbedit permet de gérer la base SAM des utilisateurs Samba, que le backend soit au format tdbsam ou ldapsam, contrairement à la commande smbpasswd qui est limitée au format tdbsam.

Syntaxe de la commande pdbedit.

```
pdbedit [-a|-r|-x|-L] [-u username] ...
```

Exemple :

```
[root]# pdbedit -L
stagiaire:1000:Stagiaire SYS
```

Tableau 3.3. Options principales de la commande pdbedit

Option	Observation
-a	Ajouter un utilisateur

Option	Observation
-r	Modifier un utilisateur
-x	Supprimer un utilisateur
-L	Lister les utilisateurs
-u	Spécifier le nom de l'utilisateur pour les options -a, -r, et -x

Ajouter un utilisateur

Le format de cryptage du mot de passe entre le monde Microsoft et le monde Linux étant différent, Samba doit soit tenir à jour une base de données contenant les mots de passe au bon format ou déléguer cette gestion au serveur LDAP, ce qui explique l'usage de la commande `smbpasswd`.

```
pdbeedit -a -u username [-f description]
```

Exemple :

```
[root]# pdbeedit -a -u bob -f "Bob Leponge"
Unix username:      bob
User SID:          S-1-5-21-3024208064-2128810558-4043545969-1000
Full Name:         Bob Leponge
Home Directory:   \\srvfichiers\bob
Domain:           SRVFICHIERS
...
```

Option	Observation
-a	Ajouter un utilisateur. L'utilisateur doit exister dans le fichier /etc/passwd.
-u	Spécifier le nom de l'utilisateur à ajouter.

3.6.2. La commande `smbpasswd`

La commande `smbpasswd` permet de gérer les mots de passe des utilisateurs Samba.

Syntaxe de la commande `smbpasswd`.

```
smbpasswd [-d|-e] username
```

Exemple :

```
[root]# smbpasswd bob
New SMB password:
Retype new SMB password:
```

Option	Observation
-e	Réactive un compte.
-d	Désactive un compte.

Il est possible de synchroniser les mots de passe Unix et Samba :

```
[global]
    unix password sync = yes
    obey pam restrictions = yes
```

Directive	Exemple	Explication
unix password sync	unix password sync = yes	Synchronise le mot de passe entre le compte unix et le compte samba. Fonctionne uniquement avec la commande smbpasswd. La directive n'est pas prise en compte par la commande tdbsql.
obey pam restrictions	obey pam restrictions = yes	Applique les restrictions PAM.

3.6.3. La commande smbclient

La commande **smbclient** permet d'accéder à des ressources Windows (ou Samba) depuis le monde Unix.

Syntaxe de la commande smbclient.

```
smbclient '//serveur/partage' -U utilisateur
```

Exemple :

```
[root]# smbclient \\\\stat-wind\\\\partage-wind -U alain  
smb: |> help
```

ou :

```
[root]# smbclient '//stat-wind/partage-wind' -U alain  
smb: |> help
```

Le programme smbclient est couramment utilisé pour créer un interpréteur de type 'ftp' permettant ainsi d'accéder à des ressources SMB réseau.

Lister les partages :

```
[root]# smbclient -L inf1-formatux
```

Se connecter à un partage data du serveur inf1-formatux avec l'utilisateur bob :

```
[root]# smbclient -L //inf1-formatux/data -U bob  
Enter bob's password:
```

4

Serveur web Apache

Le serveur **HTTP Apache** est le fruit du travail d'un groupe de volontaires : The Apache Group. Ce groupe a voulu réaliser un serveur Web du même niveau que les produits commerciaux mais sous forme de **logiciel libre** (son code source est disponible).

L'équipe d'origine a été rejoints par des centaines d'utilisateurs qui, par leurs idées, leurs tests et leurs lignes de code, ont contribué à faire d'Apache le plus utilisé des serveurs Web du monde.

L'ancêtre d'Apache est le serveur libre développé par le National Center for Supercomputing Applications de l'université de l'Illinois. L'évolution de ce serveur s'est arrêtée lorsque le responsable a quitté le NCSA en 1994. Les utilisateurs ont continué à corriger les bugs et à créer des extensions qu'ils distribuaient sous forme de "patches" d'où le nom "a patchee server".

La version 1.0 de Apache a été disponible le 1 décembre 1995 (il y a plus de 20 ans !).

L'équipe de développement se coordonne par l'intermédiaire d'une liste de diffusion dans laquelle sont proposées les modifications et discutées les évolutions à apporter au logiciel. Les changements sont soumis à un vote avant d'être intégrés au projet. Tout le monde peut rejoindre l'équipe de développement, il suffit de contribuer activement au projet pour pouvoir être nommé membre de The Apache Group.

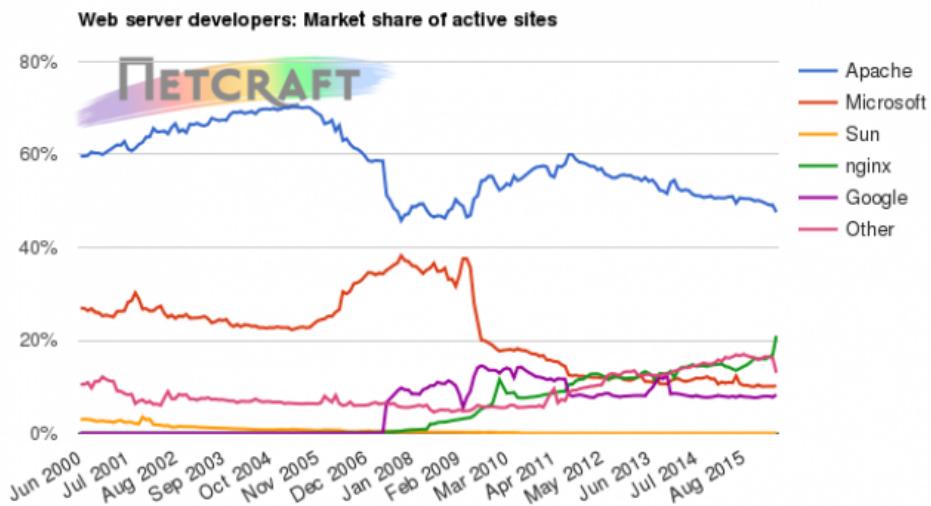


Figure 4.1. Statistiques NetCraft : Market Share of Active Sites

Le serveur Apache est très présent sur l'Internet, puisqu'il représente encore environ 50% des parts de marché pour l'ensemble des sites actifs.

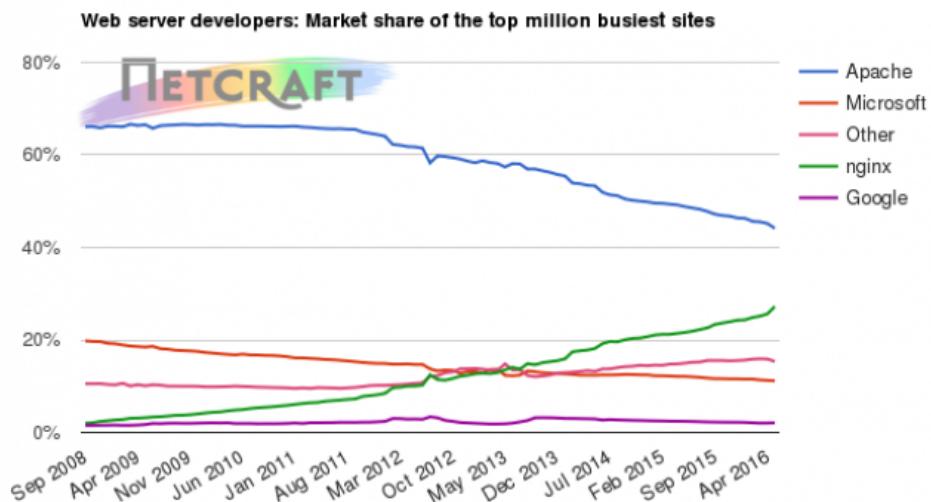


Figure 4.2. Statistiques NetCraft : Top million busiest sites

Les parts de marché perdues par Apache sont prises par son plus grand challenger : le serveur nginx. Ce dernier, plus rapide pour délivrer les pages web, et moins complets fonctionnellement parlant que le géant Apache.

4.1. Le protocole HTTP

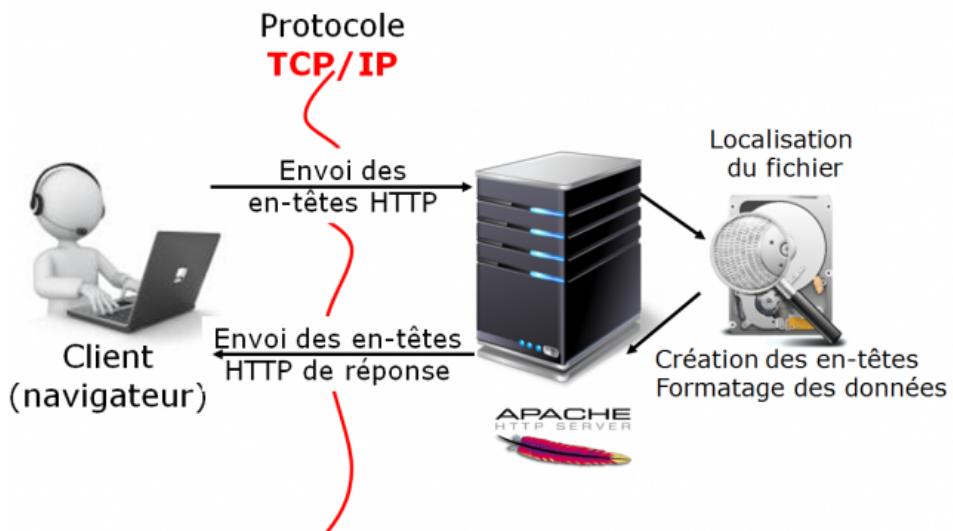
Le protocole **HTTP** (HyperText Transfer Protocol) est le protocole le plus utilisé sur Internet depuis 1990.

Ce protocole permet un transfert de fichiers (essentiellement au format HTML, mais aussi au format CSS, JS, AVI...) localisés grâce à une chaîne de caractères appelée URL entre un navigateur (le client) et un serveur Web (appelé d'ailleurs httpd sur les machines UNIX).

HTTP est un protocole "requête - réponse" opérant au dessus de TCP (Transmission Control Protocol).

1. Le client ouvre une connexion TCP vers le serveur et envoie une requête.
2. Le serveur analyse la requête et répond en fonction de sa configuration.

Le protocole HTTP est en lui même dit "**STATELESS**" : il ne conserve pas d'information sur l'état du client d'une requête à l'autre. Ce sont les langages dynamiques comme le php, le python ou le java qui vont permettre la conservation en mémoire des informations de session d'un client (comme dans le cadre d'un site de e-commerce par exemple).



Le protocole HTTP est en version 1.1. La version 2 est en cours de déploiement.

Une réponse HTTP est un ensemble de lignes envoyées au navigateur par le serveur. Elle comprend :

- **Une ligne de statut** : c'est une ligne précisant la version du protocole utilisé et l'état du traitement de la requête à l'aide d'un code et d'un texte explicatif. La ligne comprend trois éléments devant être séparés par un espace :
 - La version du protocole utilisé ;
 - Le code de statut ;
 - La signification du code .
- **Les champs d'en-tête de la réponse** : il s'agit d'un ensemble de lignes facultatives permettant de donner des informations supplémentaires sur la réponse et/ou le serveur. Chacune de ces lignes est composée d'un nom qualifiant le type d'en-tête, suivi de deux points (:) et de la valeur de l'en-tête.
- **Le corps de la réponse** : il contient le document demandé.

Voici un exemple de réponse HTTP :

Exemple de réponse HTTP.

```
HTTP/1.1 200 OK
Date : Sat, 15 Jan 2016 14:37:12 GMT Server : Apache/2.17
Content-Type : text/HTML
Content-Length : 1245
Last-Modified : Fri, 14 Jan 2016 08:25:13 GMT
```

Le rôle du serveur web consiste à traduire une URL en ressource locale. Consulter la page <http://www.free.fr/>, revient à envoyez une requête HTTP à cette machine. Le service DNS joue donc un rôle essentiel.

4.1.1. Les URL

Une URL (**Uniform Ressource Locator** - littéralement “identifiant uniforme de ressources”) est une chaîne de caractères ASCII utilisée pour désigner les ressources sur Internet. Elle est informellement appelée adresse web.

Une URL est divisée en trois parties :

Composition d'une URL.

```
<protocole>://<hôte>:<port>/<chemin>
```

- **Le nom du protocole** : il s'agit du langage utilisé pour communiquer sur le réseau. Le protocole le plus utilisé est le protocole HTTP (HyperText TransferProtocol), le protocole permettant d'échanger des pages Web au format HTML. De nombreux autres protocoles sont toutefois utilisables.
- **Identifiant et mot de passe** : permet de spécifier les paramètres d'accès à un serveur sécurisé. Cette option est déconseillée car le mot de passe est visible dans l'URL (dans le cadre de la sécurité).
- **L'hôte** : Il s'agit du nom de l'ordinateur hébergeant la ressource demandée. Notez qu'il est possible d'utiliser l'adresse IP du serveur, ce qui rend par contre l'URL moins lisible.
- **Le numéro de port** : il s'agit d'un numéro associé à un service permettant au serveur de savoir quel type de ressource est demandé. Le port associé par défaut au protocole est le port numéro 80. Ainsi, lorsque le service Web du serveur est associé au numéro de port 80, le numéro de port est facultatif.
- **Le chemin d'accès à la ressource** : Cette dernière partie permet au serveur de connaître l'emplacement auquel la ressource est située, c'est-à-dire de manière générale l'emplacement (répertoire) et le nom du fichier demandé. Si non renseignée, indique la première page de l'hôte. Sinon indique le chemin de la page à afficher.

4.1.2. Les ports

Une requête HTTP arrivera sur le port 80 (port par défaut pour http) du serveur fonctionnant sur l'hôte. L'administrateur peut toutefois choisir librement le port d'écoute du serveur.

Le protocole http se décline en une version sécurisée: le protocole https (port 443). Ce protocole chiffré s'implémente à partir du module mod-ssl.

D'autres ports peuvent être utilisés, comme le port 8080 (serveurs d'applications Java EE) ou le port 10 000 (Serveur webmin).

4.2. Installation du serveur

Apache est **multiplateforme**. Il peut être utilisé sur Linux, Windows, Mac...

L'administrateur devra choisir entre deux méthodes d'installation :

- **Installation par paquets** : l'éditeur de la distribution fourni des versions **stables et soutenues** (mais parfois anciennes) ;
- **Installation depuis les sources** : le logiciel apache est compilé, l'administrateur peut spécifier les options qui l'intéressent ce qui permet l'optimisation du service. Apache fournissant une architecture modulaire, la re-compilation du logiciel apache n'est généralement pas nécessaire pour ajouter ou supprimer des fonctionnalités complémentaires (ajout/suppression de modules).

Le choix de la méthode d'installation par paquets est fortement **conseillé**. Des dépôts complémentaires permettent d'installer des versions plus récentes d'apache sur des versions de distributions anciennes mais, en cas de problème, RedHat n'apportera pas son soutien.

Exemples de modules et de leurs rôles respectifs :

- **mod_access** : filtre l'accès des clients par leur nom d'hôte, adresse IP ou autre caractéristique
- **mod_alias** : permet la création d'alias ou répertoires virtuels
- **mod_auth** : authentifie les clients
- **mod_cgi** : exécute les scripts CGI
- **mod_info** : fournit des informations sur l'état du serveur
- **mod_mime** : associe les types de fichiers avec l'action correspondante
- **mod_proxy** : propose un serveur proxy (serveur mandataire)
- **mod_rewrite** : réécrit les URL
- ...

4.2.1. Installation par rpm

Interroger la base de données des "rpm"

```
[root]# rpm -qa "http*"
```

Installez à partir de paquets liés à la distribution

```
[root]# rpm -ivh httpd-xxx.rpm
```

Installer si nécessaire les dépendances demandées.

4.2.2. Installation par yum

Si vous avez à disposition un dépôt yum :

```
[root]# yum install httpd
```

Lancer Apache au démarrage du serveur :

```
[root]# chkconfig httpd on
```

Avant de se lancer dans une installation, il est important de savoir si une version du serveur Apache est installée :

```
rpm -qa "http*"
```

Lors de l'installation, un groupe apache et un utilisateur apache sont créés.

```
[root]# grep apache /etc/group  
apache:x:48  
  
[root]# grep apache /etc/passwd  
apache:x:48:48:Apache:/var/www:/sbin/nologin  
  
[root]# grep apache /etc/shadow  
apache:!:14411::::::
```

Le serveur Apache travaille sous l'identité d'un utilisateur “apache” appartenant à un groupe “apache”. Lors de l'installation, ce groupe et cet utilisateur sont créés. L'utilisateur apache étant un utilisateur système, aucun mot de passe ne lui est attribué, ce qui rend impossible la connexion au système avec cet utilisateur.

4.2.3. Les fichiers de configuration

Le répertoire `/etc/httpd/conf/` contient le fichier de configuration principal d'Apache `httpd.conf`. Ce fichier est très bien commenté. En général, ces commentaires suffisent pour éclairer l'administrateur sur les options dont il dispose.

Il peut être judicieux de créer un fichier de configuration sans commentaires :

```
[root]# egrep -v '^#|^$' /etc/httpd/conf/httpd.conf > httpd.conf-lite
```

Le répertoire `/etc/httpd/conf.d/` contient les fichiers de configuration des sites virtuels ou des modules installés (`ssl`, `php`, `welcome.conf`, `phpmyadmin`, etc.)

4.2.4. Manuel d'utilisation

Il existe un package contenant un site faisant office de manuel d'utilisation d'apache. Il s'agit du package `httpd-manual-xxx.noarch.rpm`. Une fois ce package installé vous pourrez accéder au manuel simplement avec un navigateur web à cette adresse <http://127.0.0.1/manual>.

4.2.5. Lancement du serveur

- Par le script d'init :

```
[root]# /etc/rc.d/init.d/httpd {start|restart|status}
```

- Avec la commande service :

```
[root]# service httpd {start|restart|status}
```

- Avec la commande apachectl fourni par apache :

```
[root]# apachectl {start|restart|stop}
```

Il est indispensable de lancer/relancer le serveur :

- après l'installation ;
- après toute modification de la configuration.

4.2.6. Parefeu

Le pare-feu “iptables” interdit l'accès aux ports 80 et 443. Pensez à le configurer (ou à désactiver ce service sur plateforme de test) !

```
[root]# system-config-firewall-tui
```

Ou :

```
[root]# service iptables stop
[root]# service ip6tables stop
```

La configuration d'un pare-feu fait l'objet d'un autre cours.

4.3. Arborescence

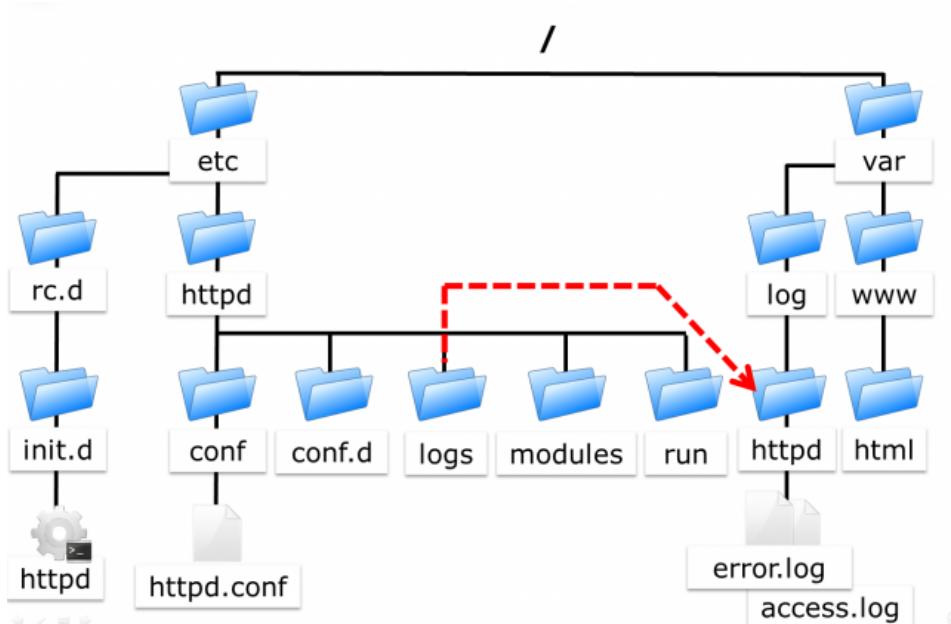


Figure 4.3. Arborescence d'Apache

L'arborescence peut varier en fonction des distributions.

- **/etc/httpd/** : Ce répertoire est la racine du serveur. Il contient l'ensemble des fichiers du serveur Apache.

- **/etc/httpd/conf/** : Ce répertoire contient l'ensemble des fichiers de configuration du serveur. Il possède des sous-dossiers pour des éléments de configuration précis.
- **/var/www/html/** : Ce répertoire est le répertoire de publication par défaut. Il contient les fichiers nécessaires à l'affichage de la page web par défaut du serveur Apache. Quand l'administrateur veut publier un site, il peut déposer ses fichiers dans ce répertoire.

D'autres répertoires existent sous /var/www :

- **cgi-bin** : contient les scripts CGI ;
- **icons** : contient des icônes, notamment celles pour identifier le type de fichier ;
- **error** : contient les messages d'erreur d'Apache, ce sont ces fichiers qu'il faudra modifier pour personnaliser les messages d'erreur.
- **/var/log/httpd/** : Ce répertoire contient les fichiers de logs du serveur Apache.
 - Le fichier access-log garde une trace des différents accès au serveur ;
 - Le fichier error-log contient la liste des erreurs rencontrées pendant l'exécution du service ;
 - Les fichiers logs sont personnalisables, l'administrateur peut aussi en créer de nouveaux.
- **/etc/httpd/modules** : Répertoire contenant les liens vers le répertoire “/usr/lib/httpd/modules” contenant les modules utilisables par Apache. Un module est une extension logicielle d'Apache, lui permettant par exemple d'interpréter le PHP (ex: mod-php5.so).
- **/etc/rc.d/init.d/httpd** : Script de démarrage du serveur httpd.

4.4. Configuration du serveur

La configuration globale du serveur se fait dans /etc/httpd/conf/httpd.conf.

Ce fichier est découpé en 3 sections qui permettent de configurer :

- en **section 1** l'environnement global ;
- en **section 2** le site par défaut et les paramètres par défaut des sites virtuels ;
- en **section 3** les hôtes virtuels.

L'**hébergement virtuel** permet de mettre en ligne **plusieurs sites virtuels** sur le même serveur. Les sites sont alors différenciés en fonction de leurs noms de domaines, de leurs adresses IP, etc.

La modification d'une valeur en section 1 ou 2 impacte l'ensemble des sites hébergés.

En environnement mutualisé, les modifications seront donc effectuées en section 3.

Pour faciliter les mises à jour futures, il est vivement recommandé de créer un fichier de configuration section 3 pour chaque site virtuel.

4.4.1. Section 1

Les différentes directives rencontrées en section 1 sont :

Tableau 4.1. Directives principales de la section 1

Directives	Observations
ServerTokens	Cette directive sera vue dans le cours Apache – sécurité.
ServerRoot	Indique le chemin du répertoire contenant l'ensemble des fichiers constituant le serveur Apache.
PidFile	Le fichier cible de la directive contient le numéro de PID du serveur à son démarrage.
Timeout	Le nombre de secondes avant le délai d'expiration d'une requête trop longue (entrant ou sortante).
KeepAlive	Connexion persistante (plusieurs requêtes par connexion TCP).
MaxKeepAliveRequests	Nombre maximum de connexions persistantes.
KeepAliveTimeout	Nombre de secondes à attendre la requête suivante du client avant fermeture de la connexion TCP.
Listen	Permettre à apache d'écouter sur des adresses ou des ports spécifiques.
LoadModule	Charger des modules complémentaires (moins de modules = plus de sécurité).
Include	Inclure d'autres fichiers de configuration au serveur.

Directives	Observations
ExtendedStatus	Afficher plus d'information sur le serveur dans le module server-status.
User et Group	Permet de lancer les processus apache avec différents utilisateurs. Apache se lance toujours en tant que root puis change son propriétaire et son groupe.

Le serveur Apache a été conçu comme un serveur puissant et flexible, pouvant fonctionner sur une grande variété de plateformes.

Plateformes différentes et environnements différents signifient souvent fonctionnalités différentes, ou utilisation des méthodes méthodes pour implémenter la même fonctionnalité le plus efficacement possible.

La conception modulaire d'apache autorise l'administrateur à choisir quelles fonctionnalités seront incluses dans le serveur en choisissant les modules à charger soit à la compilation, soit à l'exécution.

Cette modularité comprend également les fonctions les plus élémentaires du serveur web.

Certains modules, les Modules Multi-Processus (MPM) sont responsables de l'association aux ports réseau de la machine, acceptent les requêtes, et se chargent de les répartir entre les différents processus enfants.

Pour la version d'Apache de Windows, le MPM utilisé sera mpm-winnt.

Sous Linux, les sites très sollicités utiliseront un MPM threadé comme worker ou event, tandis que les sites privilégiant la stabilité utiliseront prefork.

Voir la page <http://httpd.apache.org/docs/2.2/fr/mpm.html>

Configuration par défaut des modules prefork et worker :

Configuration des modules dans /etc/http/conf/httpd.conf.

```
<IfModule prefork.c>
StartServers 8
MinSpareServers 5
MaxSpareServers 20
ServerLimit 256
MaxClients 256
```

```
MaxRequestPerChild 4000
</IfModule>

<IfModule worker.c>
StartServers 4
MaxClients 300
MinSpareThreads 25
MaxSpareThreads 75
ThreadsPerChild 25
MaxRequestsPerChild 0
</IfModule>
```

Le module Prefork, activé par défaut, s'appuie sur des processus. Il est donc plus stable mais nécessite plus de mémoire pour fonctionner. Le module Worker, quand à lui, s'appuie sur des threads, ce qui le rend plus performant, mais des modules comme le Php ne sont pas compatible.

Choisir un module plutôt qu'un autre est donc une tâche complexe, tout autant que l'optimisation du module MPM retenu (nombre de client, de requêtes, etc.).

Par défaut Apache est configuré pour un service moyennement sollicité (256 clients max).

La configuration minimal d'un serveur apache ressemble à ceci :

Configuration d'apache minimal dans /etc/httpd/conf/httpd.conf.

```
ServerRoot /etc/httpd
KeepAlive On
Listen 80
Listen 160.210.150.50:1981
LoadModule ...
Include conf.d/*.conf
User apache
Group apache
```

SELinux

Attention par défaut la sécurité via SELinux est active. Elle empêche la lecture d'un site sur un autre répertoire que "/var/www".

Le répertoire contenant le site doit posséder le contexte de sécurité httpd_sys_content_t.

Le contexte actuel se vérifie par la commande :

```
[root]# ls -Z /rep
```

Rajouter le contexte via la commande :

```
[root]# chcon -vR --type=httpd_sys_content_t /rep
```

Elle empêche également l'ouverture d'un port non standard. Il faut ouvrir manuellement le port désiré à l'aide de la commande semanage (non installée par défaut).

```
[root]# semanage port -a -t http_port_t -p tcp 1664
```

Directives User et Group

Définir un compte et un groupe de gestion d'Apache

Historiquement, apache était lancé par root, ce qui posait des problèmes de sécurité. Apache est toujours lancé par root mais change ensuite son identité. Généralement User Apache et Group Apache.



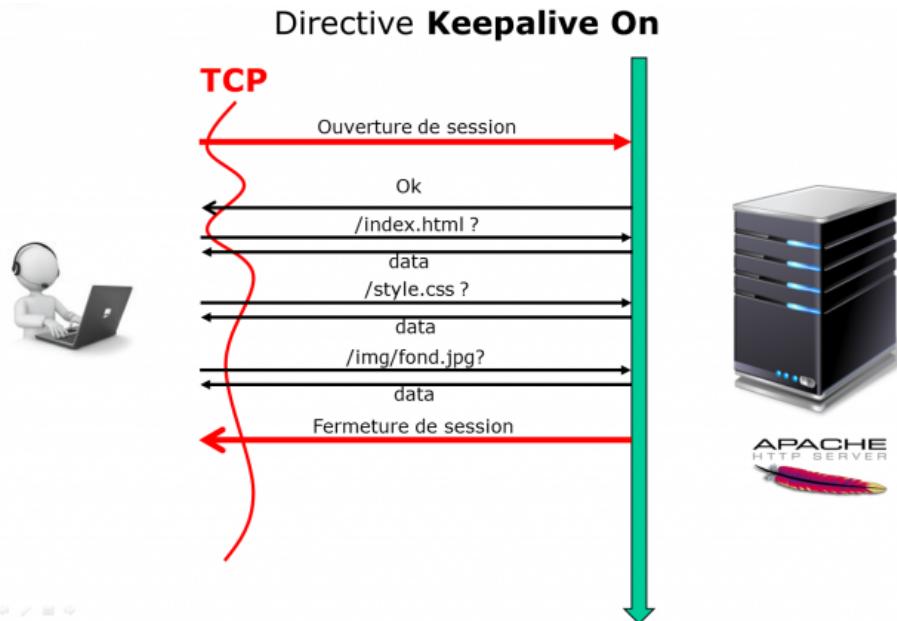
Attention jamais ROOT !!!

Le serveur Apache (processus httpd) est lancé par le compte de super-utilisateur root. Chaque requête d'un client déclenche la création d'un processus "fils". Pour limiter les risques, il faut lancer ces processus enfants avec un compte moins privilégié.

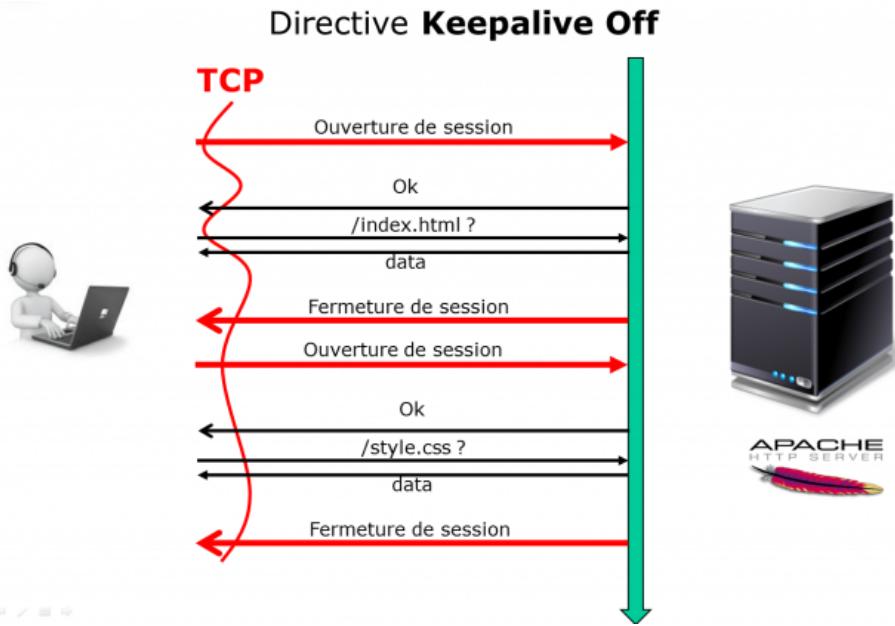
Les directives User et Group servent à déclarer le compte et le groupe utilisés pour la création des processus enfants.

```
User apache  
Group apache
```

Ce compte et ce groupe doivent avoir été créés dans le système (par défaut cela est fait à l'installation). Par mesure de précaution supplémentaire, s'assurer que le compte n'est pas interactif (ne peut pas ouvrir de session).



Avec la directive KeepAlive désactivée, chaque demande de ressource sur le serveur nécessite une ouverture de connexion TCP, ce qui est long à effectuer d'un point de vue réseau et gourmand en ressource système.



Avec la directive KeepAlive à On, le serveur conserve la connexion ouverte avec le client le temps du Keepalive.

Sachant qu'une page web est constituée de plusieurs fichiers (images, feuilles de styles, javascripts, etc.), cette stratégie est rapidement gagnante.

Il est toutefois nécessaire de bien paramétrer cette valeur au plus juste :

- Une valeur trop courte pénalise le client,
- Une valeur trop longue pénalise les ressources du serveur.

Des demandes de configuration spécifiques peuvent être faites par le client en hébergement mutualisé. Auquel cas, les valeurs de KeepAlive seront paramétrées directement dans le VirtualHost du client ou au niveau du mandataire (ProxyKeepalive et ProxyKeepaliveTimeout).



L'affichage de cette page prouve que le serveur est fonctionnel. Mais le serveur ne dispose pas encore de site à publier. Paramétrons la section 2.

4.4.2. Section 2

La section 2 paramètre les valeurs utilisées par le serveur principal. Le serveur principal répond à toutes les requêtes qui ne sont pas prises en charge par un des Virtualhosts de la section 3.

Les valeurs sont également utilisées comme valeur par défaut pour les sites virtuels.

- **ServerAdmin** : spécifie une adresse de messagerie qui apparaîtra dans certaines pages auto-générées, comme dans les pages d'erreurs.
- **ServerName** : spécifie le nom qui servira d'identification pour le serveur. Peut être déterminé automatiquement, mais il est recommandé de le spécifier explicitement (adresse IP ou nom DNS).

- **DocumentRoot** : spécifie le répertoire contenant les fichiers à servir aux clients. Par défaut **/var/www/html/**.
- **ErrorLog** : spécifie le chemin vers le fichier d'erreurs.
- **LogLevel** : debug, info, notice, warn, error, crit, alert, emerg.
- **LogFormat** : définir un format spécifique de log à utiliser avec la directive CustomLog.
- **CustomLog** : spécifie le chemin vers le fichier d'accès.
- **ServerSignature** : vue dans le cours sécurité.
- **Alias** : spécifie un répertoire extérieur à l'arborescence et le rend accessible par un contexte. La présence ou l'absence du dernier slash dans le contexte à son importance.
- **ScriptAlias** : spécifie le dossier contenant les scripts serveurs (idem alias) et les rend exécutables.
- **Directory** : spécifie des comportements et des droits d'accès par répertoire.
- **AddDefaultCharset** : spécifie le format d'encodage des pages envoyées (les caractères accentués peuvent être remplacés par des ?....).
- **ErrorDocument** : personnaliser les pages d'erreurs.
- **server-status** : rapport sur l'état du serveur.
- **server-info** : rapport sur la configuration du serveur.

La directive ErrorLog

La directive ErrorLog permet de définir le journal des erreurs.

Cette directive définit le nom du fichier dans lequel le serveur enregistre toutes les erreurs qu'il rencontre. Si le file-path n'est pas absolu, il est supposé être relatif à ServerRoot.

Syntaxe :

```
ErrorLog file-path
```

Exemple :

```
ErrorLog logs/error-log
```

La directive DirectoryIndex

La directive DirectoryIndex permet de définir la page d'accueil du site.

Cette directive indique le nom du fichier qui sera chargé en premier, qui fera office d'index du site ou de page d'accueil.

Syntaxe :

```
DirectoryIndex page-à-afficher
```

Le chemin complet n'est pas précisé car le fichier est recherché dans le répertoire spécifié par DocumentRoot

Exemple :

```
DocumentRoot /var/www/html  
DirectoryIndex index.php, index.htm
```

Cette directive indique le nom du fichier index du site web. L'index est la page par défaut qui s'ouvre quand le client tape l'URL du site (sans avoir à taper le nom de cet index). Ce fichier doit se trouver dans le répertoire indiqué par la directive DocumentRoot.

La directive DirectoryIndex peut spécifier plusieurs noms de fichiers index séparés par des espaces. Par exemple, une page d'index par défaut au contenu dynamique et en deuxième choix une page statique.

La directive ServerAdmin

La directive ServerAdmin permet d'indiquer le mail de l'administrateur.

Syntaxe :

```
ServerAdmin email-adresse
```

Exemple :

ServerAdmin webmaster@formatux.fr

La balise Directory

La balise Directory permet de définir des directives propre à un répertoire.

Cette balise permet d'appliquer des droits à un ou plusieurs répertoires. Le chemin du répertoire sera saisi en absolu.

Syntaxe :

```
<Directory directory-path>
Définition des droits des utilisateurs
</Directory>
```

Exemple :

```
<Directory /home/SitesWeb/SiteTest>
Allow from all    # nous autorisons tout le monde
</Directory>
```

La section Directory sert à définir un bloc de consignes s'appliquant à une partie du système de fichiers du serveur. Les directives contenues dans la section ne s'appliqueront qu'au répertoire spécifié (et ses sous-répertoires).

La syntaxe de ce bloc accepte les caractères génériques mais il faudra préférer alors utiliser le bloc DirectoryMatch.

Dans l'exemple suivant, nous allons refuser l'accès au disque dur local du serveur quelque soit le client. Le répertoire « / » représente la racine du disque dur.

```
<Directory />
Order deny, allow
Deny from all
</Directory>
```

Dans l'exemple suivant, nous allons autoriser l'accès au répertoire de publication / var/www/html pour tous les clients.

```
<Directory /var/www/html>
    Order allow, deny
    Allow from all
</Directory>
```

Prise en compte des modifications

La commande apachectl permet de tester la syntaxe du fichier de conf :

```
[root]# service httpd configtest
```

ou :

```
[root]# apachectl -t
```

puis :

```
[root]# /etc/rc.d/init.d/httpd {start|restart|status}
```

ou :

```
[root]# service httpd {start|restart|status}
```

ou :

```
[root]# apachectl {start|restart|stop}
```

Les commandes précédentes ont pour effet de couper les connexions en cours. Apache propose une solution plus élégante, qui lance de nouveaux serveurs et attend la fin du timeout pour détruire les anciens processus :

Ne pas couper les connexions TCP actives :

```
[root]# service httpd graceful
```

4.5. Configuration avancée du serveur

4.5.1. Le mod_status

Le mod_status permet d'afficher une page /server-status ou /server-info récapitulant l'état du serveur :

Configuration des directives server-status et server-info.

```
<Location /server-status>
  SetHandler server-status
  Order allow,deny
  Allow from 127.0.0.1
</Location>

<Location /server-info>
  SetHandler server-info
  Order allow,deny
  Allow from 127.0.0.1
  Allow from 172.16.96.105
</Location>
```

La page /server-status :

Apache Server Status for 172.16.96.105

Server Version: Apache/2.2.15 (Unix) DAV/2 PHP/5.3.3 mod_ssl/2.2.15 OpenSSL/1.0.1e-fips
Server Built: Oct 16 2014 14:45:47

Current Time: Friday, 13-Mar-2015 09:50:06 CET
Restart Time: Friday, 13-Mar-2015 09:49:56 CET
Parent Server Generation: 0
Server uptime: 10 seconds
Total accesses: 33 - Total Traffic: 29 kB
CPU Usage: u:0.02 s:0.02 cu:0 cs:0 .4% CPU load
3.3 requests/sec - 2969 B/second - 899 B/request
6 requests currently being processed, 3 idle workers

```
_.KRWKKK.....  
.....  
.....
```

Scoreboard Key:
 " " Waiting for Connection, "s" Starting up, "r" Reading Request,
 "w" Sending Reply, "k" Keepalive (read), "d" DNS Lookup,
 "c" Closing connection, "l" Logging, "g" Gracefully finishing,
 "x" Idle cleanup of worker, " ." Open slot with no current process

Srv	PID	Acc	M	CPU	SS	Req	Conn	Child	Slot	Client	VHost	
1-0	1734	4/4/4	K	0.00	0	0	0.00	0.00	172.16.96.232	mail.lemorvan.lan	GET /roundcubemail/skins/la	
2-0	1735	4/4/4	K	0.00	0	0	0.00	0.00	172.16.96.232	mail.lemorvan.lan	GET /roundcubemail/skins/la	
3-0	1736	13/13/13	W	0.04	0	0	29.7	0.03	0.03	172.16.96.232	mail.lemorvan.lan	GET /server-status HTTP/1.1
4-0	1737	4/4/4	K	0.00	0	0	0.00	0.00	0.00	172.16.96.232	mail.lemorvan.lan	GET /roundcubemail/skins/la
5-0	1738	4/4/4	K	0.00	0	0	0.00	0.00	0.00	172.16.96.232	mail.lemorvan.lan	GET /roundcubemail/skins/la
6-0	1739	4/4/4	K	0.00	0	0	0.00	0.00	0.00	172.16.96.232	mail.lemorvan.lan	GET /roundcubemail/skins/la

Srv Child Server number - generation
PID OS process ID
Acc Number of accesses this connection / this child / this slot
M Mode of operation
CPU CPU usage, number of seconds
SS Seconds since beginning of most recent request
Req Milliseconds required to process most recent request
Conn Kilobytes transferred this connection
Child Megabytes transferred this child

La page /server-info :

Module Name: `mod_alias.c`
Content handlers: `none`
Configuration Phase Participation: Create Directory Config, Merge Directory Configs, Create Server Config, Merge Server Configs
Request Phase Participation: Translate Name, Fixups
Module Directives:
 Alias - a fakename and a realname
 ScriptAlias - a fakename and a realname
 Redirect - an optional status, then document to be redirected and destination URL
 AliasMatch - a regular expression and a filename
 ScriptAliasMatch - a regular expression and a filename
 RedirectMatch - an optional status, then a regular expression and destination URL
 RedirectTemp - a document to be redirected, then the destination URL
 RedirectPermanent - a document to be redirected, then the destination URL
Current Configuration:
 In file: /etc/httpd/conf.d/phpMyAdmin.conf
 8: Alias /phpMyAdmin /usr/share/phpMyAdmin
 9: Alias /phpmyadmin /usr/share/phpMyAdmin
 In file: /etc/httpd/conf.d/roundcubemail.conf
 5: Alias /roundcubemail /usr/share/roundcubemail
 In file: /etc/httpd/conf/httpd.conf
 551: Alias /icons/ "/var/www/icons/"
 576: ScriptAlias /cgi-bin/ "/var/www/cgi-bin/"
 855: Alias /error/ "/var/www/error/"

4.5.2. Hébergement mutualisé (section 3)

Dans le cas d'un hébergement mutualisé, le client pense visiter plusieurs serveurs. En réalité, il n'existe qu'un seul serveur et plusieurs sites virtuels.

Pour mettre en place un hébergement mutualisé, il faut mettre en place des hôtes virtuels :

- en déclarant plusieurs ports d'écoute ;
- en déclarant plusieurs adresses IP d'écoute (hébergement virtuel par **IP**) ;
- en déclarant plusieurs noms de serveur (hébergement virtuel par **nom**);

Chaque site virtuel correspond à une arborescence différente.

La section 3 du fichier httpd.conf permet de déclarer ces hôtes virtuels.

Pour faciliter les mises à jour futures, il est vivement recommandé de créer un fichier de configuration section 3 pour chaque site virtuel.

Choisissez un hébergement virtuel “par IP” ou “par nom”. En production, il est déconseillé de mixer les deux solutions.

- Chaque site virtuel peut être configuré dans un fichier indépendant ;
- Les VirtualHosts sont stockés dans /etc/httpd/conf.d/ ;
- L’extension du fichier est .conf.

La balise VirtualHost

La balise VirtualHost permet de définir des hôtes virtuels.

Syntaxe:

Syntaxe d'un fichier virtualhostXXX.conf.

```
<VirtualHost adresse-IP[:port]>
    # si la directive "NameVirtualHost" est présente
    # alors "adresse-IP" doit correspondre à celle saisie
    # sous "NameVirtualHost" ainsi que pour le "port".
    ...
</VirtualHost>
```

Si nous configurons le serveur Apache avec les directives de base vues précédemment, nous ne pourrons publier qu'un seul site. En effet, nous ne pouvons pas publier plusieurs sites avec les paramètres par défaut : même adresse IP, même port TCP et absence de nom d'hôte ou nom d'hôte unique.

L'usage des sites virtuels va nous permettre de publier plusieurs sites web sur un même serveur Apache. Nous allons définir des blocs qui décriront chacun un site web. Ainsi chaque site aura sa propre configuration.

Pour des facilités de compréhension, nous associons souvent un site web à une machine unique. Les sites virtuels ou hôtes virtuels (virtual hosts) sont appelés ainsi parce qu'ils dématérialisent le lien entre machine et site web.

Exemple 1 :

```
Listen 192.168.0.10:8080
<VirtualHost 192.168.0.10:8080>
    DocumentRoot /var/www/site1/
    ErrorLog /var/log/httpd/site1-error.log
```

```
</VirtualHost>
```

```
Listen 192.168.0.11:9090
<VirtualHost 192.168.0.11:9090>
    DocumentRoot /var/www/site2/
    ErrorLog /var/log/httpd/site2-error.log
</VirtualHost>
```

L'hébergement virtuel basé sur IP est une méthode permettant d'appliquer certaines directives en fonction de l'adresse IP et du port sur lesquels la requête est reçue. En général, il s'agit de servir différents sites web sur des ports ou des interfaces différents.

La directive NameVirtualHost

La directive NameVirtualHost permet de définir des hôtes virtuels à base de nom.

Cette directive est obligatoire pour configurer des hôtes virtuels à base de nom. Nous spécifions avec cette directive l'adresse IP sur laquelle le serveur recevra des demandes des hôtes virtuels à base de nom.

Syntaxe :

```
NameVirtualHost adresse-IP[:port]
```

Exemple :

```
NameVirtualHost 160.210.169.6:80
```

Il faut placer la directive avant les blocs descriptifs de sites virtuels. Elle désigne les adresses IP utilisées pour écouter les requêtes des clients vers les sites virtuels. La syntaxe est la suivante :

Pour écouter les requêtes sur toutes les adresses IP du serveur il faut utiliser le caractère *.

4.6. Exemple de publication de sites

Fichier **/etc/httpd/conf.d/80-site1.conf** :

```
<VirtualHost 160.210.69.6:80>
    # déclaration de l'arborescence du site
    DocumentRoot "/var/sitesweb/site1"
    # déclaration des index du site
    DirectoryIndex "Index1.htm"
    # déclaration des droits sur le site
    <Directory "/var/sitesweb/site1">
        Allow from all
    </Directory>
</VirtualHost>
```

Fichier /etc/httpd/conf.d/1664-site2.conf :

```
Listen 1664
<VirtualHost 160.210.69.6:1664>
    # déclaration de l'arborescence du site
    DocumentRoot "/var/sitesweb/site2"
    # déclaration des index du site
    DirectoryIndex "Index2.htm"
    # déclaration des droits sur le site
    <Directory "/var/sitesweb/site2">
        Allow from all
    </Directory>
</VirtualHost>
```

4.7. Redirection des logs vers syslog

Il est possible, en passant par la commande **logger** de rediriger les logs d'apache vers le syslog local ou vers un serveur syslog distant avec le mot clé **apache**.

Modification des lignes **CustomLog** et **ErrorLogs** du fichier .conf correspondant au VHOST désiré :

fichier .conf du vhost.

```
ErrorLog "|usr/bin/logger -t apache -p local6.info"
CustomLog "|usr/bin/logger -t apache -p local6.info" combined
```

5

Sécurisation du serveur web Apache

5.1. Introduction

La sécurisation d'un site internet nécessite la mise en place de trois mesures :

- La sécurité du service en lui-même.

Par défaut un serveur Apache va envoyer des informations avec ses pages web. Ces informations contiennent des détails sur sa version ou ses modules. Pour un pirate, ces informations vont lui permettre de cibler les attaques en fonction des failles connues. Il est donc impératif de masquer ces informations sur un serveur de production.

- La sécurité des accès aux données.

Pour empêcher l'accès aux données, il est nécessaire de mettre en place une authentification. Cette authentification peut être d'ordre applicative, et s'appuyer par exemple sur une base de données, ou être gérée par le service Apache.

- La sécurité des échanges (protocole https).

La mise en place d'un processus d'authentification sur le serveur pour protéger l'accès aux données n'empêche pas l'attaquant d'intercepter les requêtes sur le réseau, et d'ainsi obtenir les documents désirés voir les informations d'identification (utilisateur et mot de passe). L'authentification va donc de pair avec le chiffrement des échanges.



Moins d'informations = intrusion plus difficile = Masquer l'identité du serveur Apache.

5.2. Masquage de l'identité d'Apache

5.2.1. Directive ServerSignature

Afficher une ligne de bas de page pour les documents générés par le serveur (messages d'erreur)

Syntaxe de la directive ServerSignature.

```
ServerSignature On | Off | EMAil
```

Exemple :

```
ServerSignature Off
```

La directive "ServerSignature" ajoute un pied de page aux documents générés par le serveur (messages d'erreur, liste des répertoires, ftp, etc.).

L'utilité d'ajouter ces informations apparaît par exemple lorsqu'une réponse à une requête traverse plusieurs proxys. Dans ces conditions, sans ces informations, il devient difficile pour l'utilisateur de déterminer quel élément de la chaîne de proxy a produit un message d'erreur.

Sur un serveur de production, pour des raisons de sécurité, il est préférable de positionner cette option à Off.

ServerSignature On :

Authorization Required

This server could not verify that you are authorized to access the document requested.
The credentials required.

Apache/2.2.6 (Fedora) Server at 160.210.125.126 Port 80

ServerSignature Off :

Authorization Required

This server could not verify that you are authorized to access the document requested.
the credentials required.

L'information fournie par cette page semble anodine mais pourtant est très précieuse pour un attaquant.

En effet, un éventuel attaquant apprend que le serveur apache, disponible à l'adresse IP 160.210.125.126 est un serveur Apache 2.2.6. La version actuelle d'apache étant la version 2.2.29 (en décembre 2014).

L'attaquant peut donc utiliser le document situé à cette adresse : http://www.apache.org/dist/httpd/CHANGES_2.2 pour cibler ses attaques.

5.2.2. Directive ServerTokens

Renseigner le champ "server" de l'en-tête http.

Syntaxe de la directive ServerTokens.

```
ServerTokens Prod[uctOnly] | Min[imal] | OS | Full
```

Exemple :

```
ServerTokens Prod
```

Dans un navigateur web, la page internet que nous consultons n'est que la partie visible par l'utilisateur du contenu d'une requête HTTP. Les en-têtes HTTP fournissent de nombreuses informations, que ce soit du client vers le serveur ou du serveur vers le client.

Ces en-têtes peuvent être visualisées par exemple avec :

- la commande wget, qui permet le téléchargement d'une URL en ligne de commande,
- avec le module "Développement Web" fourni en standard avec le navigateur Firefox.

En-têtes sans ServerTokens (full)

```
[root]# wget -S http://160.210.125.126
--14:30:07-- http://160.210.125.126/
=> `index.html'
Connexion vers 160.210.125.126:80...connecté.
requête HTTP transmise, en attente de la réponse...
HTTP/1.1 200 OK
Date: Mon, 01 Mar 2010 13:30:07 GMT
Server: Apache/2.2.6 (Fedora) DAV/2
Last-Modified: Mon, 01 Mar 2010 08:26:10 GMT
ETag: "16e9e-3c55-594c80"
Accept-Ranges: bytes
Content-Length: 15445
Connection: close
Content-Type: text/html; charset=UTF-8
Longueur: 15445 (15K) [text/html]

100%[=====] 15445          ----K/s

14:30:07 (86.86 MB/s) - « index.html » sauvegardé [15445/15445]
```

Pour les mêmes raisons que pour la directive ServerSignature vue précédemment, il est impératif de limiter les informations transmises par un serveur de production.

Tableau 5.1. La directive ServerTokens

Valeur	Information
Prod[uctOnly]	Server : Apache
Min[imal]	Server : Apache/1.3.0
OS	Server : Apache/1.3.0 (Unix)
Full	Server : Apache/1.3.0 (Unix) PHP/3.0 MyMod/1.2

En-têtes avec ServerTokens Prod.

```
[root]# wget -S http://160.210.125.126
--14:30:07-- http://160.210.125.126/
=> `index.html'
Connexion vers 160.210.125.126:80...connecté.
```

```

requête HTTP transmise, en attente de la réponse...
HTTP/1.1 200 OK
Date: Mon, 01 Mar 2010 13:30:07 GMT
Server: Apache
Last-Modified: Mon, 01 Mar 2010 08:26:10 GMT
ETag: "16e9e-3c55-594c80"
Accept-Ranges: bytes
Content-Length: 15445
Connection: close
Content-Type: text/html; charset=UTF-8
Longueur: 15445 (15K) [text/html]

100%[=====] 15445           - . . . K/s

14:30:07 (86.86 MB/s) - « index.html » sauvegardé [15445/15445]

```



Le choix le plus judicieux est généralement de modifier le fichier /etc/httpd/conf/httpd.conf pour mettre la directive ServerTokens à Prod

5.3. Gestion des authentifications

L'authentification est la procédure qui consiste à vérifier l'identité d'une personne ou d'un ordinateur afin d'autoriser l'accès de cette entité à des ressources (systèmes, réseaux, applications,...).

Il existe de nombreuses méthodes d'authentifications :

- Authentification classique (simple) ;
- Authentification LDAP ;
- Authentification via serveur de base de données ;
- Authentification via PAM ;
- Authentification via SSO.

Apache permet par défaut d'assurer ce processus, avec une authentification classique ou simple, qui s'appuie sur des fichiers de textes contenant les informations de connexion des utilisateurs.

Cette fonctionnalité basique peut être enrichie par des modules, et permettre à apache de s'appuyer sur :

- Une authentification via un serveur LDAP. Ce procédé permet de déléguer l'authentification des utilisateurs (et leur appartenance à des groupes) à un serveur LDAP, dont c'est la fonctionnalité première. Apache utilise alors le module `mod_authnz_ldap`, qui dépend du module Apache d'accès à LDAP, `mod_ldap`, qu'il faut aussi installer.
- Une authentification via un serveur de base de données. Ce procédé s'appuiera sur le langage SQL (Structured Query Language) pour la gestion des utilisateurs, de leurs mots de passe et leur appartenance à des groupes.
- Le module PAM (Pluggable Authentication Modules). Ce procédé permet de déléguer l'authentification au système d'exploitation. Pour mettre en place cette procédure, il faut installer deux modules : `mod_auth_pam` et `pam_auth_external`.

Ces modules ne sont pas actifs par défaut mais sont présents dans les dépôts.



Dans ce chapitre nous ne verrons que l'authentification classique (dite aussi « simple »).

5.3.1. Authentification classique

Authentification classique : protéger l'accès à un site ou à un dossier d'un site par la mise en place d'une authentification par mot de passe.

L'activation du module d'authentification pour un site peut se faire :

- Soit dans la configuration globale du site, si l'administrateur du site est également administrateur du serveur, ou que l'administrateur lui en a laissé l'accès.

Les directives de configuration se positionne entre les balises `<Directory>` ou `<Location>`.



C'est la méthode à privilégier.

- Si la modification du fichier de configuration globale n'est pas possible, ce qui est souvent le cas d'un serveur mutualisé, la protection se fera alors dans un fichier .htaccess directement dans le dossier à protéger.

Dans ce cas, l'administrateur du serveur aura explicitement configuré cette possibilité dans la configuration globale avec la directive AllowOverride à All ou à AuthConfig.

Ce fichier étant évalué à chaque accès, cela peut induire une petite perte au niveau des performances.



L'authentification sécurise l'accès aux données, mais la donnée transite toujours en clair durant la transmission au client.

5.3.2. Directive AuthType

Renseigner le type de contrôle des autorisations

Directives associées : AuthName, AuthUserFile

Syntaxe de la directive AuthType.

```
AuthType Basic | Digest
```

Exemple :

```
AuthType Basic
```

AuthType indique à Apache d'utiliser le protocole Basic ou Digest pour authentifier l'utilisateur :

- Authenfication Basic : Transmission du mot de passe client en clair

Pour mettre en place cette méthode, il faut utiliser la commande htpasswd qui permet de créer un fichier qui va contenir les logins (ou les groupes) et les mots de passe des utilisateurs (ou les groupes) habilités à accéder au dossier Web sécurisé (la commande étudiée plus loin).

- Authentification Digest : Hachage MD5 128 bits du mot de passe avant transmission

Ce module implémente l'authentification HTTP basée sur les condensés MD5, et fournit une alternative à mod_auth_basic en ne transmettant plus le mot de passe en clair.

Cependant, cela ne suffit pas pour améliorer la sécurité de manière significative par rapport à l'authentification basique. En outre, le stockage du mot de passe sur le serveur est encore moins sûr dans le cas d'une authentification à base de condensés que dans le cas d'une authentification basique.

C'est pourquoi l'utilisation de l'authentification basique associée à un chiffrement de la connexion via mod_ssl constitue une bien meilleure alternative.

Plus d'informations : http://httpd.apache.org/docs/2.2/fr/mod/mod_auth_digest.html.



Pour des raisons de sécurité, seul le mécanisme Basic sera utilisé par la suite.

Configuration du fichier /etc/httpd/conf/httpd.conf :

Dans les balises <Directory> ou <Location>, ajouter les directives d'authentification :

Syntaxe Apache pour protéger l'accès à un dossier.

```
<Directory directory-path >
    AuthType Basic | Digest
    AuthName "text"
    AuthUserFile directory-path/.PrivPasswd
    Require user | group | valid-user
</Directory>
```

- AuthName est le message qui est affiché dans la boîte de dialogue de saisie du login et du mot de passe.
- AuthUserFile indique le chemin et le nom du fichier contenant le nom des utilisateurs et leur mot de passe. Pour une identification sur un groupe il faut travailler avec la directive AuthGroupFile. Ces deux directives font parties du module mod_auth .
- Require est la règle d'authentification à proprement parler. Elle indique à Apache qu'un utilisateur ayant réussi à s'authentifier à partir du fichier des

mots de passe (spécifié dans la directive `AuthUserFile`) peut accéder au site Web sécurisé.

Exemple :

```
<VirtualHost www.monsite.com >
.....
<Directory /var/www/html/sitetest>
# Type d'authentification
AuthType Basic
# texte affiché dans la boite de dialogue
AuthName " Acces Securise "
# fichier contenant les logins et mdp
AuthUserFile /var/www/private/sitetest/.PrivPasswd
# Accès par vérification du mot de passe
require valid-user
</Directory>
</VirtualHost>
```

Le fichier `.PrivPasswd` est créé avec la commande `htpasswd`, que nous verrons plus loin dans ce chapitre.



Le fichier de gestion des logins et des mots de passe `.PrivPasswd` est un fichier sensible. Il ne faut pas le placer dans le répertoire de publication de votre site, mais plutôt dans un répertoire extérieur à l'arborescence de votre site suivi du nom du site.

Exemple

```
[root]# mkdir -p /var/www/private/SiteTest
```

Le fichier `.PrivPasswd` sera ensuite créé dans ce répertoire à l'aide de la commande `htpasswd`.

Le fichier `.htaccess` utilise les mêmes directives que précédemment, mais non encadrés par les balises `Directory` ou `Location`.

Syntaxe:

```
AuthType Basic | Digest
```

```
AuthName "text"
AuthUserFile directory-path/.PrivPasswd
Require user | group | valid-user
```

Avec dans le fichier « /etc/httpd/conf/httpd.conf » <Directory directory-path > AllowOverride AuthConfig </Directory>

Un serveur web répond généralement pour plusieurs sites. Dans ce cas, il est plus simple de configurer ce type de spécificité de configuration directement dans le dossier en question.

- Avantages : l'usage d'un fichier .htaccess a le mérite d'être simple et permet notamment de protéger un dossier Web racine ainsi que les sous-dossiers (sauf si un autre fichier .htaccess y contrevient), il est possible de déléguer la configuration ou la personnalisation du service à un administrateur du site.
- Inconvénients : il n'est pas simple de maintenir un nombre élevé de fichiers .htaccess. L'évaluation du fichier .htaccess par le serveur peut induire sur les performances.

Il ne faut pas oublier d'autoriser la configuration du module d'identification par fichier .htaccess à l'aide de la directive : AllowOverride AuthConfig.

Sans cette directive, apache ne permettra pas au fichier .htaccess d'écarter la configuration définie dans la configuration globale.

```
<VirtualHost www.monsite.com >
    <Directory /home/SitesWeb/SiteTest>
        # déclaration de l'utilisation de .htaccess
        AllowOverride AuthConfig
    </Directory>
</VirtualHost>
```

Exemple de fichier .htaccess.

```
# Type d'authentification
AuthType Basic
# texte affiché dans la boite de dialogue
AuthName " Acces Securise "
# fichier contenant les logins et mdp
AuthUserFile /var/www/private/sitetest/.PrivPasswd
# Accès par vérification du mot de passe
```

```
require valid-user
```

Lors du traitement d'une requête, Apache cherche la présence du fichier .htaccess dans tous les répertoires du chemin menant au document depuis la directive DocumentRoot.

Exemple, avec une directive DocumentRoot à /home/sitesweb/ avant de retourner /home/sitesWeb/sitetest/indextest.htm Apache examine les fichiers :

- /home/sitesWeb/.htaccess
- /home/sitesWeb/sitetest/.htaccess

Mieux vaut désactiver cette fonctionnalité sur / (activé par défaut) : <Directory /> AllowOverride None </Directory>

5.3.3. Commande htpasswd

La commande htpasswd permet de gérer les utilisateurs du site.

Syntaxe de la commande htpasswd.

```
htpasswd [-options] passwordfile username [password]
```

Exemples :

```
[root]# cd /var/www/private/sitetest
[root]# htpasswd -cb .PrivPasswd user1 mdpuser1
[root]# htpasswd -D .PrivPasswd user
```

Tableau 5.2. La directive ServerTokens

Option	Observation
-c	Créer un nouveau fichier
-b	Indiquer le mot de passe sur la ligne de commande
-m	Chiffrer le mot de passe en md5
-D	Supprimer un utilisateur de la liste d'accès

La commande htpasswd met en place la liste des utilisateurs habilités à accéder au site sécurisé, dans le cas de l'utilisation de la directive « AuthType » avec la

valeur « Basic » et vérifie les droits sur le répertoire, afin qu'au moins le groupe « apache » puisse y accéder.

Pour créer le fichier et définir le premier utilisateur :

```
[root]# cd /var/www/private/siteTest  
[root]# htpasswd -c .PrivPasswd user1
```

Puis saisir le mot de passe

Pour ajouter les autres utilisateurs dans le fichier :

```
[root]# htpasswd .PrivPasswd user2
```

Puis saisir le mot de passe

Pour ajouter un utilisateur et définir son mot de passe à la suite de la commande

```
[root]# htpasswd -b .PrivPasswd user3 mdpuser3
```

Ajouter un utilisateur avec un mot de passe chiffré en md5 :

```
[root]# htpasswd -bm .PrivPasswd user4 mdpuser4
```

Le résultat donne un fichier /var/www/private/sitetest/.PrivPasswd :

```
user1:$1Pd$EsBY75M  
user2:Mku4G$j4p£k11  
user3:Ng7Rd$5F$68f  
...
```

5.4. Utilisation du module SSL

Le protocole TLS (Transport Layer Security), successeur du protocole SSL (Secure Socket Layer) est un protocole de sécurisation des échanges sur Internet.

Le protocole SSL a été créé à l'origine par Netscape. Le principe de fonctionnement du TLS repose sur le recours à un tiers, l'Autorité de Certification (CA/Certificate Authority).

C'est un protocole de niveau 4 sur lequel les protocoles des couches OSI supérieures s'appuient. Il est donc commun pour les protocoles imaps, pops, ldaps, etc.

Le fichier openssl.cnf (/etc/pki/tls/openssl.cnf) peut être configuré de façon à minimiser les données à renseigner à chaque invocation des utilitaires openssl lors de la création des clefs de chiffrement.

Etant donné son caractère hautement critique, l'administrateur veillera à utiliser une version d'openssl à jour de correctifs.

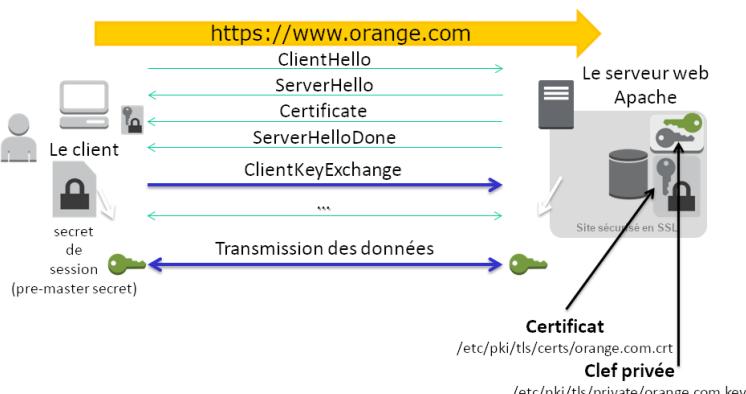
Le protocole SSL/TLS existe en 5 versions :

- SSLv2
- SSLv3
- TLSv1
- TLSv1.1
- TLSv1.2

5.4.1. Prérequis

- Le port 443 (https) est-il ouvert sur les pare-feu ?
- Le logiciel OpenSSL est-il installé et à jour sur le serveur ?
- Le module mod_ssl est-il installé sur le serveur Apache et activé ?

5.4.2. Établissement d'une session TCP https (port 443)



Lors du ClientHello, le client se présente au serveur. Il lui soumet les méthodes de cryptologie, de compression, et les standards SSL qu'il connaît.

Le serveur répond au client avec un message ServerHello. Les 2 parties se sont mises en accord sur le CypherSuite qu'ils allaient utiliser, par exemple avec un CypherSuite TLS_RSA_WITH_AES_128_MD5.

Ils doivent maintenant s'échanger un secret de session : le pre-master secret. Son chiffrement s'effectue avec le certificat public du serveur transmis pendant le message Certificate.

Le pre-master secret est dérivé en clefs symétriques sur le client et le serveur. Ces clefs symétriques serviront à la transmission des données.

La chaîne (CypherSuite) est composée de 5 éléments distincts :

- Le protocole SSL/TLS
- L'algorithme asymétrique utilisé, principalement RSA et ECDSA
- L'algorithme symétrique utilisé, comme AES, Camelia, SEED, 3DES ou RC4
- Le mécanisme de protection des données pour éviter qu'un assaillant puisse modifier les messages chiffrés (HMAC ou AEAD). Dans le cas du HMAC on choisira une fonction de hachage (MD5, SHA-1, SHA-2)
- La présence ou non de confidentialité persistante ou PFS (Perfect Forward Secrecy)

5.4.3. Mise en place d'un site TLS

La mise en place d'un site TLS respecte les étapes suivantes :

1. Installation du module mod_ssl
2. Configuration du module mod_ssl
3. Création de la clé privée du serveur
4. Création du certificat du serveur depuis sa clé privée
 1. Création d'un Certificat Signing Request (CSR) depuis le certificat et transmission à la CA pour signature
 2. Installation du certificat

3. L'hôte virtuel peut être configuré en TLS

5.4.4. Le logiciel OpenSSL

Le logiciel OpenSSL, utilitaire cryptographique, implémente les protocoles réseaux :

- Secure Sockets Layer (SSL V2/V3, couche de sockets sécurisés) ;
- Transport Layer Security (TLS v1, sécurité pour la couche de transport).

OpenSSL permet :

- Création de paramètres des clefs RSA, DH et DSA
- Création de certificats X.509, CSRs et CRLs
- Calcul de signature de messages
- Chiffrement et Déchiffrement
- Tests SSL/TLS client et server
- Gestion de mail S/MIME signé ou chiffrés

Création des clés et certificats

Les clés et certificats sont stockés dans le répertoire /etc/pki/tls/.

Un dossier private accueille les clés privées. Un dossier certs accueille les certificats publiques.



Les droits doivent être à 440.

Pour installer le module mod_ssl :

```
[root]# yum install mod_ssl
```

Pour vérifier la présence du module dans le serveur Apache :

```
[root]# httpd -t -D DUMP_MODULES | grep ssl_module
```

```
ssl_module (shared)
```

La commande nous indique que le module est disponible mais pas qu'il est activé.

Après cette installation, vous devez trouver le module "mod_ssl.so" dans le répertoire /etc/httpd/modules" qui est en fait un lien symbolique sur /usr/lib/httpd/modules.

Pour activer le mod_ssl dans /etc/httpd/conf/httpd.conf, la ligne suivante doit être présente :

```
LoadModule ssl_module modules/mod_ssl.so
```

Le module mod_ssl peut être configuré dans le fichier /etc/httpd/conf.d/ssl.conf

N'oubliez pas de redémarrer le service Apache :

```
[root]# service httpd restart
```

Pour accepter les requêtes TLS, le serveur Apache a besoin de deux fichiers :

- une clé privée : NomDeFichier.key;
- un certificat signé : NomDeFichier.crt

La signature de ce certificat est réalisée par un organisme de certification tiers (tel que Verisign ou Thawte). Cependant vous pouvez signer vous même votre certificat, la seule différence sera un avertissement par le navigateur lors de l'accès à une ressource TLS de votre serveur. La sécurité est la même, que le certificat soit signé par vous même ou pas un organisme.



Si vous décidez d'utiliser une autorité de certification auto-signée, son certificat devra être installée sur l'ensemble de vos postes.

- 1ère étape : Générer la clef privée

```
openssl genrsa \
-out /etc/pki/tls/private/orange.com.key \
```

```
2048
Generating RSA private key, 2048 bit long modulus
-----+++  
-----+++  
e is 65537 (0x10001)

chmod 400 /etc/pki/tls/private/orange.com.key
```

- 2ème étape : Générer une demande de certificat

```
openssl req  
-new  
-key /etc/pki/tls/private/orange.com.key  
-out /etc/pki/tls/certs/orange.com.csr
```

- 3ème étape : Envoi de la demande de certificat à l'autorité de certification (CA)
 - 4ème étape : L'autorité de certification (CA) renvoie un certificat signé
 - 5ème étape : Sécuriser et sauvegarder les certificats.

```
chmod 400 /etc/pki/tls/private/orange.com.key  
chmod 400 /etc/pki/tls/certs/orange.com.crt
```

- 6ème étape : Déployer les certificats sur le serveur
 - 7ème étape : Configurer le vhost

```
NameVirtualHost 192.168.75.50:443
<VirtualHost 192.168.75.50:443>
    ServerName www.orange.com

    SSLEngine on
    SSLCertificateFile /etc/pki/tls/certs/orange.com.crt
    SSLCertificateKeyFile /etc/pki/tls/private/orange.com.key

    DocumentRoot /home/SitesWeb/SiteOrange
    DirectoryIndex IndexOrange.htm

<Directory /home/SitesWeb/SiteOrange>
    allow from all
</Directory>
```

```
</VirtualHost>
```

Certificats Auto-Signés

Auto-signer ses certificats revient à créer sa propre autorité de certification.

- 1ère Etape : Générer la paire de clefs de l'autorité de certification

```
openssl genrsa -out /etc/pki/CA/private/cakey.pem  
openssl req \  
    -new \  
    -x509 \  
    -key /etc/pki/CA/private/cakey.pem \  
    -out /etc/pki/CA/certs/cacert.pem \  
    -days 365
```

- 2ème Etape : Configurer openssl

/etc/pki/tls/openssl.cnf.

```
[ ca ]  
default_ca = CA_default  
  
[ CA_default ]  
  
dir = /etc/pki/CA  
certificate = $dir/certs/cacert.pem  
private_key = $dir/private/cakey.pem
```

Vérifier la présence du fichier /etc/pki/CA/index.txt.

Si celui-ci s'avère absent, il faut le créer :

```
touch /etc/pki/CA/index.txt  
echo '1000' > /etc/pki/CA/serial
```

Puis faire :

```
echo '1000' > /etc/pki/CA/serial
```

- 3ème Etape : Signer une demande de certificat

```
openssl ca \
-in /etc/pki/tls/certs/orange.com.csr \
-out /etc/pki/tls/certs/orange.com.crt
```



Pour que ce certificat soit reconnu par les clients, le certificat cacert.pem devrait également être installé sur chaque navigateur.

Le certificat orange.com.crt est à envoyer au client

6

Apache - haute disponibilité

Objectifs de ce cours :

- paramétrier Apache en serveur frontal
- répartir la charge entre plusieurs serveurs applicatifs
- paramétrier Apache pour la tolérance de panne

6.1. Introduction

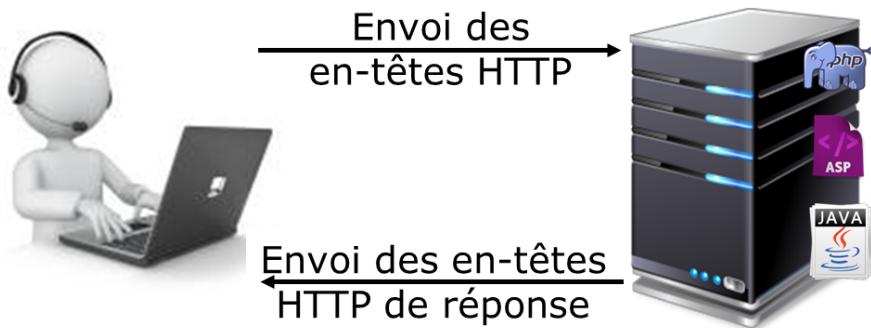
Apache est un serveur **modulaire** web (protocole HTTP).

Il peut :

- Gérer un cache de données ;
- Faire office de serveur mandataire ;
- Compresser les données envoyées aux clients ;
- et plein d'autres choses encore...

6.2. Serveur Applicatif

Un serveur applicatif héberge des applications dynamiques développées dans un langage de programmation web : Php / Asp / Java / Python ...



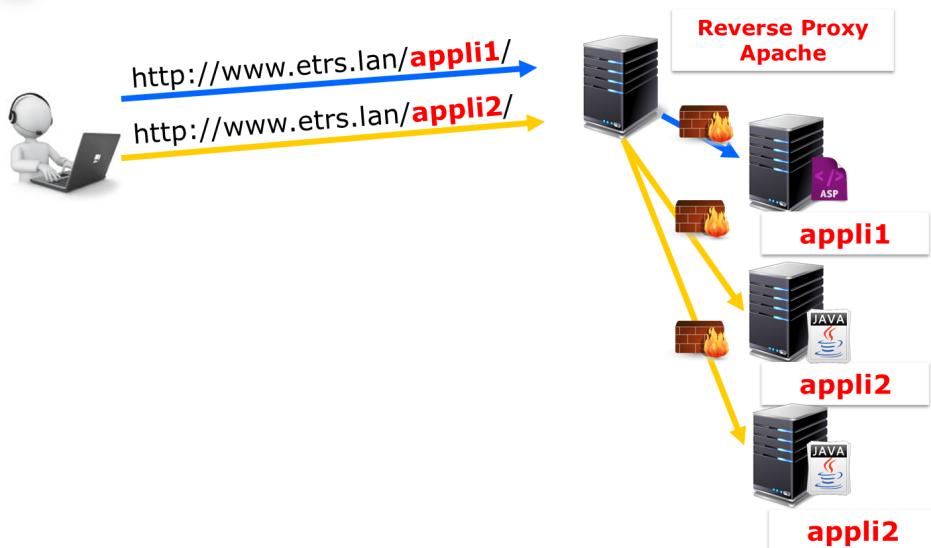
6.3. Reverse proxy

Un serveur reverse proxy (ou serveur mandataire) est mandaté par des clients pour interroger les serveurs applicatifs.

Aux yeux du client, un seul serveur est en ligne pour lui répondre. La complexité du SI lui est masqué.

Les serveurs applicatifs sont isolés derrière un (ou plusieurs) pare-feu. Leurs ressources sont dédiées aux processus métiers (SQL, Web-services, etc.).

Le reverse proxy se charge de la compression, de la mise en cache, du chiffrement, et de la répartition de charge et/ou tolérance de panne.



6.3.1. Le module mod_proxy

Le module **mod_proxy** est une extension d'apache pour faire office de serveur mandataire / Passerelle HTTP.

- La directive **ProxyPreserveHost On** utilise l'en-tête de requête entrante Host pour la requête du mandataire
- La directive **ProxyPass chemin url** référence des serveurs distants depuis l'espace d'URLs du serveur local
- La directive **ProxyPassReverse chemin url** ajuste l'URL dans les réponses HTTP envoyées par le mandataire en inverse

Les différentes fonctionnalités de mandataire d'Apache sont réparties entre plusieurs modules complémentaires :

- mod_proxy_http,
- mod_proxy_ftp,
- mod_proxy_ajp,
- mod_proxy_balancer
- et mod_proxy_connect.

Apache peut être configuré comme mandataire directe (proxy) ou comme mandataire inverse (reverse proxy ou mode passerelle).

Pour la configuration d'un mandataire direct, reportez vous à la documentation du serveur SQUID.

Un mandataire inverse apparait au client comme un serveur web standard. Aucune configuration du client n'est nécessaire. Le client adresse ses demandes de contenus ordinaires dans l'espace de nommage du mandataire inverse. Ce dernier décide alors où envoyer ces requêtes et renvoie le contenu au client comme s'il l'hébergeait lui-même.

L'accès des utilisateurs à un serveur situé derrière un pare-feu est une utilisation typique du mandataire inverse.

Configuration exemple d'un VirtualHost avec reverse-proxy.

```
<VirtualHost 127.0.0.1:8080>
    ProxyPreserveHost On
    ProxyVia On
    <Proxy *>
        Order deny,allow
        Allow from all
    </Proxy>

    ProxyPass /pagebleue http://127.0.0.1:8080/pagebleue
    ProxyPassReverse /pagebleue http://127.0.0.1:8080/pagebleue
</VirtualHost>
```

- La directive **ProxyPreserveHost on**, lorsqu'elle est activé, va transmettre l'en-tête Host: de la requête entrante vers le serveur mandaté au lieu du nom d'hôte spécifié par la directive ProxyPass.
- La directive **ProxyVia On** permet de contrôler l'utilisation de l'en-tête http **Via:**. Définie à On, chaque requête ou réponse se verra ajouter une ligne d'en-tête **Via:** pour le serveur courant.

Des solutions spécialisées, comme **HaProxy** ou des équipements physiques comme les **Citrix NetScaler**, existent sur le marché, et sont plus puissantes qu'Apache. Au moment de choisir la solution de mandataire inverse, il faudra prendre en compte les éléments du tableau suivant :

	Points positifs	Points négatifs
HaProxy	Plus rapide	Spécifique au reverse proxy (pas de cache, nécessite varnish ou memcached)
	Développement actif	La syntaxe est différente des autres logiciels utilisés (NginX, Varnish, etc.)
Apache	Grand nombre de fonctionnalités, il peut tout faire	Performances moindres
	Solution unique pour tout le SI	
	Facilité de maintenance	

6.4. Simuler la charge

La commande **ab** (apache Benchmark) permet d'envoyer des requêtes http à un client

Syntaxe de la commande ab.

```
ab -n X -c Y url
```

Exemple :

Exemple d'utilisation de la commande ab.

```
ab -n 5000 -c 2 http://127.0.0.1:8080/page1
```

Tableau 6.1. Options de la commande ab

Options	Commentaires
-n	Nombre de requêtes à envoyer
-c	Nombre de requêtes à envoyer par paquets

6.5. Répartir la charge

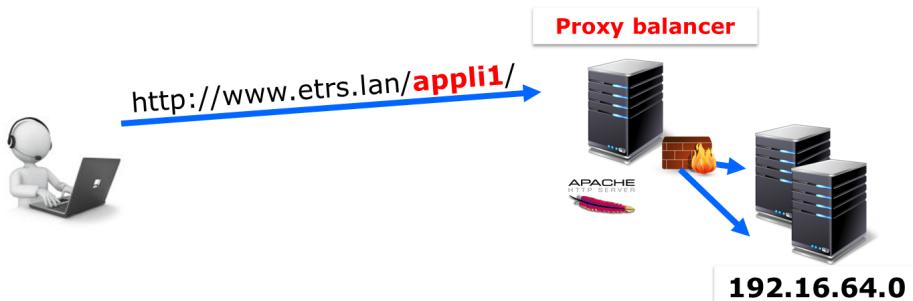
Pourquoi répartir la charge ?

- Améliorer les performances
- Accueillir plus de clients
- Gérer la maintenance
- Faire de la tolérance de panne



Plusieurs petits serveurs seront plus puissants qu'un gros pour un coût moindre.

Pour répartir la charge, Apache utilise un proxy-balancer.



6.5.1. Le module `mod_proxy_balancer`

Le module `mod_proxy_balancer` est une extension pour la répartition de charge.

- La directive **BalancerMember url [clé=valeur]** ajoute un membre à un groupe de répartition de charge
- La directive **loadfactor=X** est un facteur de charge entre 1 et 100
- La directive **ProxySet url [clé=valeur]** définit les différents paramètres relatifs à la répartition de charge
- La directive **lbmethod=byrequests|bytraffic|bybusyness** spécifie la méthode de répartition de charge (par défaut à byrequests). Les 3 méthodes de répartition de charge sont :

- byrequests : réparti la charge entre les membres du cluster en fonction du nombre de requêtes traitées
- bytraffic : réparti la charge entre les membres du cluster en fonction du trafic traitées
- bybusyness : réparti la charge entre les membres du cluster en fonction de la charge actuelle
- La directive **stickySession=JSESSIONID|PHPSESSIONID** spécifie un nom de session persistant du répartiteur et dépend du serveur d'applications d'arrière plan.

```
<Proxy balancer://mycluster>
    BalancerMember http://192.16.164.1:8080/pages loadfactor=50
    BalancerMember http://192.16.164.2:8080/pages loadfactor=50

    ProxySet lbmethod=bytraffic
</Proxy>

ProxyPass /monsite balancer://mycluster
ProxyPassReverse /monsite balancer://mycluster
```

6.5.2. Le module mod_status

Le module **mod_status** permet de suivre l'état du load-balancer.

- La directive **ProxyStatus On** affiche l'état du répartiteur de charge du mandataire
- La directive **SetHandler balancer-manager** force le traitement par le gestionnaire balancer-manager

```
ProxyStatus On

<Location /balancer-manager>
    SetHandler balancer-manager
    Allow from all
</Location>
```



| Vous pouvez maintenant visiter la page /balancer-manager

6.6. Tolérance aux pannes

Apache peut réagir en cas de panne du serveur métier et exclure du cluster le service qui ne répond plus.

```
<Proxy balancer://mycluster>
    BalancerMember http://127.0.0.1:8080/pagebleue loadfactor=50 retry=30
    BalancerMember http://127.0.0.1:8080/pageverte loadfactor=50 retry=30

    ProxySet lbmethod=byrequests failonstatus=404

</Proxy>
```

Serveur de messagerie Postfix

7.1. Généralités



Le courrier électronique est apparu sur le réseau **ARPANET** dans les années 1970, ce qui fait du protocole **SMTP** un des plus anciens protocoles de **TCP/IP**.

Avec Internet, les systèmes de messagerie ont pris de l'importance et en 1980, **Sendmail** a été développé. Sendmail est devenu le premier serveur de messagerie important et utilisait déjà le protocole SMTP.

Postfix, dont le développement a été aidé par IBM, est apparu dès **1998**, afin de résoudre les problèmes de sécurité de Sendmail, tout en offrant une **administration beaucoup plus souple et modulaire**.

Le **MTA (Mail Transfer Agent)** par défaut de la distribution RedHat (Sendmail) est remplacé par Postfix sur la RedHat 6 (Novembre 2010).

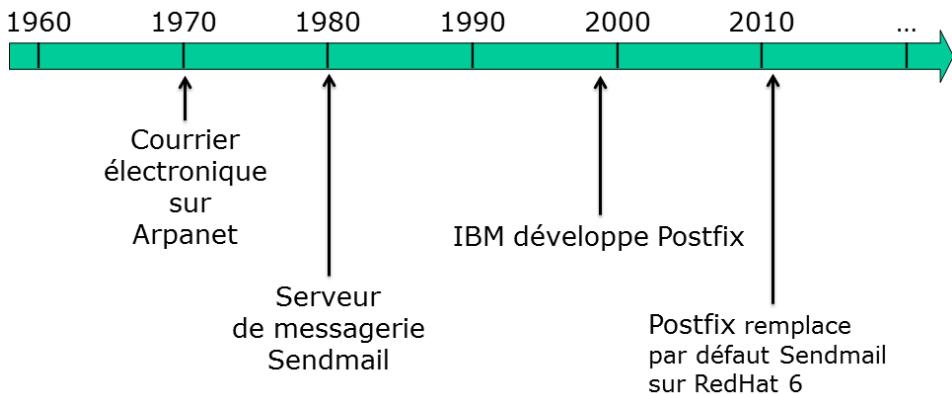


Figure 7.1. Historique des serveurs de messagerie sous Linux

Les systèmes de messagerie électronique reposent sur plusieurs standards et protocoles, définissant la façon dont sont composés les messages et leur acheminement, du rédacteur au destinataire.

L'**envoi de message** est assuré par le protocole **Simple Mail Transfer Protocol** (SMTP). SMTP est un protocole de communication de type texte.



Le fait que le protocole SMTP soit de type texte est intéressant. En effet, cela permet de tester son bon fonctionnement avec la commande **telnet**.

Les ports utilisés sont :

- **25** (sans authentification) ;
- **587** (avec authentification) ;
- **465** (SSL).

Local Mail Transfer Protocol (LMTP) est une variante de **ESMTP**, l'extension de SMTP. LMTP est défini dans la RFC 2033. LMTP a été conçu comme alternative aux échanges SMTP normaux dans les situations où la partie réceptrice ne possède pas de file d'attente des messages reçus. C'est le cas par exemple d'un agent de transfert de courrier agissant en tant qu'agent de distribution du courrier.

La **réception de message** peut se faire à l'aide de deux protocoles :

- **Internet Message Access Protocol (IMAP)** ;

- **Post Office Protocol (POP).**

IMAP est un protocole permettant de récupérer les courriers électroniques déposés sur des serveurs de messagerie. Les messages sont conservés sur le serveur, ses fonctionnalités avancées en font le protocole de préférence (accès **multipostes**, accès via **webmail**, etc.).

IMAP utilise le port **143** en clair ou via **STARTTLS** et le port **993** via **IMAPS** (déprécié) en SSL.

POP est également un protocole permettant de récupérer les courriers électroniques de l'utilisateur. En règle générale, POP se connecte sur le serveur, récupère le courrier, efface le courrier sur le serveur et se déconnecte. Ce fonctionnement par défaut ne permet pas l'accès aux mails depuis plusieurs postes, ni l'itinérance.

POP utilise le port **110** en clair ou via **STARTTLS** et le port **995** (déprécié) en SSL.

Le langage Sieve a été conçu pour permettre de filtrer des messages directement sur les serveurs.

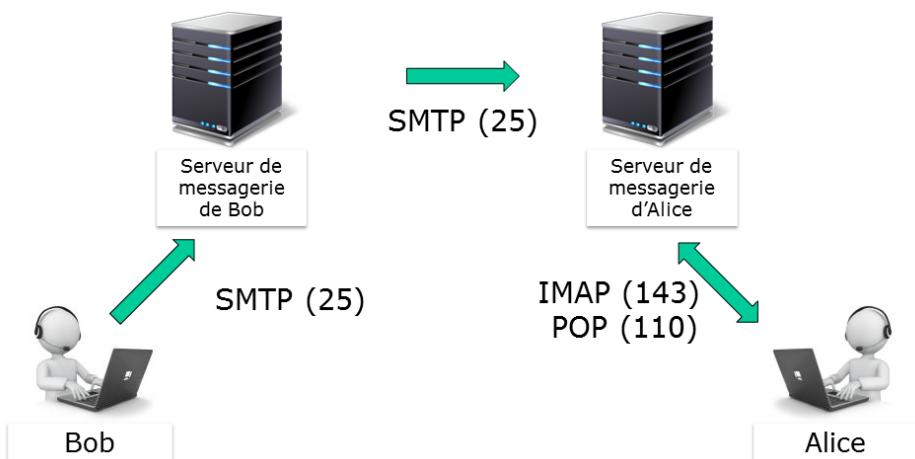


Figure 7.2. Les protocoles de messagerie mis en oeuvre

7.1.1. Les agents de messagerie

Mail User Agent

Un **Mail User Agent (MUA)**, ou client de messagerie, est un logiciel servant à **lire** et à **envoyer** des courriers électroniques. Ce sont en général des clients lourds, mais il existe aussi des applications Web : les **webmails**. Les webmails tentent d'offrir les mêmes fonctionnalités qu'un client lourd.

Ces logiciels prennent en charge le protocole **SMTP** pour l'envoi de messages et les protocoles **POP** et **IMAP** pour leur réception.

Parmi les MUA les plus connus, citons : Thunderbird (Mozilla), Evolution (Novell), Outlook et Windows Mail (Microsoft), Kmail, Lotus Notes, Apple Mail, Opéra Mail, etc.

Mail Transfer Agent

Lorsqu'un message est envoyé depuis un client de messagerie **MUA**, il est transféré au serveur de courrier, le **Mail Transfer Agent (MTA)**. Un MTA implémente à la fois le **client** (transfert de mail) et le **serveur** (réception) du protocole SMTP.

Les termes Mail Server, Mail Exchanger, Mail Relay et MX Hosts peuvent aussi faire référence à un serveur MTA.

Le système de nom de domaine **DNS** associe à un domaine un ou plusieurs serveur de messagerie (Mail Exchanger - MX) par ordre de priorité (la plus haute priorité étant la plus faible valeur). Un MTA peut donc transférer un mail pour lequel il n'est pas destinataire soit à un serveur relais (si configuré), soit au serveur désigné par l'enregistrement MX du système DNS (le saut suivant - Next Hop).

Les MTA font donc en sorte qu'un message soit délivré d'un système à un autre. Si le MTA ne peut ni accepter, ni relayer un message, il le renvoie à son expéditeur.

Le message arrive au MTA responsable du domaine qui le stocke dans la boîte aux lettres de l'utilisateur.

Parmi les MTA les plus connus, citons : Postfix, Sendmail, Exim, Exchange, Qmail, Lotus Notes.

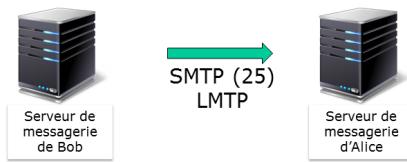


Figure 7.3. Le transfert de message entre MTA

Mail Delivery Agent

Le **Mail Delivery Agent (MDA)**, agent de **distribution du courriel**, est le logiciel qui intervient dans la dernière étape du processus de distribution d'un courrier électronique. Il est responsable de la **disposition du message dans la boîte aux lettres** de l'utilisateur. Il peut également être appelé **Local Delivery Agent (LDA)**.

C'est le MDA qui est chargé de gérer les problèmes comme un disque plein, une corruption de la boîte aux lettres, etc. et de signaler au MTA toute erreur de distribution.

Le MTA communique avec le MDA par l'intermédiaire des canaux d'entrées-sorties standards ou par un protocole spécialisé comme LMTP ou UUCP.

Parmi les MDA les plus connus, citons : Dovecot, Cyrus, Procmail, Local.

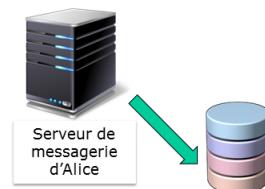


Figure 7.4. Le rôle de MDA

Mail Access Agent

Le **Mail Access Agent (MAA)**, permet à l'utilisateur final, après **authentification**, de récupérer le message. Le MUA de l'utilisateur communique avec le MAA par le protocole IMAP ou POP.

Parmi les MAA les plus connus, citons : Dovecot, Cyrus, Courier, Exchange.

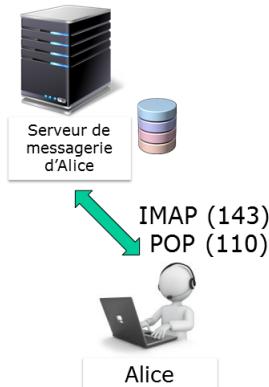


Figure 7.5. Le rôle de MAA

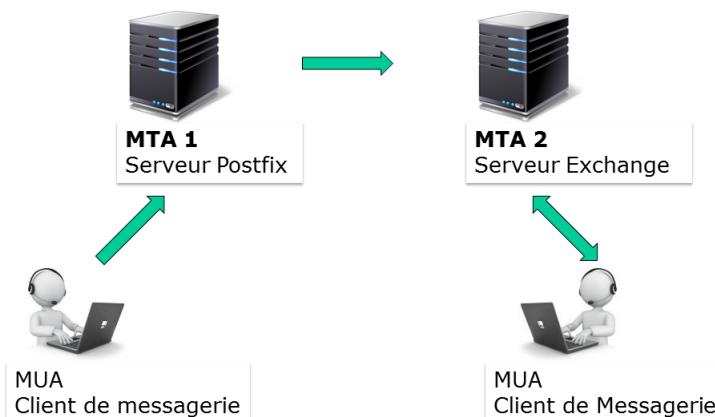


Figure 7.6. Les agents de transmission de message

7.1.2. Les relais SMTP

Le terme de “**relais de messagerie**” est couramment utilisé, dans le cas d’un MTA qui n’assurerait pas lui même la livraison du message au MTA final, mais qui se contenterait de servir d’intermédiaire entre le MUA du client et le MTA qui prend réellement l’acheminement du message en charge.

Cette fonctionnalité de relais est courante. Elle est par exemple présente dans nos box internet, qui acheminent l’ensemble des messages émis par un domicile vers un des serveurs MTA centraux. Les messages peuvent ensuite être filtrés (lutte anti-spam ou surveillance ?). Les fournisseurs d'accès à Internet évitent, en bloquant le port 25 en sortie des box, que des serveurs SMTP soient directement sollicités.

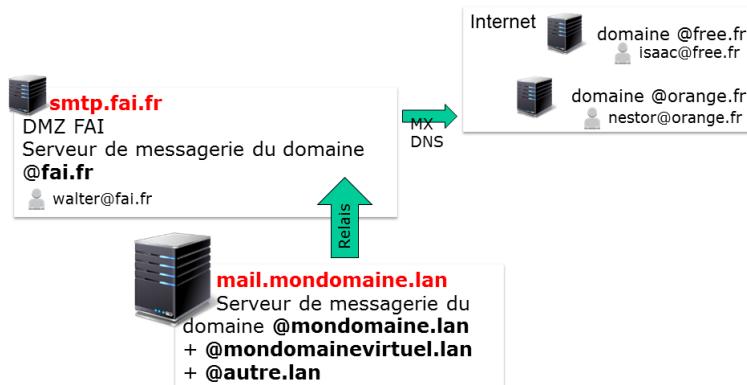


Figure 7.7. Système de messagerie avec relais

7.1.3. Les formats de stockage

Le format **mbox** est un format ouvert de stockage de courrier. Il repose sur le principe d'attribuer un fichier à chaque dossier (au lieu d'un fichier par message ou d'un répertoire par dossier).

Le format mbox permet un **affichage rapide d'une liste de mail**, puisqu'il ne nécessite l'ouverture que d'un seul fichier. La suppression ou l'ajout de mail est plus long et plus complexe à réaliser d'un point de vue système. L'accès concurrent à la même boîte aux lettres n'est pas possible car un verrou est positionné sur le fichier lors d'une action d'ajout ou de suppression.

Le format mbox est le format par défaut de postfix

Le format **Maildir** est une **structure de répertoires particulière**. Les courriels sont sauvegardés dans un fichier séparé, au sein d'une arborescence spécifique, ce qui lève le problème de verrou du format mbox.

De par son architecture, le format **Maildir est performant et fiable**. Il est plus adapté au protocole IMAP. À noter toutefois que l'affichage d'une liste de mails sera moins rapide qu'avec le format mbox.

Chaque répertoire Maildir contient au moins 3 sous-répertoires : **tmp**, **new**, **cur**. Les mails sont placés dans le répertoire tmp, puis déplacés par le MTA dans le répertoire new, pour enfin être déplacés après accès par un MUA dans le répertoire cur.

7.1.4. Synoptique

Un **MTA** pourra assurer les fonctionnalités suivantes :

- **Serveur de messagerie local** pour les comptes systèmes locaux comme root, bob, alice, etc.
- **Serveur d'un ou de plusieurs domaines de messagerie**, pour des comptes [¹](mailto:root@mondomaine.lan), [²](mailto:bob@mondomaine.lan), etc.
- **Serveur relais pour les domaines extérieurs** à son périmètre de gestion.

Un MTA peut également posséder des **routes spéciales**, contenues dans une **table de routage**, pour délivrer des messages à des serveurs de messagerie sans prendre en compte le chemin standard (par le relais ou par le serveur MX désigné lors d'une requête DNS).

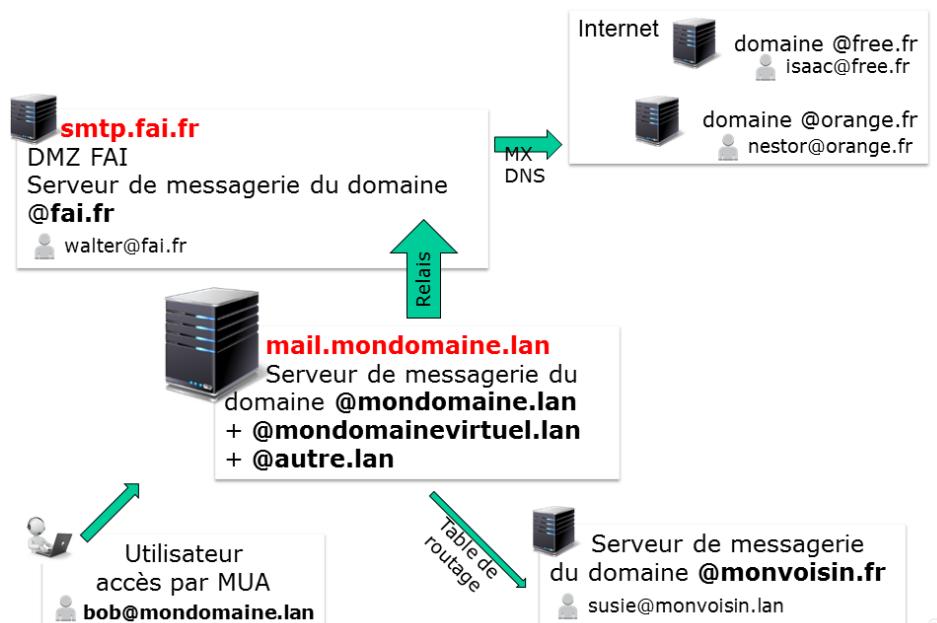


Figure 7.8. Synoptique global du système de messagerie

¹ <mailto:root@mondomaine.lan>

² <mailto:bob@mondomaine.lan>

7.2. Installation du service

Postfix devrait normalement être installé par défaut sur une RedHat/CentOS 6 ou 7.

```
[root]# yum install postfix
```

Postfix nécessite bien évidemment que le service network soit correctement configuré.

L'installation de postfix ajoute un utilisateur postfix, membre du groupe postfix.

```
[root]# grep postfix /etc/passwd  
postfix:x:89:89::/var/spool/postfix:/sbin/nologin  
  
[root]# grep postfix /etc/group  
postfix:x:89:x
```

Postfix étant un service réseau, il faut le paramétrier pour un démarrage à minima aux niveaux 3 et 5, puis le démarrer.

L'installation d'un nouveau service sur les distributions RedHat/CentOS n'implique pas leur démarrage automatique.

Après toute installation d'un service, il ne faut pas oublier de le démarrer avec la commande service, et d'automatiser le démarrage au reboot avec la commande chkconfig.

Tout service dépendant du réseau devrait être démarré comme le service network (chkconfig --list network).

```
[root]# chkconfig postfix on  
[root]# service postfix start
```

7.3. Arborescence et fichiers



Figure 7.9. Arborescence et fichiers du serveur postfix

- Le fichier de configuration du serveur Postfix : /etc/postfix/main.cf ;
- Les files d'attente sont groupées dans le répertoire : /var/spool/postfix/ ;
- Les boîtes de messagerie mbox sont stockées dans /var/spool/mail/ ;
- Les logs sont dans le fichier : /var/log/maillog.

Le fichier /etc/postfix/main.cf contient les paramètres de configuration de Postfix. Les paramètres qui n'y sont pas explicitement renseignés sont initialisés avec leur valeur par défaut. Seule la dernière occurrence du paramètre compte lorsque ce paramètre est défini plusieurs fois.

En cas de besoin, un fichier commenté de main.cf existe sous /usr/share/postfix/main.cf.

Les directives sont modifiables avec un éditeur de texte, mais la commande postconf permet l'édition en limitant le risque d'erreur.

Dans le fichier /etc/postfix/main.cf :

- Chaque instruction doit être en début de ligne (pas d'espace avant).
- Les espaces autour du signe “=” sont ignorés, comme ceux situés à la fin de la ligne logique.
- Une ligne démarrant avec un espace continue la ligne logique précédente.

7.4. Mise en oeuvre

La commande **telnet** est particulièrement adaptée pour tester des protocoles en mode texte tel SMTP.

Sa syntaxe est la suivante :

```
[root]# telnet localhost 25
```

Pour communiquer avec le serveur, il faut respecter les étapes attendues par le service.

Déroulé d'une session telnet sur le port 25 :

1. HELO
2. MAIL FROM:
3. RCPT TO:
4. DATA

Pour terminer la rédaction du mail, il conviendra de saisir un . sur une ligne seule.

7.4.1. Déroulé d'une session SMTP

Lancer dans un terminal la commande telnet localhost 25. Voici le déroulé de la session :

```
[root]# telnet localhost 25
Trying ::1...
Connected to localhost.
Escape character is '^]'.
220 mail.mondomaine.lan ESMTP Postfix
HELO mail.mondomaine.lan
250 mail.mondomaine.lan
MAIL FROM: bob
250 2.1.0 Ok
RCPT TO: alice
250 2.1.5 Ok
DATA
354 End data with<CR><LF>.<CR><LF>
From: test
To: monalice
Subject: Test de message

Ceci est un test.
Merci de votre coopération

.
250 2.0.0 Ok: queued as 642A59F6E8
```

```
QUIT
221 2.0.0 Bye
Connection closed by foreign host.
```

Les champs From: et To: du contenu du mail (après DATA) ne sont pas vérifiés par le serveur SMTP. Ils peuvent différer des valeurs fournies au service SMTP, un peu comme l'adresse du destinataire d'un courrier postal peut différer de l'adresse affichée dans le courrier.

Le message se termine lorsque le serveur reçoit une ligne ne contenant qu'un point.

Suivi du mail

Le traitement du mail par les différents agent peut être suivi dans le fichier **/var/log/maillog**. Dans ce cas, l'utilisation de la commande **tail -f nomdufichier** est particulièrement bien adaptée. L'utilisation de la commande **ccze**, qui permet la coloration syntaxique des fichiers de log, peut également être utilisée via un **grep** :

```
: tail -f /var/log/maillog | grep ccze
```

Voici un extrait du fichier de log, généré par la session telnet précédente :

```
[root]# tail -f /var/log/maillog | grep ccze
postfix/smtpd[19747]: connect from localhost[::1]
postfix/smtpd[19747]: 642A59F6E8: client=localhost[::1]
postfix/cleanup[19828]: 642A59F6E8: message-id=<...>
postfix/qmgr[19742]: 642A59F6E8: from=<bob@mail.mondomaine.lan>,
size=462, nrcpt=1 (queue active)
postfix/local[20100]: 642A59F6E8: to=<alice@mail.mondomaine.lan>,
orig_to=<alice>,
relay=local, ..., status=sent (delivered to mailbox)
postfix/qmgr[19742]: 642A59F6E8: removed
postfix/smtpd[19747]: disconnect from localhost[::1]
```

L'agent **smtpd** a pris en charge la connexion du client par le réseau en passant par **telnet** sur le port **25**. Si la connexion avait été faite via un outil local, c'est l'agent **pickup** qui aurait alors pris en charge le message, comme nous le verrons à la section suivante.

L'agent **cleanup** a ensuite pris en charge le message. Le destinataire d'origine, bob, n'étant pas **pleinement qualifié** et non conforme à la norme **RFC 822**,

cleanup l'a fourni au démon **trivial-rewrite** (évènement qui est non journalisé) pour permettre la réécriture de l'adresse de messagerie d'origine.

L'agent **cleanup** a ensuite déposé le message dans la file d'attente **incoming**, en attendant que l'agent **qmgr** le déplace dans la file **active** (queue active).

Le message ayant une portée locale (**orig_to=<alice>**), l'agent **trivial-rewrite** est de nouveau appelé pour rendre conforme cette adresse (**to=alice@mail.mondomaine.lan**³) par l'agent **qmgr**, qui le délivre à l'agent **local** pour stockage dans la boîte aux lettres d'**Alice**.

L'agent **qmgr** supprime alors le message de la file active.

7.4.2. La commande mailx

La commande **mailx** est une commande de traitement du courrier (un MUA) dont nous n'étudierons que la partie envoi du courrier.

```
mailx [-iInv] [-s sujet] [-a en-tete] [-c adresses cc] [-b adresses bcc]
adresse[s]
```

Le tableau suivant récapitule les principales options :

Tableau 7.1. Options principales de la commande mailx

Options	Information
-v	Affiche les détails de la livraison sur le terminal
-s	Spécifie le sujet en ligne de commande (seul le premier argument après le flag -s est utilisé en tant que sujet ; pensez à mettre des guillemets autour des sujets contenant des espaces).
-c liste	Liste les destinataires en copie carbone. 'liste' doit être une liste de noms séparés par des virgules.
-b	Liste les destinataires en copie cachée (Blind Carbon Copy).

Quelques options supplémentaires :

³ <mailto:alice@mail.mondomaine.lan>

Tableau 7.2. Options supplémentaires de la commande mailx

Options	Information
-i	Ignore les signaux d'interruption du terminal. Cela est particulièrement utile lors de l'utilisation de mail sur des lignes téléphoniques à bruit.
-I	Force mailx à se lancer en mode interactif même lorsque l'entrée n'est pas un terminal. En particulier, le caractère de commande spécial ~, utilisé lors de l'envoi d'un courrier, est seulement disponible interactivement.
-N	Désactive l'affichage initial des en-têtes du message lors de la lecture d'un courrier ou de l'édition d'un dossier de courriers.
-a	Spécifie des champs d'en-tête additionnels dans la ligne de commande comme "X-Loop: foo@bar", etc. Vous devez utiliser des guillemets si la chaîne contient des espaces. Cet argument peut être spécifié plus d'une fois, les en-têtes étant dans ce cas concaténés.
-e	N'envoie pas de courrier vide. Si le corps est vide, le message est sauté.
-f nom	Procède à la lecture du contenu de votre boîte aux lettres (ou le fichier spécifié nom) ; lorsque vous quittez, mail réécrit les messages non supprimés dans ce fichier.
-u utilisateur	Est équivalent à "mail -f /var/mail/utilisateur" sauf qu'il y a verrouillage.

Exemples :

```
[stagiaire]$ less corps
Bonjour
Au revoir
[stagiaire]$ mailx -s "coucou" "alice,bob" < corps
```

```
[stagiaire]$ mailx -s "coucou" alice@mail.formatux.lan
->Bonjour
->Au revoir
```

```
->. (ou ctrl+d)
```

Suivi du mail

Voici un extrait du fichier de log, généré par la session mailx précédente :

```
[root]# tail -f /var/log/maillog
postfix/pickup[19741]: 5707A9F8A: uid=1000 from=<stagiaire>
postfix/cleanup[22647]: 5707A9F8A: message-id=<...>
postfix/qmgr[19742]: 5707A9F8A: from=<stagiaire@mail.mondomaine.lan>,
size=549, nrcpt=2 (queue active)
postfix/local[22649]: 5707A9F8A: to=<alice@mail.mondomaine.lan>,
orig_to=<alice>,
relay=local, ..., status=sent (delivered to mailbox)
postfix/local[22650]: 5707A9F8A: to=<bob@mail.mondomaine.lan>,
orig_to=<bob>,
relay=local, ..., status=sent (delivered to mailbox)
postfix/qmgr[19742]: 5707A9F8A: removed
```

Les messages générés **localement** (mailx, php, scripts, etc.) sont placés par **sendmail** dans la file d'attente **maildrop**.

Le message ayant été émis par la commande local mailx, c'est l'agent **pickup** qui a pris en charge le message placé dans maildrop.

L'agent **cleanup** a ensuite pris en charge le message, notamment en le fournissant au démon **trivial-rewrite** (non journalisé par défaut) pour permettre la réécriture des adresses de messagerie locale (FROM: root et TO: alice et TO: bob) en adresses conformes à la norme RFC 822 (FROM: [root@mail.formatux.lan⁴](mailto:root@mail.formatux.lan), TO: [alice@mail.formatux.lan⁵](mailto:alice@mail.formatux.lan) et TO: [bob@mail.formatux.lan⁶](mailto:bob@mail.formatux.lan)).

Cleanup a ensuite déposé le message dans la file d'attente **incoming**. De la file d'attente **incoming**, le message est passé dans la file active (**queue active**) par l'agent **qmgr**.

⁴ <mailto:root@mail.formatux.lan>

⁵ <mailto:alice@mail.formatux.lan>

⁶ <mailto:bob@mail.formatux.lan>

Le message ayant une portée local, il est transmis à l'agent **local** pour stockage dans la boîte aux lettres d'Alice puis de nouveau transmis à l'agent **local** pour stockage dans la boîte aux lettres de Bob.

qmgr supprime alors le message de la file **active**.

Utilisation interactive de mailx

```
[alice]$ mailx
Heirloom Mail version 12.4 7/29/08. Type ? for help.
"/var/spool/mail/alice": 2 messages 2 new
>N 1 test@mail.formatux.lan ...      ... "Test de mail"
  N 2 stagiaire                      "coucou"
& _
```

La première ligne identifie la version de mail utilisée.

La deuxième désigne la boîte aux lettres.

Le N (new) placé au début de la ligne indique qu'il s'agit d'un nouveau message, tandis que la lettre U (unread) indique qu'elle n'a pas encore été lue lors de la session précédente du programme mailx.

Pour lire le mail, il faut saisir après l'espérillette (et commercial) le numéro du mail à lire.

Pour quitter, simplement saisir la lettre q.

Lorsque la touche q est saisie, mailx sauvegarde le contenu de la boîte aux lettres dans le fichier mbox du répertoire personnel de l'utilisateur, ainsi que les éventuelles modifications ou suppressions effectuées.

Le fichier /home/alice/mbox doit maintenant contenir les messages qui ont été lus.

Tableau 7.3. Gestion des mails avec mailx

Options	Information
num	Affiche le mail n° num
d num	Supprime le mail num
h	Affiche la liste des mails

Options	Information
q	Quitte mailx

7.4.3. La commande swaks

La commande swaks (SWiss Army Knife for Smtip) est un outil de test orienté transaction pour SMTP.

Syntaxe de la commande swaks.

```
swaks --to user@formatux.lan --server smtp.formatux.lan
```

7.5. Configuration du serveur

La commande **postconf** permet la configuration de Postfix.

Syntaxe de la commande postconf.

```
postconf [-d] [-e] [-n] ['directive']
```

Exemple :

```
postconf -e 'myhostname = mail.formatux.fr'
```

Tableau 7.4. Options principales de la commande postconf

Options	Information
-d	Affiche les valeurs par défaut des paramètres
-e	Modifie le fichier main.cf avec le paramètre précisé.
-n	Affiche seulement les valeurs qui ne sont pas celles par défaut.

Utilisé sans option ni argument, la commande postconf affiche la configuration courante de Postfix.

La commande postfix check vérifie la configuration du fichier **main.cf** :

```
[root]# postfix check
```

Lorsque la configuration est correcte, la commande ne retourne aucune information.

7.5.1. Les alias

Comment rediriger une adresse vers une autre ? Comment créer une liste de diffusion ? Comment créer une adresse en prenom.nom ? En utilisant les **alias** !

Les alias sont contenus dans le fichier /etc/aliases :

/etc/aliases.

```
...
postmaster:    root
...
# Person who shold get root's mail
root:          bob
# Alias locaux
bob.leponge:   bob
# Liste de diffusion
admins:        bob,alice
```

Les modifications sont prises en compte avec la commande **newaliases** :

```
[root]# newaliases
```

Suivi des mails

Suivi d'un mail généré avec mailx à root :

```
postfix/local[25354]: 12C969F84F: to=<bob@mail.formatux.lan>,
 orig_to=<root>, relay=local, ..., status=sent (delivered to mailbox)
```

Suivi d'un mail généré avec mailx à admins :

```
postfix/local[25639]: 8DD1A9F84F: to=<bob@mail.formatux.lan>,
 orig_to=<admins>, relay=local, ..., status=sent (delivered to mailbox)
postfix/local[25639]: 8DD1A9F84F: to=<alice@mail.formatux.lan>,
 orig_to=<admins>, relay=local, ..., status=sent (delivered to mailbox)
```

Suivi d'un mail généré avec mailx à bob.leponge :

```
postfix/local[25920]: 0CD8B9F84F: to=<bob@mail.formatux.lan>,
 orig_to=<bob.leponge>, relay=local, ..., status=sent (delivered to
 mailbox)
```

7.5.2. Configurer un serveur relais

Mon serveur est protégé par une DMZ ! Mon fournisseur d'accès bloque le protocole SMTP ! Comment faire ?

Lorsque le serveur n'est pas directement connecté à Internet, il faut utiliser un serveur relais !

Les messages à destination d'utilisateurs non locaux sont relayés par le serveur MTA de la DMZ ou le MTA du fournisseur d'accès.

Configuration du relais

```
[root]# postconf -e 'relayhost = [svrmail.formatux.lan]'
[root]# service postfix reload
[root]# mailx bob.leponge@free.fr
```

Suivi du mail :

/var/log/maillog.

```
postfix/smtp[26595]: 0D7809F84F:
 to=<bob.leponge@free.fr>, relay=svrmail.formatux.lan[XXX.XXX.XXX.XXX]:25,
 ..., status=sent (250 2.0.0 Ok: queued as 2997B686008)
```

Lorsqu'un serveur n'est pas **directement** connecté à l'Internet, les mails qu'il émet devront être envoyés à un serveur intermédiaire : un **MTA relais**.

Il faut alors renseigner la directive **relayhost**.



Pour ne pas utiliser de résolution DNS sur le champ MX du domaine, il convient de mettre le FQDN ou l'adresse IP du serveur relais entre crochets.

Dans l'exemple au dessus, le serveur 'svrmail.formatux.lan' étant le serveur de messagerie pointé par l'enregistrement MX du DNS pour le domaine

'formatux.lan', les deux configurations suivantes sont identiques, mais la deuxième affranchit le serveur d'une requête DNS :

```
[root]# postconf -e 'relayhost = formatux.lan'
```

est identique à :

```
[root]# postconf -e 'relayhost = [svrmail.formatux.lan]'
```

Le message est cette fois-ci transmis par l'agent **mgr** au démon **smtp**, chargé de transférer le message via le protocole **SMTP**.

7.5.3. Prendre en compte un domaine

Je veux transformer mon serveur de messagerie, je voudrais centraliser les messages pour tout le domaine formatux.lan !

Il faut configurer les directives **mydomain** et **mydestination** !

Avant de modifier la configuration du serveur, envoyer un mail à bob@formatux.lan⁷ :

```
[root]# mailx bob@formatux.lan
```

Suivi du message :

```
postifx/smtp[27446]: B522C9F84F:  
to=<bob@formatux.lan>, relay=svrmail.formatux.lan[172.16.160.7]:25, ...,  
status=sent (250 2.0.0 Ok: queued as CB201686008)
```

Le serveur ne sait pas pour l'instant qu'il doit délivrer à l'agent local ce message.

Les directives **mydomain** et **mydestination**

La directive **mydomain** contient le nom de domaine internet du système de messagerie. Par défaut, \$mydomain vaut \$myhostname obtenu de son premier composant.

⁷ <mailto:bob@formatux.lan>

Si \$myhostname vaut serveur.formatux.lan alors \$mydomain vaut formatux.lan.

La directive **mydestination** liste les domaines livrés par l'agent local.

Pour visualiser la valeur par défaut de la directive mydestination :

```
[root]# postconf 'mydestination'  
mydestination = $myhostname, localhost.$mydomain, localhost
```

Le serveur transmettra donc à l'agent local tous les mails correspondant à @serveur.formatux.lan, @localhost.formatux.lan et @localhost.

Pour ajouter le domaine formatux.lan à la liste des domaines gérés localement :

```
[root]# postconf -e 'mydestination = $myhostname, localhost.$mydomain,  
localhost, $mydomain'  
[root]# postconf -e 'mydomain = formatux.lan'  
[root]# service postfix reload
```

Le même test que précédemment peut être rejoué :

```
mailx bob@formatux.lan
```

Suivi du mail :

/var/log/maillog.

```
postfix/local[28603]: AE6D99F84F: to=<bob@formatux.lan>, relay=local, ...,  
status=sent delivered to mailbox)
```

Le message est cette fois-ci pris en compte par le serveur et délivré à une boîte aux lettres locale via le démon **local**.

7.5.4. La directive mynetworks

Par défaut, le serveur postfix refuse de prendre en compte des messages provenant du réseau (excepté depuis sa loopback localhost), ce qui aurait pour effet de devenir un serveur OpenRelay à la merci des spammers.

Maintenant que les clients du réseau local disposent d'une boîte mails, ils vont devoir envoyer leur messages au serveur. Ils doivent donc avoir accès à postfix via le réseau.

Il faut configurer postfix pour qu'il accepte les connexions réseaux, tout en prenant soin de le limiter aux connexions du réseau local.

Dans un premier temps, il convient d'autoriser postfix à écouter sur toutes les interfaces réseaux :

```
[root]# postconf -e 'inet_interfaces = all'
```

Dans un second temps, il faut vérifier que le firewall autorise les connexions (au moins sur le port 25).

La directive **mynetworks** précise les réseaux autorisés à envoyer des messages sur le serveur.

```
mynetworks = 192.168.96.0/19
```

La directive **mynetworks_style** est ignorée si **mynetworks** est définie. Sinon elle précise le type d'hôtes autorisés à envoyer des messages au serveur (host, subnet ou class).

```
# autoriser tous les hôtes de mon sous-réseau  
mynetworks_style = subnet
```



Attention à ne pas devenir un serveur “open-relay”, et ainsi servir de serveur relais pour les spameurs.

7.5.5. Le format de stockage

Par défaut, postfix stocke les mails au format mbox.

Pour passer du format mbox au format maildir :

```
[root]# postconf -e 'home_mailbox = Maildir/'  
[root]# service postfix reload
```

Le changement pourra être vérifié :

```
[root]# mailx bob@formatux.lan  
[root]# ls /home/bob/Maildir/new/
```

7.5.6. La table de routage

J'aimerais que les messages vers monvoisin.fr ne passe pas par le serveur relais. Il faudrait définir une route spécifique !

La table de routage définit un serveur relais pour un domaine donné en renseignant la directive **transport_maps** du fichier par défaut `/etc/postfix/transport`.

/etc/postfix/transport.

```
monvoisin.fr      smtp:[172.16.96.100]:25
```

Postfix doit prendre en compte cette nouvelle route :

```
[root]# postmap /etc/postfix/transport  
[root]# service postfix reload
```

Dans ce cas, le serveur de messagerie transmettra les messages directement au serveur indiqué, sans tenir compte d'une requête DNS ou du serveur relais par défaut.

Après avoir modifié la table des transports, il est nécessaire de lancer la commande **postmap**.

Cette commande permet de transformer le fichier en base de données type clef:valeur interprétable par postfix.

N'oubliez pas de relancer le service postfix.

7.6. Protocoles POP/IMAP

Le serveur Dovecot est un serveur POP3/IMAP orienté vers la sécurité.

Installer le serveur dovecot :

```
[root]# yum install dovecot  
[root]# chkconfig dovecot on
```

La configuration de dovecot est répartie entre de nombreux fichiers de configuration.

- Activer le protocole IMAP.
- Le serveur dovecot écoutera sur toutes les interfaces réseaux.

/etc/dovecot/dovecot.conf.

```
protocols = imap  
listen= *
```

- Spécifier à dovecot que le serveur Postfix stocke les mails au format Maildir (maildir:) dans le répertoire Maildir du répertoire de connexion de l'utilisateur (~/).

/etc/dovecot/conf.d/10-mail.conf.

```
mail_location = maildir:~/Maildir
```

- L'authentification plaintext permet de gérer l'authentification des clients par login et mot de passe qui transitent en clair sur le réseau. Cette option n'est pas sécurisée si elle n'est pas couplée avec un moyen de chiffrement du flux imap (IMAPS) ce qui explique sa désactivation par défaut.

/etc/dovecot/conf.d/10-auth.conf.

```
disable_plaintext_auth = no # signifie authentification plaintext enable
```

Redémarrer le service

```
[root]# service dovecot restart
```

7.7. Architecture de postfix

Postfix est un serveur modulaire basé sur des boîtes aux lettres et régi par le démon principal **master**.

Chaque démon assume une fonction, chaque fonction correspondant à une tâche distincte. Les démons sont gérés par le démon master, qui est le premier à être lancé.

Les messages sont stockés dans des files d'attente, où ils sont récupérés par les démons.

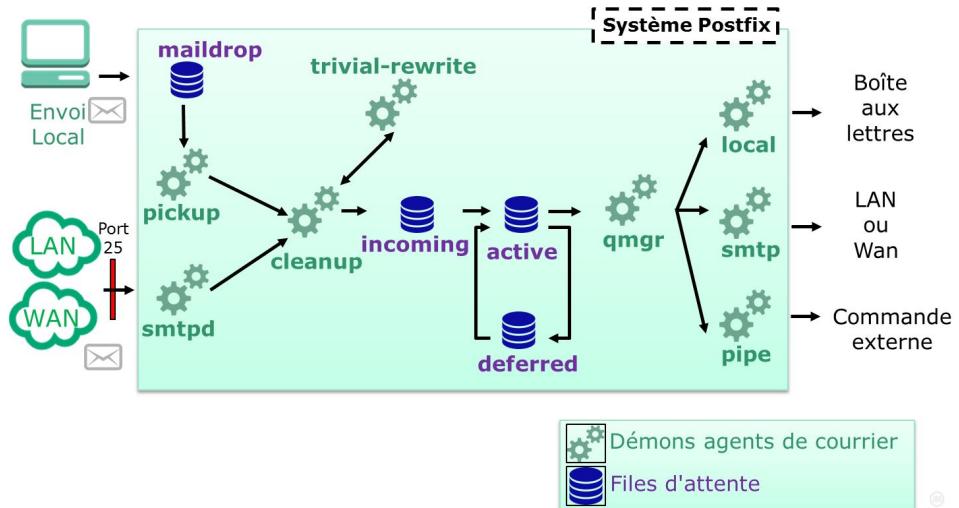


Figure 7.10. Synoptique global de postfix

Postfix accepte des messages provenant de plusieurs sources :

- Source locale : envoyé par un utilisateur du serveur via un logiciel local (mailx, php, etc.) ;
- En provenance du réseau connecté au serveur ;
- Produit par Postfix lui-même ;
- Un message ressoumis pour être transféré à une autre adresse.

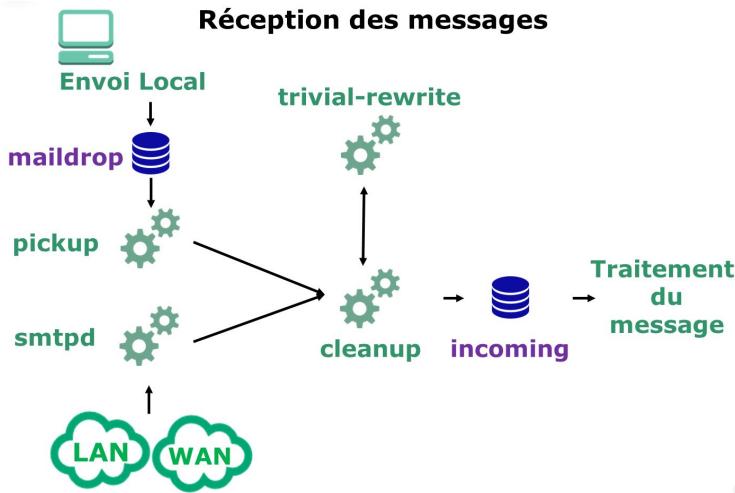


Figure 7.11. Synoptique de postfix partie réception des messages

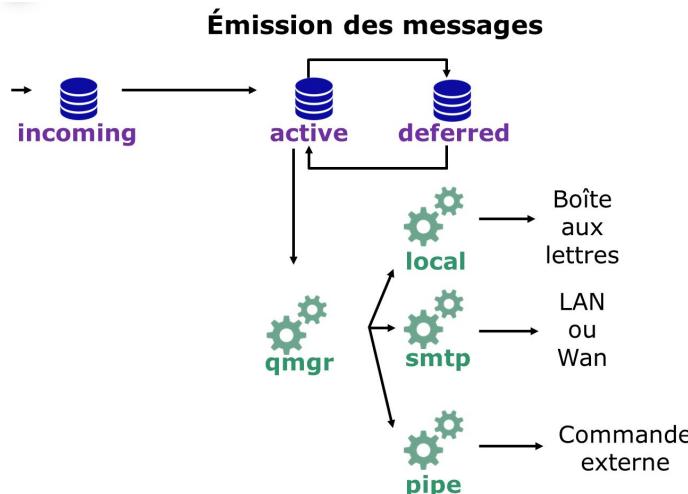


Figure 7.12. Synoptique de postfix partie émission des messages

Postfix produit lui-même les messages de service pour indiquer la non-réception d'un message ou son report. Ces messages suivent le même cheminement que les messages locaux.

Les messages locaux sont déposés dans la file d'attente maildrop.

Le démon pickup prend les messages dans la file d'attente et les passe à cleanup.

Les messages provenant du réseau sont pris en charge par le démon `smtpd`. Ce dernier vérifie qu'ils vont pouvoir être traités par le serveur et ensuite il les transfère à `cleanup`.

Un message se doit de respecter certaines normes de formatage. Les adresses de provenance ou de destination doivent être pleinement qualifiées, il ne doit pas manquer d'en-têtes, etc. Un message ne respectant pas ces règles sera reformaté (remis en forme) par `trivial-rewrite`. Une fois corrigé, il sera de nouveau pris en charge par `cleanup` qui le placera dans la file `incoming` et préviendra le gestionnaire `qmgr`.

Le gestionnaire de file d'attente `qmgr` effectue l'essentiel du traitement du courrier. Il gère les files d'attente `incoming`, `active` et `deferred`.

Après traitement par `cleanup`, les messages sont placés dans la file `incoming`. Si les ressources systèmes sont disponibles, `qmgr` déplace alors le message dans la file `active` et appelle l'un des agents de distribution pour le délivrer.

Les messages qui ne peuvent pas être distribués sont mis dans la file `deferred` où ils attendent que `qmgr` tente de nouveau de les distribuer.

Si le message n'est pas à destination d'un utilisateur géré par le serveur, `qmgr` le transfère au démon `smtp` qui l'expédie au MTA concerné.

Si le destinataire fait bien partie du domaine géré par le serveur Postfix, le démon `qmgr` achemine le message vers `local`.

Le démon `local` dépose les messages dans l'espace de stockage local des messages. Il contrôle également les alias et vérifie si les messages doivent être délivrés ailleurs.

Le message peut être délivré à un autre processus, comme un gestionnaire de liste de diffusion, ou tout autre processus.

7.7.1. Démons agents de courrier

- **pickup⁸**: collecteur de messages locaux acheminés par `maildrop`. Il fournit à l'agent `cleanup` les messages déposés dans la file `maildrop`.

⁸ <http://www.postfix.org/pickup.8.html>

- **smtpd⁹** : collecteur de messages reçus du réseau. L'agent smtpd accepte les connexions réseaux et effectue les transactions SMTP pour fournir les messages à l'agent cleanup.
- **trivial-rewrite¹⁰** : agent de résolution et de réécriture des adresses (en lien avec le domaine). L'agent trivial-rewrite offre 3 types de services aux requêtes des clients :
 - Réécriture du contexte d'adresse vers une forme standard : ajoute le nom de domaine spécifié par \$myorigin ou par \$mydomain aux adresses incomplètes des messages postés localement,
 - Résolution de l'adresse en un quadruple (transport, saut suivant, récipiendaire, drapeaux) :
 - Le transport correspond à l'agent de délivrance à utiliser,
 - Le MTA suivant à qui délivrer le mail,
 - L'adresse du destinataire à fournir au prochain MTA,
 - Les drapeaux : la classe d'adresse, si l'adresse nécessite un relais, si l'adresse a un problème ou si la requête a échoué.
 - Résolution de l'adresse de l'envoyeur (pour des besoins de vérifications).
- **cleanup¹¹** : agent de formatage des messages selon la norme RFC822. Il traite les messages entrant, les insère dans la file d'attente incoming puis informe qmgr de leur arrivée.
 - L'agent cleanup opère toujours ces transformations :
 - Ajout des en-têtes manquantes : From:, To:, Message-Id: et Date:.
 - Transforme au besoin les adresses de l'enveloppe et des en-têtes au standard utilisateur@fqdn qui est attendu par les autres agents postfix. Cette tâche est déléguée à l'agent trivial-rewrite.
 - Supprime les adresses dupliquées de l'enveloppe,
 - Supprime les en-têtes : Bcc:, Content-Length:, Resent-Bcc, Return-Path:..

⁹ <http://www.postfix.org/smtpd.8.html>

¹⁰ <http://www.postfix.org/trivial-rewrite.8.html>

¹¹ <http://www.postfix.org/cleanup.8.html>

- Optionnellement, les adresses peuvent être transformées en fonction de la table canonical ou virtual et du paramètre masquerade_domains,
- **qmgr**¹²: agent de gestion des files d'attente *active* et *deferred*. L'agent qmgr attend l'arrivée de message entrant et s'assure de leur livraison via un des agents de livraison. La stratégie de routage des messages est délégué au démon trivial-rewrite.
 - qmgr maintient les files d'attente incoming, active, deferred, corrupt et hold.
 - qmgr surveille les rapports de livraison par message dans les répertoires suivant et demande aux agent concernés d'envoyer les rapports :
 - bounce : rapport des messages refusés. Cette file est maintenue par l'agent bounce.
 - defer : rapport des messages retardés. Cette file est maintenue par l'agent bounce.
 - trace : rapport des messages suivis. Cette file est maintenue par l'agent trace.
- **local**¹³ : agent de livraison des messages locaux, MDA. L'agent local met à jour les files d'attente des messages et marque les messages si finis ou informe qmgr si les messages doivent être retraités plus tard. Les messages de livraison sont transmis à l'agent approprié (bounce, defer ou trace).
- **smtp**¹⁴ : agent de livraison des messages vers le réseau. Il implémente les protocoles SMTP et LMTP. Il procède à la livraison des messages à la demande de qmgr. L'agent met à jour les files d'attente des messages et marque les messages si finis ou informe qmgr si les messages doivent être retraités plus tard. Les messages de livraison sont transmis à l'agent approprié (bounce, defer ou trace).
 - L'agent smtp interroge le service DNS pour obtenir une liste d'adresses de serveurs de messagerie MX du domaine du destinataire de message, les trie par ordre de préférence et tente une connection vers chacun jusqu'à ce qu'il en trouve un qui réponde.

¹² <http://www.postfix.org/qmgr.8.html>

¹³ <http://www.postfix.org/local.8.html>

¹⁴ <http://www.postfix.org/smtp.8.html>

- **pipe**¹⁵ : agent de livraison des messages vers une commande externe.
- **bounce**¹⁶ : agent de suivi des messages (informations sur la délivrance des messages). Ce démon procède à deux types de requêtes :
 - Ajoute un enregistrement de (non-)délivrance à un fichier de suivi de message (un fichier par message).
 - Génère un message de notification de délivrance, avec une copie du fichier de suivi du message correspondant. Lorsque le message est correctement généré, le fichier de suivi est supprimé.

7.7.2. Files d'attente

- **maildrop** : messages locaux postés par *sendmail*.
- **incoming** : messages après formatage en attente de traitement. Tous les messages entrant dans le système Postfix sont écrits par l'agent cleanup dans la file incoming. Les nouveaux fichiers sont créés avec comme propriétaire l'utilisateur "postfix" avec des droits à 0600. Une fois que le fichier est prêt à être traité, l'agent cleanup change les droits à 0700 et notifie qmgr d'une nouvelle arrivée. Les messages qui n'ont pas les droits à 0700 sont tout simplement ignorés car considérés comme en cours d'écriture par cleanup.
- **active** : messages prêts à être acheminés. La file d'attente active n'est pas uniquement un ensemble de fichiers sur le disque. La file d'attente "réelle" active comprend également un ensemble de structures de données dans la mémoire de l'agent gmgr, ce qui explique que la quantité de message traités dans la file active soit limitée, pour éviter un dépassement de mémoire libre.
- **deferred** : messages n'ayant pas pu être livrés et pour lesquels un envoi ultérieur pourrait réussir.

7.8. Boîtes aux lettres virtuelles

Est-il vraiment utile de créer un compte système Linux pour chaque adresse de messagerie ?

¹⁵ <http://www.postfix.org/pipe.8.html>

¹⁶ <http://www.postfix.org/bounce.8.html>

Il est possible d'héberger la messagerie de différents domaines sans associer les boîtes aux lettres à des comptes système.

Les boîtes seront stockées (par exemple) sous /var/mail/vmail et gérées par l'utilisateur vmail (uid=5000, gid=5000).

Créer l'utilisateur virtual mailbox:

```
[root]# groupadd -g 5000 vmail
[root]# useradd vmail -u 5000 -g 5000 -s /sbin/nologin -d /var/mail/
vmail
```

/etc/postfix/main.cf.

```
virtual_mailbox_domains = mondomaine.com, autre.com
virtual_mailbox_base = /var/mail/vmail
virtual_mailbox_maps = hash:/etc/postfix/vmailbox
virtual_minimum_id = 100
virtual_uid_maps = static:5000
virtual_gid_maps = static:5000
virtual_alias_maps = hash:/etc/postfix/virtual
```



Ne jamais lister ici un domaine renseigné dans la directive mydestination ou virtual_alias_domain.

/etc/postfix/vmailbox.

```
bob@mondomaine.com mondomaine.com/bob/
alice@mondomaine.com mondomaine.com/alice/
bob@autre.com autre.com/bob/
```

Le / à la fin des chemins vers les boîtes aux lettres précise qu'ici le format de stockage est Maildir.

/etc/postfix/vmailbox.

```
postmaster@mondomaine.com postmaster
```

Postfix ne peut pas traiter directement les fichiers vmailbox et virtual dans leur format humain. Il a besoin de générer une base de données au format clé-valeur, plus couramment appelée hash-table.

Générer les tables clés/valeurs (hachées) :

```
[root]# postmap /etc/postfix/vmailbox
[root]# postmap /etc/postfix/virtual
```

Ce rôle est rempli par la commande postmap, qui dans notre exemple, générera deux fichiers : vmailbox.db et virtual.db.

Créer les répertoires de stockage :

```
[root]# su - vmail -s /bin/bash
[vmail]$ mkdir /var/mail/vmail/mondomaine.com/
[vmail]$ mkdir /var/mail/vmail/autre.com/
```

Authentification des utilisateurs :

/etc/dovecot/users.

```
bob@mondomaine.com:{PLAIN}password:5000:5000:::/var/mail/vmail/
mondomaine.com/bob/
alice@mondomaine.com:{PLAIN}password:5000:5000:::/var/mail/vmail/
mondomaine.com/alice/
```

Authentification par fichier plat :

/etc/dovecot/conf.d/10-auth.conf.

```
disable_plaintext_auth = no
!include auth-passwdfile.conf.ext
```

Nouvel emplacement de stockage :

/etc/dovecot/conf.d/10-mail.conf.

```
mail_location = maildir:/var/mail/vmail/%d/%n
```

Les macros dovecot :

Dovecot est capable de remplacer dynamiquement des valeurs renseignées dans ses fichiers de configuration.

Dans notre exemple, le %d sera remplacé par le domaine de messagerie et le %n par le nom de la boîte aux lettres. Par exemple, un mail destiné à bob@mondomaine.lan¹⁷ sera stocké dans le sous-dossier mondomaine.lan/bob/.

Les comptes de messagerie ne nécessitent plus de compte système, ce qui facilite l'administration et améliore la sécurité.

Les fichiers plats utilisés dans nos exemples peuvent facilement être remplacés par une table mysql ou un annuaire LDAP.



Pourquoi ne pas utiliser uniquement des utilisateurs virtuels dans ce cas ?

7.9. Suivi des messages à des fins légales

Il peut être demandé de conserver le sujet, le rédacteur et le destinataire d'un message.

Pour cela, le processus cleanup doit vérifier les entêtes des messages et générer un log lorsqu'il rencontre la valeur attendue. Ces valeurs sont stockées dans le fichier /etc/postfix/header_checks sous forme de regex :

/etc/postfix/header_checks.

```
/^subject:/      WARN
/^Subject:/     WARN
/^to:/          WARN
/^To:/          WARN
/^from:/        WARN
/^From:/        WARN
```

Postfix utilise la directive **header_checks** :

```
postconf -e 'header_checks = regexp:/etc/postfix/header_checks'
service postfix restart
```

Un message généré avec la commande swaks :

¹⁷ <mailto:bob@mondomaine.lan>

```
swaks --to alice@formatux.lan --from bob@formatux.lan --header "Subject:  
test test test" --server 127.0.0.1
```

Générera les logs suivants :

```
tail -f /var/log/maillog | grep warning  
postfix/cleanup[13423]: 125D162F88: warning: header To:  
alice@formatux.lan [...]  
postfix/cleanup[13423]: 125D162F88: warning: header From:  
bob@formatux.lan [...]  
postfix/cleanup[13423]: 125D162F88: warning: header  
Subject: test test test [...]
```

8

Serveur d'annuaire OpenLDAP

8.1. Généralités

Le protocole **LDAP (LightWeight Directory Access Protocol)** est une suite **standardisée** de protocoles permettant l'accès à un annuaire centralisé. Cet annuaire centralisé stocke des informations diverses comme :

- des noms ;
- des adresses ;
- des numéros de téléphone ;
- des utilisateurs ;
- des groupes ;
- etc.

La version actuelle de ce protocole est la version 3.

Le protocole LDAP s'inspire de la spécification **X.500** mais de manière moins complexe. La norme X.500 désigne l'ensemble des normes informatiques sur les services d'annuaires définies par l'IUT (International Union Telecommunication). Il s'agit donc d'un annuaire version électronique dont l'organisation est hiérarchique, à la manière du DNS, qui correspond généralement à l'organisation de l'entreprise. Le terme Lightweight ne doit pas être compris comme "allégé", ce qui signifierait qu'il serait amputé de certaines fonctionnalités, mais bien dans ce cas de "simplifié", car la spécification X.500 est très lourde.

Le système LDAP est né en 1993 à l'université du Michigan. Le dérivé libre OpenLDAP est lui apparu en 1998.

Une base de données LDAP, de part sa nature, est optimisée pour la lecture d'informations :

- authentification ;
- recherche dans l'annuaire.

Les ports utilisés par le protocole LDAP sont les ports **389** (en clair comme en chiffré par **startTLS**) et **636** pour une connexion en TLS (solution dépréciée).

L'implémentation la plus célèbre du protocole LDAP sous Windows est l'Active Directory.

Sous Linux, les choix sont nombreux :

- OpenLDAP ;
- RedHat Directory Studio ;
- Samba4 qui intègre un serveur LDAP ;
- LinID de Linagora ;
- 389 DS ;
- IBM Tivoli ;
- ...



À noter que la suite OpenLDAP au sein de RedHat 6 (et versions supérieures) n'utilise plus OpenSSL. Elle utilise à la place l'implémentation de Mozilla de NSS (Network Security Services).

8.1.1. La DIT

Un annuaire LDAP est **un arbre de données**. Cette **structure hiérarchique** est appelée **DIT (Directory Information Tree)**.

Une entrée de l'arbre est un ensemble d'**attributs**.

Chaque entrée dispose d'un identifiant unique : son **DN (Distinguished Name)**.

8.1.2. L'OLC

OpenLDAP est le serveur d'annuaire de référence pour les distributions Linux.

Dans les versions récentes d'OpenLDAP (>2.4), sa configuration n'est plus stockée dans un fichier de configuration.

La configuration réside directement dans la base de données elle-même, au sein d'une **DIT** spécifique : c'est la fonctionnalité **OLC (On-Line Configuration)**, aussi connue sous le nom **cn=config**.

L'approche consistant à stocker 'en ligne' la configuration LDAP peut paraître complexe, mais est justifiée par la criticité du service LDAP. Stocker la configuration dans des fichiers plats imposait un redémarrage du service à chaque modification, ce qui représentait beaucoup de temps d'arrêt pour de grosses bases de données.



Il n'y a plus de fichiers .conf à modifier dans les versions récentes d'OpenLDAP.

8.1.3. Le schéma

Le contenu des entrées d'un annuaire est régi par des schémas. Les schémas définissent les types d'attributs d'une entrée regroupés par classe d'objets.

- schéma : ensemble des classes et des attributs disponibles.
- objectClass : une classe objet rassemble un ensemble d'attributs obligatoires ou facultatifs (par exemple la classe `inetOrgPerson`).
- attribut : exemple
 - mail: john.doe@formatux.lan¹ ;
 - preferredLanguage: french.

8.1.4. SASL

SASL (Couche d'Authentification et de Sécurité Simple) est une méthode pour ajouter le support d'**authentification** aux **protocoles basés sur la connexion** (LDAP, SMTP, IMAP, XMPP, IRC, etc.). Pour utiliser SASL, un protocole inclut une **commande d'identification et d'authentification** d'un utilisateur sur un serveur

¹ <mailto:john.doe@formatux.lan>

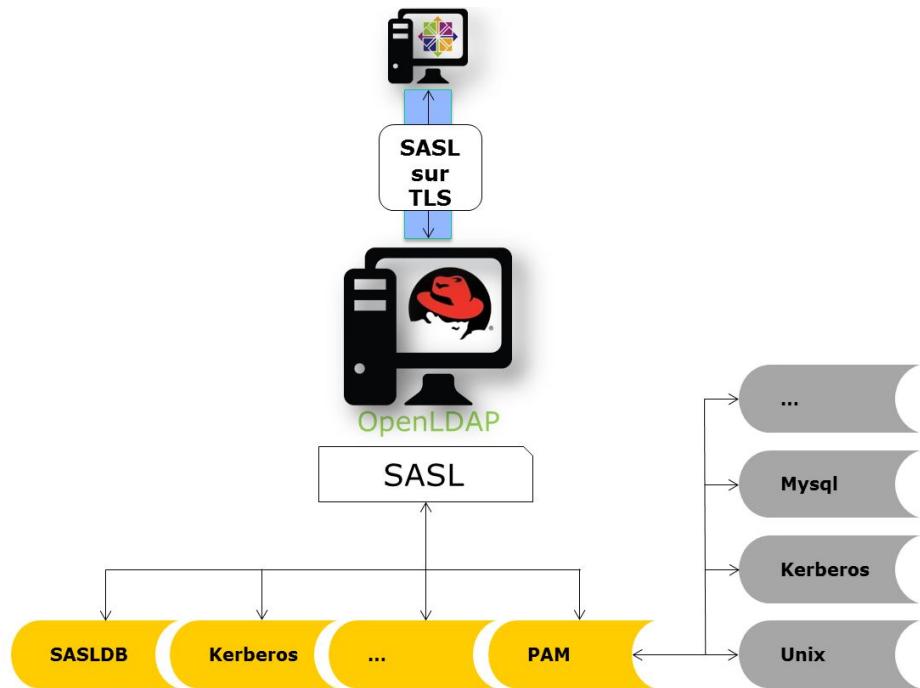
et la **négociation éventuelle de la protection** des interactions consécutives du protocole.

Si son utilisation est négociée, **une couche de sécurité est insérée entre le protocole et la connexion**.

Séparer ainsi la couche d'authentification de la couche applicative permet en théorie à n'importe quel mécanisme d'authentification pris en charge par SASL d'être employé à partir de n'importe quel protocole d'application capable d'utiliser SASL.

Les mécanismes principaux SASL sont :

- **EXTERNAL** : l'authentification est dérivée du contexte (authentification système);
- **ANONYMOUS** : accès anonyme sans authentification ;
- **PLAIN** : mot de passe en clair ;
- **OTP** : mot de passe unique (One Time Password) ;
- **CRAM-MD5 et DIGEST-MD5** : basés sur MD5 ;
- **NTLM** : authentification pour réseau local NT ;
- **GSSAPI** : authentification Kerberos via GSSAPI.



8.1.5. Le format LDIF

Le format LDIF (LDAP Data Interchange Format) est un format de fichier texte utilisé lors des échanges d'informations en client/serveur ou entre serveurs.

Exemple de fichiers LDIF :

```

dn: cn=John Doe,dc=example,dc=com
objectClass: top
objectClass: person
objectClass: inetOrgPerson
objectClass: organizationalPerson
cn: John Doe
givenName: John
sn: Doe
mail: john.doe@example.com

```

Dans cette entrée, nous retrouvons, outre le Distinguished Name (DN) de l'objet :

- 4 objectClass : inetOrgPerson, organizationPerson, person et top ;
- 4 attributs : cn, givenName, sn et mail.

Les objectClass permettent d'inclure des attributs obligatoires ou optionnels dans une entrée. Toujours dans notre exemple, c'est l'ajout de l'objectClass inetOrgPerson qui va permettre d'ajouter un attribut mail.

L'ensemble des objectClass et des attributs sont définis dans des schémas, qu'il conviendra d'ajouter en fonction du rôle du serveur. Par exemple, pour permettre l'authentification Samba depuis le serveur LDAP, il faudra ajouter le schéma samba.schema à la configuration du serveur.



Le format LDIF a des caractéristiques très importantes :

- les séries d'entrées sont séparées par des lignes vides ;
- la dernière ligne doit être vide, sinon la dernière entrée pourrait ne pas être prise en compte.

8.1.6. Les outils clients LDAP

Des outils en ligne de commande permettent l'administration du serveur en utilisant en entrée des fichiers LDIF ou la console.

- ldapadd : ajouter des entrées ;
- ldapdelete : supprimer des entrées ;
- ldapmodify : modifier des entrées ;
- ldappasswd : modifier un mot de passe ;
- ldapsearch : rechercher dans l'annuaire.

8.2. Installation du serveur

Prérequis à l'installation :

- disposer des droits root ou sudo ;
- disposer d'un dépôt yum configuré ;
- avoir ouvert les ports 389 et 636 sur le parefeu local et sur les éléments actifs

Installation :

```
[root]# yum install openldap-servers openldap-clients
```

```
[root]# cp /usr/share/openldap-servers/DB_CONFIG.example /var/lib/ldap/
DB_CONFIG
[root]# chown ldap:ldap /var/lib/ldap/DB_CONFIG
[root]# chkconfig slapd on
[root]# service slapd start
```

LDAP a besoin d'un fichier de configuration (/var/lib/ldap/DB_CONFIG) pour sa base de données. Le fichier fourni en exemple lors de l'installation convient parfaitement.

Le fichier /etc/openldap/ldap.conf contient la configuration pour les clients openldap comme Ldapsearch, Ldapadd, etc. Toutes les informations inscrites dans ce fichier allègeront d'autant les lignes de commande interactives, puisqu'il ne sera plus nécessaire de préciser les options positionnées ici.

Le fichier /etc/openldap/ldap.conf.

```
#  
# LDAP Defaults  
#  
  
#BASE dc=example,dc=com  
#URI ldap://ldap.example.com ldaps://ldap.example.com:666  
  
#SIZELIMIT 12  
#TIMELIMIT 15  
#DEREF never  
  
TLS_CACERTDIR /etc/openldap/certs
```

Le répertoire /etc/openldap/slapd.d/ contient les bases de données et le schéma :

Arborescence du service OpenLDAP.

```
# /etc/openldap/slapd.d/  
|-- cn=config  
|   |-- cn=schema # les schémas disponibles  
|   |   |-- cn={10}ppolicy.ldif  
|   |   |-- cn={1}core.ldif  
|   |   |-- cn={2}cosine.ldif  
|   |   |-- cn={5}inetorgperson.ldif  
|   |   |-- cn={8}nis.ldif  
|   |   |-- cn={9}openldap.ldif
```

```

|   |-- cn=schema.ldif # le schéma du serveur
|   |-- olcDatabase={0}config.ldif
|   |-- olcDatabase={-1}frontend.ldif
|   |-- olcDatabase={1}monitor.ldif
|   |-- olcDatabase={2}bdb.ldif # la DIT principale au format BDB
|-- cn=config.ldif # configuration globale du serveur

```



Les bases de données du serveur LDAP ne doivent jamais être modifiées manuellement !!!

8.3. Configuration du serveur

Avant de pouvoir utiliser les outils en ligne de commande, il convient de configurer les options par défaut :

```

BASE dc=formatux,dc=lan
URI ldap://localhost

```



En version TLS sécurisée, l'URI doit impérativement correspondre au FQDN renseigné dans le certificat !

Pour la suite du cours, nous retiendrons que :

- le dn de base est : dc=formatux,dc=lan ;
- l'administrateur LDAP est cn=admin,dc=formatux,dc=lan ;
- les utilisateurs sont stockés dans l'unité d'organisation : ou=users,dc=formatux,dc=lan.

Il est intéressant à ce stade de visualiser la configuration par défaut avec la commande slapcat :

```

[root]# slapcat -b cn=config | less
...
dn: olcDatabase={2}bdb,cn=config # base de données de l'annuaire
...
olcSuffix: dc=my-domain,dc=com # suffix par défaut
olcRootDN: cn=Manager,dc=my-domain,dc=com # administrateur par défaut

```

...



À noter que l'administrateur n'a pas de mot de passe (olcRootPW)

8.3.1. Le suffixe

Le suffixe représente la racine de l'organisation. C'est l'identité même de l'entreprise. Elle correspond habituellement au suffixe DNS.

Nous allons le changer avec la commande ldapmodify :

```
[root]# ldapmodify -Y EXTERNAL -H ldapi:///dn: olcDatabase={2}bdb,cn=configchangetype: modifyreplace: olcSuffixolcSuffix: dc=formatux,dc=lan
```



Le suffixe étant défini à l'installation, il faut le modifier !

8.3.2. Le RootDN et son mot de passe

L'entrée RootDN contient le DN de l'utilisateur autorisé à faire des modifications de l'annuaire.

Son mot de passe est défini par RootPW.

Nous allons le configurer avec la commande ldapmodify :

```
[root]# ldapmodify -Y EXTERNAL -H ldapi:///dn: olcDatabase={2}bdb,cn=configchangetype: modifyreplace: olcRootDNolcRootDN: cn=admin,dc=formatux,dc=lan
```



Le suffixe étant défini à l'installation, il faut le modifier !

Pour définir un mot de passe utilisable par openldap, il faut utiliser la commande slappasswd.

Ajouter le RootPW :

```
[root]# ldapmodify -Y EXTERNAL -H ldapi:///dn: olcDatabase={2}bdb,cn=configchangetype: modifyadd: olcRootPWolcRootPW: {SSHA}Eke0fnWgD90xZWPT/UivZEBjzBgC/Z+
```



Cette fois-ci, le RootPW n'ayant pas été défini à l'installation, il faudra l'ajouter!

Les trois commandes auraient pu être regroupées en une seule. Dans ce cas, il faut séparer chaque modification de l'objet par une ligne contenant un “-“.

```
[root]# ldapmodify -Y EXTERNAL -H ldapi:///dn: olcDatabase={2}bdb,cn=configchangetype: modifyreplace: olcSuffixolcSuffix: dc=formatux.lan-replace: olcRootDNolcRootDN: cn=admin,dc=formatux,dc=lan-add: olcRootPWolcRootPW: {SSHA}Eke0fnWgD90xZWPT/UivZEBjzBgC/Z+
```

8.3.3. Connexion avec le RootDN

Un RootDN et son mot de passe ayant maintenant été définis dans la DIT dc=formatux,dc=lan, il est possible de les utiliser pour se connecter :

```
[root]# ldapmodify -x -D cn=admin,dc=formatux,dc=lan -W
```



Il n'est pas nécessaire de préciser ici le serveur à contacter (options -H ou -h), la commande ldapmodify utilisera les informations du fichier /etc/openldap/ldap.conf qui a été renseigné précédemment.

8.3.4. La commande *slapcat*

Exporter le contenu de l'annuaire au format LDIF.

Syntaxe de la commande *slapcat*.

```
slapcat -b suffix
```

Exemple :

```
[root]# slapcat -b cn=config | less
...
dn: olcDatabase={2}bdb,cn=config
...
olcSuffix: dc=my-domain,dc=com
olcRootDN: cn=Manager,dc=my-domain,dc=com
...
```

Option	Description
-b	Détermine quelle base de données est exportée.

8.3.5. La commande *ldapmodify*

La commande *ldapmodify* permet de modifier le contenu de l'annuaire.

Authentification (*binding*) par SASL

Syntaxe de la commande *ldapmodify* avec authentification SASL.

```
ldapmodify [-y SASLMecanisme] [-H host] [-v] [-f fichier.ldif]
```

Exemple :

```
[root]# ldapmodify -Y EXTERNAL -H ldapi:/// -v -f modldap.ldif
```

Option	Description
-Y	Mécanisme SASL à utiliser pour l'authentification.
-v	Mode verbeux pour diagnostique.

Option	Description
-H	Spécifier un serveur. Le protocole Idapi permet une communication sécurisée via une socket UNIX (nécessaire pour utiliser SASL).

Authentification (binding) simple

Syntaxe de la commande ldapmodify avec authentification simple.

```
ldapmodify [-x] [-D RootDN] [-W|-w pwd] [-H host] [-f fichier.ldif]
```

Exemple :

```
[root]# ldapmodify -x -D cn=admin,dc=formatux,dc=lan -W
```

Option	Description
-x	Utiliser l'authentification simple au lieu de SASL
-D	BindDN à utiliser pour la connexion à la base.
-W ou -w	Demander le mot de passe (interactif ou non).
-f	Lire les modifications à effectuer depuis un fichier



Il n'est pas nécessaire de préciser le serveur à contacter (options -H ou -h) si celui-ci est renseigné dans le fichier /etc/openldap/ldap.conf

Exemples

The screenshot shows a LDAP browser window. On the left, the directory structure (DIT) is displayed with nodes like DIT, Root DSE, dc=etrs,dc=lan, ou=groups, ou=users, and cn=bleponge. On the right, a detailed view of the cn=bleponge object is shown in a table format:

Description d'attribut	Valeur
objectClass	person (structural)
objectClass	posixAccount (auxiliary)
objectClass	shadowAccount (auxiliary)
objectClass	top (abstract)
cn	bleponge
gidNumber	100
homeDirectory	/home/bleponge
sn	Leponge
uid	bleponge
uidNumber	10000
description	compte de bob leponge
gecos	bleponge
loginShell	/bin/bash
shadowLastChange	10877
shadowMax	9999

Il faut séparer les cas suivants :

- Ajouter/Supprimer un objet. Supprimer bleponge ou ajouter asaglisse.
- Modifier un objet en lui ajoutant, supprimant ou modifiant un attribut.

Ajouter un objet

```
[root]# ldapmodify -Y EXTERNAL -H ldapi:///dn:dn:delobjetaajouterchangetype: add...
```

Supprimer un objet

```
[root]# ldapmodify -Y EXTERNAL -H ldapi:///dn:dn:delobjetasupprimerchangetype: delete
```

Modifier un objet

- Ajouter un attribut :

```
[root]# ldapmodify -Y EXTERNAL -H ldapi:///
```

```
dn: dndelobjetamodifier
changetype: modify
add: nomdelattribut
nomdelattribut: nouvellevaleur
```

- Supprimer un attribut

```
[root]# ldapmodify -Y EXTERNAL -H ldapi:/// 
dn: dndelobjetamodifier
changetype: modify
delete: nomdelattribut
```

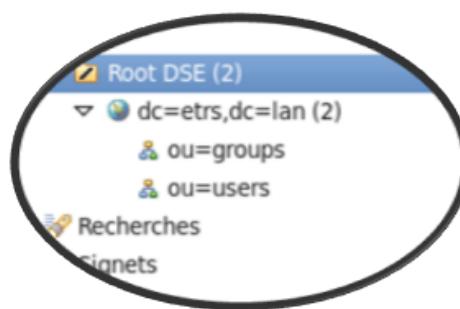
- Modifier un attribut

```
[root]# ldapmodify -Y EXTERNAL -H ldapi:/// 
dn: dndelobjetamodifier
changetype: modify
replace: nomdelattribut
nomdelattribut: nouvellevaleur
```

8.3.6. La structure de la DIT

Les données de l'arbre de l'annuaire doivent être rangées dans des unités d'organisation (OU).

Les OU **users** et **groups** sont généralement utilisées.



Commencer par ajouter une entrée dans l'annuaire correspondant à l'organisation de l'entité :

```
[root]# ldapmodify -x -D cn=admin,dc=formatux,dc=lan -W
Enter LDAP Password:
dn: dc=formatux,dc=lan
changetype: add
objectClass: dcObject
objectClass: organization
dc: formatux
o: formatux
description: Serveur formatux
```

Puis les deux OU concernées :

Le fichier /root/structure.ldif.

```
dn: ou=users,dc=formatux,dc=lan
changetype: add
objectClass: top
objectClass: organizationalUnit
ou: users
description: Utilisateurs de Formatux

dn: ou=groups,dc=formatux,dc=lan
changetype: add
objectClass: top
objectClass: organizationalUnit
ou: groups
description: Groupes d'utilisateurs de Formatux
```

```
[root]# ldapmodify -x -D cn=admin,dc=formatux,dc=lan -W -f /root/
structure.ldif
```

8.3.7. Activation des logs

Dans certains cas, il sera intéressant d'activer la journalisation dans la base cn=config.

Celle-ci étant très verbeuse, elle sera activée ou désactivée au besoin.

```
[root]# ldapmodify -Y EXTERNAL -H ldapi:///
```

```
dn: cn=config
changeType: modify
replace: olcLogLevel
olcLogLevel: stats
```

Il faut paramétrer le service syslog pour qu'il accepte les logs (niveau local4).

Le fichier /etc/rsyslog.conf.

```
local4.* /var/log/slapd.log
```

sans oublier de redémarrer le démon syslog :

```
[root]# service rsyslogd restart
```



Le logLevel 4 permet de journaliser les requêtes effectuées sur la base.

8.3.8. Activation du TLS

Avant de pouvoir configurer le TLS sous OpenLDAP, il convient de disposer du certificat et de la clef pour le serveur ainsi que le certificat de l'autorité de certification, qui est indispensable au bon fonctionnement d'OpenLDAP.

Pour créer ces certificats, il est possible d'utiliser easy-rsa, qui sera abordé dans la quatrième partie du document.

Une autre méthode est d'utiliser l'outil certtool du paquet gnutls-utils.



Si l'accès au serveur LDAP se fait via le FQDN ldap.formatux.lan, il faudra impérativement créer le certificat qui répondra à ce nom. Il ne sera plus possible par la suite de se connecter en LDAPS ou en starttls via l'adresse de loopback localhost.

Création des certificats avec certtools

Installer le paquet gnutls-utils :

```
[root]# yum install gnutls-utils
```

Dans le cas d'un certificat auto-signé, il faut dans un premier temps créer une clef privée pour l'autorité de certification :

```
[root]# certtool --generate-privkey --outfile /etc/pki/CA/private/ca-key.key
```

et décliner cette clef privée en certificat public.

```
[root]# certtool --generate-self-signed --load-privkey /etc/pki/CA/private/ca-key.key --outfile /etc/pki/CA/certs/ca-cert.pem
```

Il faut ensuite générer un certificat privé pour le serveur (ldap.formatux.lan par exemple)

```
[root]# certtool --generate-privkey --outfile /etc/pki/tls/private/ldap.key
```

Puis son certificat public signé par la clef privée de l'autorité de certification créée ci-dessus :

```
[root]# certtool --generate-certificate --load-privkey /etc/pki/tls/private/ldap.key --outfile /etc/pki/tls/certs/ldap.pem --load-ca-certificate /etc/pki/CA/certs/ca-cert.pem --load-ca-privkey /etc/pki/CA/private/ca-key.key
```

Prise en compte des certificats

Le fichier /root/tls.ldif.

```
dn: cn=config
changetype: modify
replace: olcTLSCertificateFile
olcTLSCertificateFile: /etc/pki/certs/ldap.pem
-
replace: olcTLSCertificateKeyFile
olcTLSCertificateKeyFile: /etc/pki/private/ldap.key
-
```

```
replace: olcTLSCACertificateFile
olcTLSCACertificateFile: /etc/pki/CA/certs/ca-cert.pem
```

```
[root]# ldapmodify -Y EXTERNAL -H ldapi:/// -f /root/tls.ldif
```

La chaîne de connexion du fichier /etc/openldap/ldap.conf doit également être mise à jour :

Le fichier /etc/openldap/ldap.conf.

```
BASE dc=formatux,dc=lan
URI ldaps://ldap.formatux.lan

TLS_CACERTDIR /etc/openldap/certs
TLS_REQCERT try
```

La commande **cacertdir_rehash** permet de créer un lien symbolique vers le certificat de la CA dont le nom correspond au hash de ce certificat. Ceci est nécessaire au fonctionnement d'openLDAP en TLS !

```
[root]# cacertdir_rehash /etc/pki/CA/certs
[root]# ls -l /etc/pki/CA/certs
-rw-r--r--. 1 root root 1281 4 déc. 10:52 ca-cert.pem
1rwxrwxrwx. 1 root root 11   4 déc. 10:54 ce6a8cab.0 -> ca-cert.pem
```

Par défaut, le service slapd n'écoute pas sur le port 636 (ldaps) et il faut privilégier le startTLS sur le port 389. Pour activer le ldaps :

Le fichier /etc/sysconfig/ldap.

```
SLAPD_LDAPS=yes
```

sans oublier de relancer le serveur :

```
[root]# service slapd restart
```

Tester la connexion

La commande openssl permet de tester la connexion uniquement sur le port 636 :

```
[root]# openssl s_client -connect ldap.formatux.lan:636 -showcerts
```

Le certificat de la CA

De nombreuses applications auront besoin du certificat de la CA.

Il est recommandé de le mettre à disposition des utilisateurs sur le serveur web du serveur LDAP :

```
[root]# cp /etc/pki/CA/certs/ca-cert.pem /var/www/html/
```

Le certificat est ainsi accessible via l'adresse <http://ldap.formatux.lan/cacert.pem>.

Configuration du PAM

PAM peut être configuré pour utiliser le service openldap avec la commande authconfig :

```
[root]# yum install nss-pam-ldapd
```

```
[root]# authconfig --enableldap --enableldapauth --ldapserver=ldap://
ldap.formatux.lan --ldapbasedn="ou=users,dc=formatux,dc=lan" --
enableldaptls --ldaploadcacert=http://ldap.formatux.lan/ca-cert.pem --
enablemkhomedir --update
```

Création des utilisateurs

Création du fichier pour l'utilisateur :

```
vim /root/antoine.ldif

dn: cn=alemorvan,ou=users,dc=formatux,dc=lan
objectClass: top
objectClass: person
objectClass: posixAccount
objectClass: shadowAccount
cn: alemorvan
sn: Le Morvan
uid: alemorvan
```

```
uidNumber: 10000
gidNumber: 500
homeDirectory: /home/alemorvan
loginShell: /bin/bash
userPassword: {crypt}password
gecos: alemorvan
shadowWarning: 7
```

```
ldapadd -x -D cn=admin,dc=formatux,dc=lan -W -f /root/antoine.ldif
```

9

Installation d'un serveur Shinken

9.1. Généralités

Shinken est un logiciel de supervision créé en 2009 par Jean Gabes. Il est une réécriture de Nagios en Python et en reprend complètement l'esprit. Il va cependant permettre de répondre à des contraintes techniques auxquelles Nagios ne pouvait pas.

Shinken dépend essentiellement de Pyro, librairies Python. Son code est ouvert, libre et communautaire.

Son architecture est décentralisée et se base sur plusieurs processus. Elle prévoit également la haute disponibilité et de hautes performances.

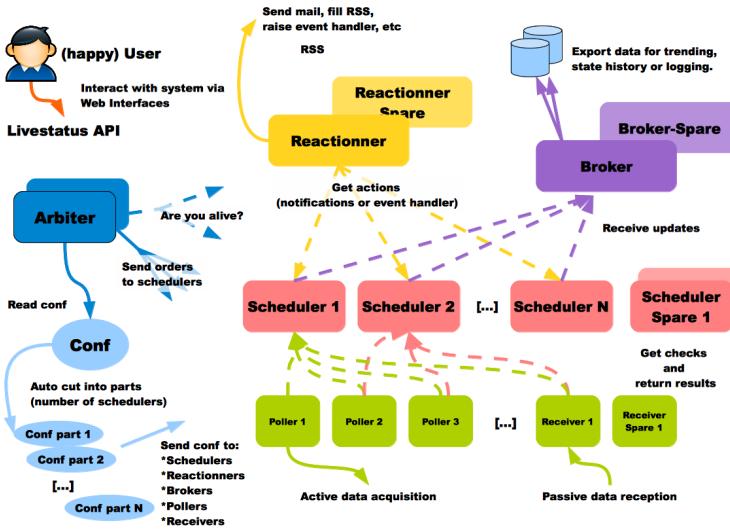


Figure 9.1. Architecture

9.2. Prérequis

L'installation est effectuée sur un serveur CentOS 7 minimal. Il est nécessaire de posséder un serveur correctement configuré (nom et adressage IP).

Ajouter l'utilisateur **shinken** qui sera utilisé pour le déploiement, lui configurer un mot de passe et l'ajouter au groupe **wheel** pour l'utilisation de sudo.

```
[root@srv-shinken ~]# useradd shinken
[root@srv-shinken ~]# passwd shinken
[root@srv-shinken ~]# usermod -aG wheel shinken
```

Vérifier ou activer si besoin %wheel avec la commande visudo et désactiver requiretty :

```
Defaults    !requiretty
%wheel     ALL=(ALL)      ALL
```

Se connecter ensuite avec l'utilisateur shinken pour poursuivre l'installation.

```
[root@localhost ~]# su - shinken
```

Configurer un premier dépôt epel :

```
[shinken@srv-shinken ~]$ sudo yum install epel-release
```

Puis un second dépôt pour mongodb :

```
[shinken@srv-shinken ~]$ sudo vim /etc/yum.repos.d/mongodb.repo
[mongodb]
name=MongoDB Repository
baseurl=http://downloads-distro.mongodb.org/repo/redhat/os/x86_64/
gpgcheck=0
enabled=1
```

9.3. Installation

9.3.1. Installation des composants nécessaires à Shinken

Installation des composants Python :

```
[shinken@srv-shinken ~]$ sudo yum install -y python-pip python-pycurl
python-setuptools git python-pymongo python-cherrypy
```

Installation d'une base de données nécessaire à WebUI qui sera utilisé pour l'affichage graphique. Ici, mongodb sera utilisé mais d'autres bases de données peuvent être utilisées.

```
[shinken@srv-shinken ~]$ sudo yum -y install mongodb-org mongodb-org-
server
```

Activation et démarrage :

```
[shinken@srv-shinken ~]$ sudo chkconfig mongod on
[shinken@srv-shinken ~]$ sudo systemctl start mongod
```

Revenir en tant que root pour installer pyro.

```
[shinken@srv-shinken ~]$ exit
```

```
[root@srv-shinken ~]#
```

Installation de Pyro :

```
[root@srv-shinken ~]# easy_install pyro
```

9.3.2. Installation de shinken

De nouveau avec l'utilisateur shinken, récupérer les sources sur le dépôt github :

```
[root@srv-shinken ~]# su - shinken
[shinken@srv-shinken ~]$ git clone https://github.com/naparuba/
shinken.git
```

Se déplacer dans le dossier shinken et lancer l'installation:

```
[shinken@srv-shinken ~]$ cd shinken
[shinken@srv-shinken shinken]$ sudo python setup.py install
```

Installation du package shinken. Ce package installe seulement un outil d'administration et d'installation de modules supplémentaires pour shinken.

```
[shinken@localhost shinken]$ sudo yum install -y shinken
```

Donner les droits nécessaires à l'utilisateur shinken :

```
[shinken@srv-shinken ~]$ sudo chown -R shinken:shinken /var/lib/shinken/
[shinken@srv-shinken ~]$ sudo chown -R shinken:shinken /var/log/shinken/
[shinken@srv-shinken ~]$ sudo chown -R shinken:shinken /var/run/shinken/
```

Activer tous les services shinken au démarrage :

```
[shinken@srv-shinken ~]$ sudo chkconfig shinken on
[shinken@srv-shinken ~]$ sudo chkconfig shinken-arbiter on
[shinken@srv-shinken ~]$ sudo chkconfig shinken-broker on
[shinken@srv-shinken ~]$ sudo chkconfig shinken-poller on
[shinken@srv-shinken ~]$ sudo chkconfig shinken-reactionner on
[shinken@srv-shinken ~]$ sudo chkconfig shinken-receiver on
```

```
[shinken@srv-shinken ~]$ sudo chkconfig shinken-scheduler on
```

Initialisation de shinken. Création d'un fichier **.shinken.ini** caché dans la home directory de l'utilisateur shinken.

```
[shinken@srv-shinken ~]$ shinken --init
```

9.3.3. Installation et configuration des modules

- Module permettant d'initialiser l'interface graphique **webui2**

```
[shinken@srv-shinken ~]$ shinken install webui2
Grabbing : webui2
OK webui2
```

Il faut alors modifier le fichier **/etc/shinken/brokers/broker-master.cfg** afin de définir le module **webui2**.

```
[shinken@srv-shinken ~]$ vim /etc/shinken/brokers/broker-master.cfg
...
modules webui2
...
```

Pour que **webui2** fonctionne, il faut également installer les dépendances suivantes avec **easy_install** en tant que root.

```
[shinken@srv-shinken ~]$ exit
[root@srv-shinken ~]#
[root@srv-shinken ~]# easy_install bottle
[root@srv-shinken ~]# easy_install pymongo
[root@srv-shinken ~]# easy_install requests
[root@srv-shinken ~]# easy_install arrow
[root@srv-shinken ~]# easy_install passlib
```

Se reconnecter ensuite avec l'utilisateur shinken.

```
[root@srv-shinken ~]# su - shinken
[shinken@srv-shinken ~]$
```

- Module utilisé comme moyen d'authentification.

```
[shinken@srv-shinken ~]$ shinken install auth-cfg-password
Grabbing : auth-cfg-password
OK auth-cfg-password
```

- Module utilisé pour la base de données mongodb.

```
[shinken@srv-shinken ~]$ shinken install mod-mongodb
Grabbing : mod-mongodb
OK mod-mongodb
```

Il faut ensuite modifier le fichier **/etc/shinken/modules/webui2.cfg** afin de définir les deux modules précédemment installés.

```
[shinken@srv-shinken ~]$ vim /etc/shinken/modules/webui2.cfg
...
modules auth-cfg-password, mongodb
...
```

Le module **auth-cfg-password** étant utilisé, il faut modifier le mot de passe dans le fichier **/etc/shinken/contacts.admin.cfg**. Ce sont les identifiants définis dans ce fichier (contact_name et password) qui seront utilisés pour s'authentifier.

```
[shinken@srv-shinken ~]$ vim /etc/shinken/contacts/admin.cfg
...
password *****
...
```

9.4. Démarrage de shinken

Afin d'accéder à l'URL de shinken <http://srv-shinken:7767>, il est nécessaire de configurer le pare-feu en créant une nouvelle règle et de redémarrer le service.

```
[shinken@srv-shinken ~]$ sudo vim /etc/sysconfig/iptables
...
-A INPUT -p tcp -m state --state NEW -m tcp --dport 7767 -j ACCEPT
...
```

```
[shinken@srv-shinken ~]$ sudo systemctl restart iptables
```

Démarrer ensuite le service shinken afin de prendre en compte la configuration.

```
[shinken@srv-shinken ~]$ sudo systemctl start shinken
```

L'URL donnée ici n'est bien sûr valable que si une résolution de noms permet de définir le nom du serveur (srv-shinken). Il est également possible d'utiliser l'adresse IP (<http://@IP:7767>).

Le port 7767 est le port de fonctionnement par défaut.

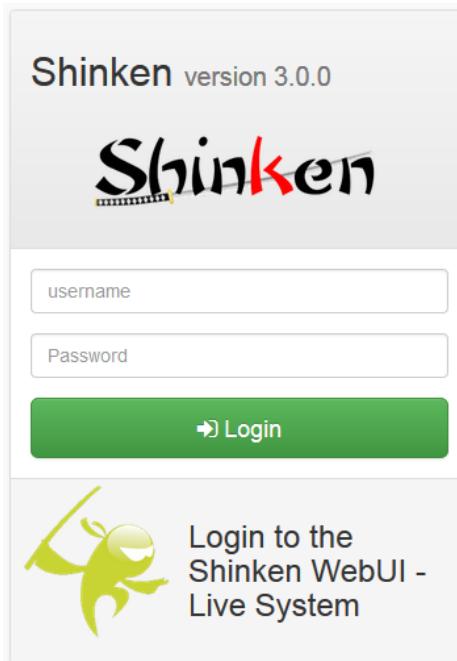


Figure 9.2. Fenêtre de connexion

9.5. Références

<https://shinken.readthedocs.io/en/latest/index.html>

<https://www.it-connect.fr/installer-shinken-3-0-sur-centos-7-en-10-etapes/>

10

Serveur proxy SQUID

10.1. Principes de fonctionnement

La mise en place d'un serveur proxy nécessite de choisir entre deux types d'architectures :

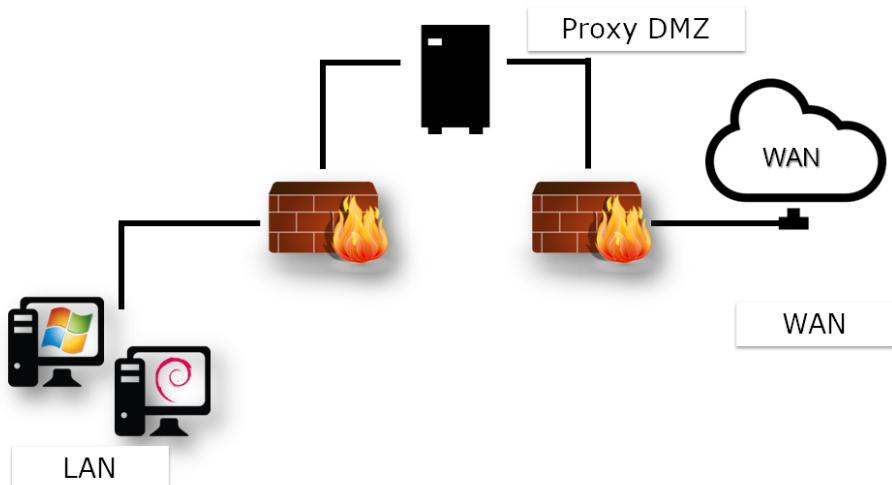
- Une architecture proxy standard, qui nécessitera une configuration spécifique de chaque client et de leurs navigateurs internet,
- Une architecture dite proxy captif, qui nécessitera l'interception des trames émises par le client et de les réécrire vers le serveur proxy.

Dans un cas comme dans l'autre, il y a rupture au niveau du réseau.

Un client ne peut physiquement plus s'adresser directement à un serveur distant, sans passer par un mandataire, appelé plus couramment serveur proxy.

Le poste client est protégé par deux pare-feux et ne communique jamais directement vers le réseau extérieur.

Architecture avec proxy

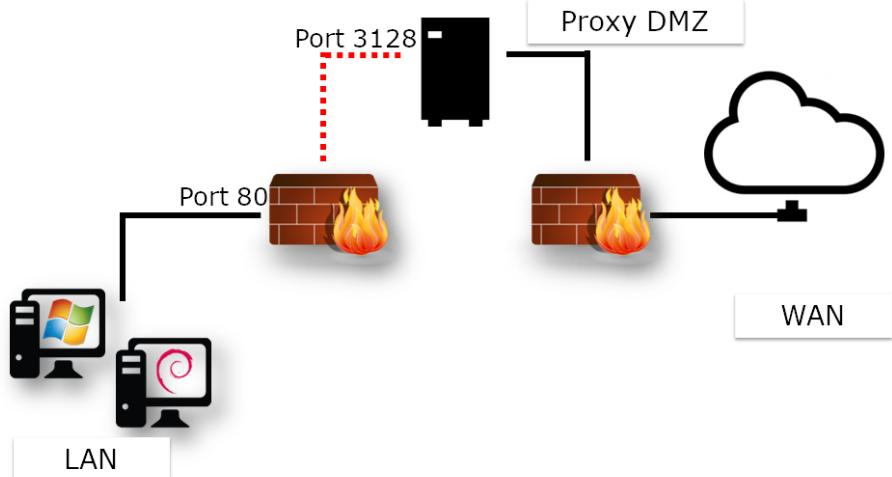


Cette architecture nécessite la configuration du navigateur sur le poste client.

Dans le cas du proxy captif, il n'y a pas de nécessité de configurer l'ensemble des postes clients.

La configuration se passe au niveau de la passerelle, qui reçoit les demandes des clients, et qui va, de manière totalement transparente, réécrire les trames pour les envoyer vers le proxy.

Architecture avec proxy captif

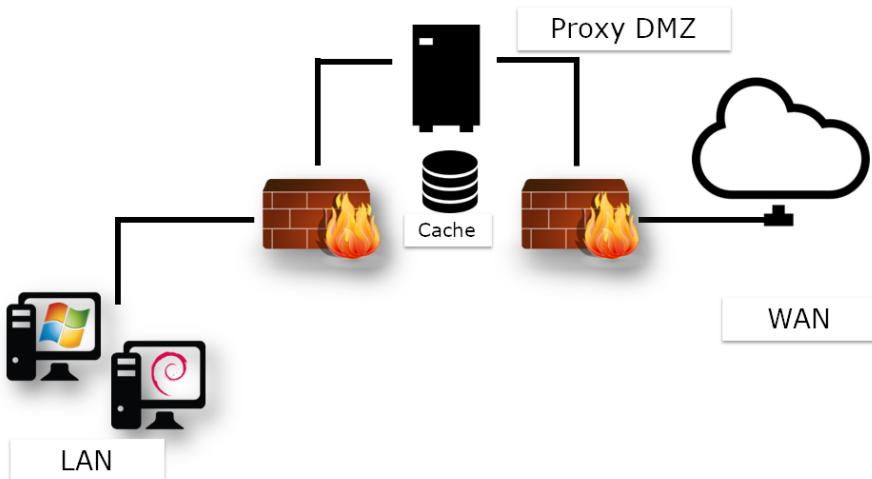


Cette architecture nécessite une configuration spécifique sur le routeur.

Dans le cas de l'architecture proxy standard ou du proxy captif, l'un des premiers intérêts de ce genre de service est bien évidemment de faire office de cache.

Ainsi, un fichier qui aura été téléchargé une première fois depuis le WAN (et donc potentiellement une liaison plus lente que le LAN), sera conservé en mémoire par le proxy-cache pour être resservi au profit des clients suivants. Ainsi, la bande passante de la liaison lente est optimisée.

Architecture avec proxy-cache



Comme nous le verrons dans la suite de ce chapitre, ce n'est bien évidemment pas la seule utilité d'un proxy.

Un proxy pourra être déployé pour :

- Interdire l'accès à certaines ressources en fonction de différents paramètres,
- Mettre en place une authentification et un suivi des activités sur internet des clients,
- Mettre en place une hiérarchie de cache distribués,
- Masquer l'architecture du LAN d'un point de vue WAN (combien y a-t-il de clients sur le LAN ?).

Les intérêts sont multiples :

- Anonymat sur Internet ;
- Authentification ;
- Journaliser les activités des clients ;
- Filtrage ;
- Limiter les accès ;
- Optimisation de la bande passante ;

- Sécurité.



Mettre en place l'authentification bloque une grande partie des effets malveillants des virus sur le LAN.



Le service proxy devient un service critique nécessitant une haute disponibilité.

Durant l'exploitation d'un serveur Proxy Squid, l'administrateur est amené à exploiter les logs. Il est donc primordiale de connaître les principaux codes réponses HTTP.

Tableau 10.1. Les codes réponses HTTP

Code	Catégories
1XX	Info
2XX	Succès
3XX	Redirection
4XX	Erreur de requête client
5XX	Erreur sur le serveur

Exemples :

- 200 : ok
- 301 : Moved Permanently
- 302 : Moved Temporatly
- 304 : Not modified
- 400 : Bad request
- 401 : Unauthorized
- 404 : Not found

10.2. Le serveur SQUID

Squid prend en charge les protocoles http et ftp.

Les intérêts d'installer une solution basée sur le serveur Squid :

- Les solutions matérielles sont coûteuses ;
- Il est développé depuis 1996 ;
- Il est publié sous licence GNU/GPL.

10.2.1. Dimensionnement

- Prévoir des solutions de haute disponibilité ;
- Privilégier des disques durs rapides pour le cache ;
- Mémoire vive et CPU correctement dimensionnés.



Il faut prévoir 14Mo de RAM par Go de cache sur le disque.

10.2.2. Installation

L'installation du serveur Squid se fait avec le paquet squid.

```
yum install squid  
chkconfig squid on
```



Attention à ne pas démarrer le service tant que le cache n'a pas été initialisé !

10.2.3. Arborescence et fichiers du serveur Squid

Le fichier de configuration unique est le fichier **/etc/squid/squid.conf**.

Les logs du service (arrêt et relance) sont enregistré dans le fichier **/var/log/squid.cache.log** tandis que les requêtes des clients **/var/log/squid/access.log**. Les fichiers de cache seront par défaut stockés dans **/var/spool/squid/**.

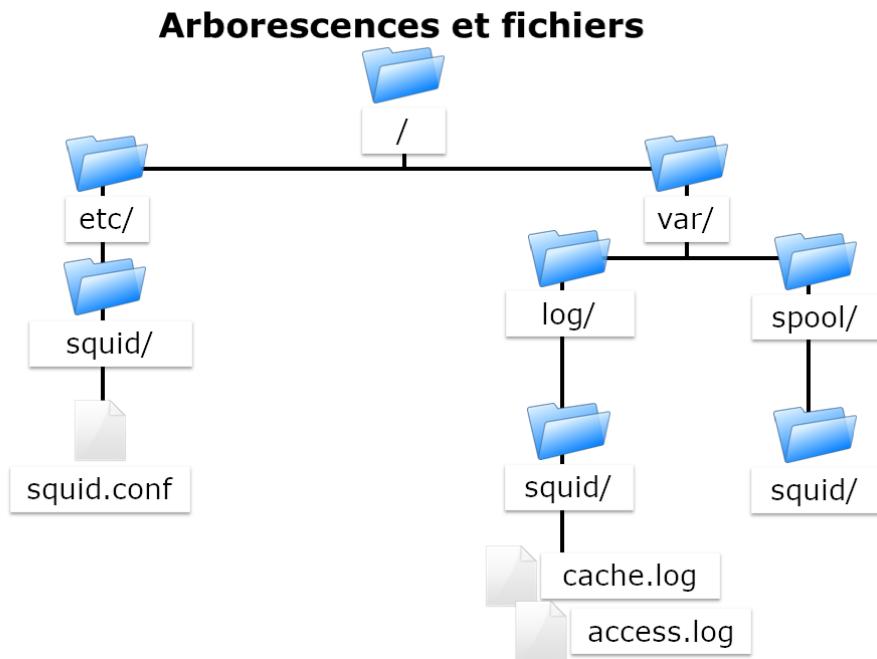


Figure 10.1. Arborescence et fichiers du serveur Squid

10.2.4. La commande squid

Commande squid permet de contrôler le serveur squid.

Syntaxe de la commande squid.

```
squid [-z|-s|-k parse|-k rotate]
```

Option	Description
-z	Initialise les répertoires du cache
-s	Active la journalisation syslog
-k parse	Test le fichier de configuration
-k rotate	Effectue une rotation des logs

Journaliser les requêtes clientes peut rapidement entraîner le stockage des volumes conséquents de données.

Il est opportun de régulièrement créer un nouveau fichier de log et d'archiver l'ancien dans un format compressé.

Cette action peut être effectuée manuellement avec l'option **-k rotate** de la commande squid ou via le service Linux dédié **Logrotate**.

10.3. Configuration basique

La configuration de Squid se fait dans le fichier de configuration **/etc/squid/squid.conf**.

- Numéro de port du proxy (port d'écoute)

Syntaxe de la directive **http_port**.

```
http_port num_port
```



Par défaut, le numéro de port est fixé à 3128 mais il est fréquemment changé à 8080. Il faudra penser à ouvrir le port correspondant du pare-feu !

Par exemple :

```
http_port 8080
```

Au redémarrage du service, le serveur Squid se mettra en écoute sur le port défini par la directive **http_port**.

- Réservation de la mémoire vive

Syntaxe de la directive **cache_mem**.

```
cache_mem taille KB|taille MB|taille GB
```

Par exemple :

```
cache_mem 1 GB
```



Bonne pratique : 1/3 du total de la mémoire vive allouée

- Protocole de Cache Internet (ICP)

Le protocole ICP (Internet Cache Protocol) permet aux serveurs Squid voisins de s'échanger des requêtes. Il est courant de proposer une hiérarchie de proxy qui se partagent leurs bases d'informations.

La directive `icp_port` permet de définir le numéro de port sur lequel Squid envoie et reçoit les requêtes ICP des serveurs Squid voisins.

Par exemple :

```
icp_port 3130
```



Positionner à 0 pour le désactiver.

- Utilisateur FTP anonyme

La directive `ftp_user` permet d'associer un utilisateur FTP aux connexions FTP anonymes. L'utilisateur dit être une adresse de messagerie valide.

```
ftp_user bob@formatux.lan
```

- Mettre en place des Access Control List

Syntaxe des ACL.

```
acl nom type argument  
http_access allow|deny nomacl
```

Exemple :

```
acl REPAS time 12:00-14:00  
http_access deny REPAS
```

Les ACL sont étudiées plus en détail dans la partie "Configuration avancée".

- Taille maximum d'un objet en cache

Syntaxe de la directive `maximum_object_size`.

```
maximum_object_size size
```

Exemple :

```
maximum_object_size 32 MB
```

Si la taille de l'objet est supérieur à la limite maximum_object_size, l'objet n'est pas conservé en cache.

- Nom du serveur proxy

Syntaxe de la directive visible_hostname.

```
visible_hostname nom
```

Exemple :

```
visible_hostname proxysquid
```



La valeur fournie peut être différente du nom d'hôte.

- Définir un cache pour squid

```
cache_ufs format chemin taille nbDossierNiv1 nbDossierNiv2
```

Plusieurs caches peuvent être définis sur différents systèmes de fichiers pour optimiser les temps d'accès.

Exemple :

```
cache_dir ufs /var/spool/squid/ 100 16 256
```

Option	Description
ufs	Unix File System
100	Taille en méga

Option	Description
16	16 dossiers de premier niveau
256	256 dossiers de second niveau

Au premier lancement du service, il faut initialiser le dossier de cache :

```
[root]# squid -z
[root]# service squid start
```

10.4. Configurations avancées

10.4.1. Les Access Control List (ACL)

Syntaxe de la directive `http_access`.

```
http_access allow|deny [!]nom_acl
```

Exemple :

```
http_access allow REPAS
http_access deny !REPAS
```



L'ACL `!nom_acl` est le contraire de l'ACL `nom_acl`.

Syntaxe de la directive `acl`.

```
acl nom type argument
```

L'ordre des ACL est cumulatif. Plusieurs ACL de même nom représentent une seule ACL.

Exemples :

- Autoriser à l'heure des repas :

```
acl REPAS time 12:00-14:00
```

```
http_access allow REPAS
```

- Interdire les vidéos :

```
acl VIDEOS rep_mime_type video/mpeg  
acl VIDEOS rep_mime_type video/avi  
http_access deny VIDEOS
```

- Gestion des adresses IP :

```
acl XXX src 192.168.0.0/255.255.255.0  
acl XXX dst 10.10.10.1
```

- Gestion des FQDN :

```
acl XXX srcdomain .formatux.lan  
acl XXX dstdomain .linux.org
```

- Gestion des ports :

```
acl XXX port 80 21
```

- Gestion des protocoles :

```
acl XXX proto HTTP FTP
```

10.4.2. Les algorithmes de cache

Il existe différents algorithmes de cache qui disposent de caractéristiques différentes :

- LRU - **Least Recently Used** : supprime les objets les plus anciens de la mémoire vive.
- LRU-THOLD : copie en fonction de sa taille un objet dans le cache.
- MRU : **Most Recently Used** : les données les moins demandées sont supprimées.

- GDSF : **Greedy Dual Size Frequency** : supprime en fonction de la taille et du temps d'accès d'origine. Les plus petits sont conservés.
- LFUDA : **Least Frequently Used With Dynamic Aging** : idem que GDSF sans notion de taille. Utile pour les caches avec des fichiers de grande taille.

10.5. Authentification des clients

Squid s'appuie sur des programmes externes pour gérer l'authentification. Il peut ainsi s'appuyer sur un simple fichier plat type htpasswd ou sur un service LDAP, SMB, PAM, etc.

L'authentification peut être une nécessité juridique : pensez à faire signer une charte d'usage à vos utilisateurs !

10.6. Outils

10.6.1. La commande squidclient

La commande squidclient permet de tester une requête vers le serveur squid.

Syntaxe de la commande squidclient.

```
squidclient [-s] [-h cible] [-p port] url
```

Exemple :

```
squidclient -s -h localhost -p 8080 http://localhost/
```

Option	Description
-s	Mode silencieux (n'affiche rien dans la console)
-h	Définir un proxy cible
-p	Port d'écoute (par défaut 3128)
-r	Forcer le serveur à recharger l'objet

10.6.2. Analyser les logs

Les enregistrements du journal de Squid peuvent être suivis avec la commande :

```
tail -f /var/log/squid/access.log
```

- Décomposition d'une ligne de log

Option	Description
Date	Horodatage du log
Tps reponse	Temps de réponse pour la requête
@ client	Adresse IP du client
Code status	Code HTTP de la réponse
Taille	Taille du transfert
Méthode	Méthode HTTP (Put / Get / Post / etc.)
URL	URL de la requête
Peer Code	Code de réponse inter-proxy
Type fichier	Type mime de la cible de la requête

10.6.3. La commande sarg

La commande sarg (**Squid Analysis Report Generator**) permet de générer un rapport au format HTML.

Syntaxe de la commande sarg.

```
sarg -x
```

- Installer sarg :

```
[root]# yum install httpd
[root]# yum install sarg
```

- Configurer sarg :

```
[root]# vim /etc/sarg/sarg.conf
access_log /var/log/squid/access.log
```

Le rapport est généré sous **/var/www/html/squid_reports/**.

10.6.4. SquidGuard

SquidGuard permet de filtrer les URLs à partir de blacklists (éventuellement disponibles sur Internet).

Sa mise en œuvre dépasse toutefois le cadre de ce support.

11

Serveur de log Syslog

Le système génère des logs qu'il faut surveiller pour la sécurité du système ou pour réagir avant la panne.

Sous Linux, c'est le rôle du protocole Syslog.

11.1. Généralités

Syslog est le protocole de journalisation standard sous Linux. Syslog gère le journal d'évènement Linux, que ce soit pour le noyau Linux ou pour les services hébergés sur la station.

- Les logs peuvent être archivés localement (dans ce cas il faut prévoir une rotation des logs).
- Syslog peut également fonctionner en local en mode client/serveur. Syslog utilise le port 514 en UDP ou TCP pour sa communication réseau.

Exemple de fichier /var/log/messages :

```
Nov 23 08:30:00 centos6 dhcp service[warning] 110 message
```

Sous CentOS 6, c'est le logiciel rsyslog qui est utilisé pour gérer les logs du système. A noter que la syntaxe rsyslog est compatible avec les clients syslog standard (syslog, syslog-ng), mais l'inverse n'est pas vrai.

Un journal au format syslog comporte dans l'ordre les informations suivantes :

- la date à laquelle a été émis le log,

- le nom de l'équipement ayant généré le log (hostname),
- une information sur le processus qui a déclenché cette émission,
- le niveau de priorité du log,
- un identifiant du processus ayant généré le log
- le corps de message.

Certaines de ces informations sont optionnelles.

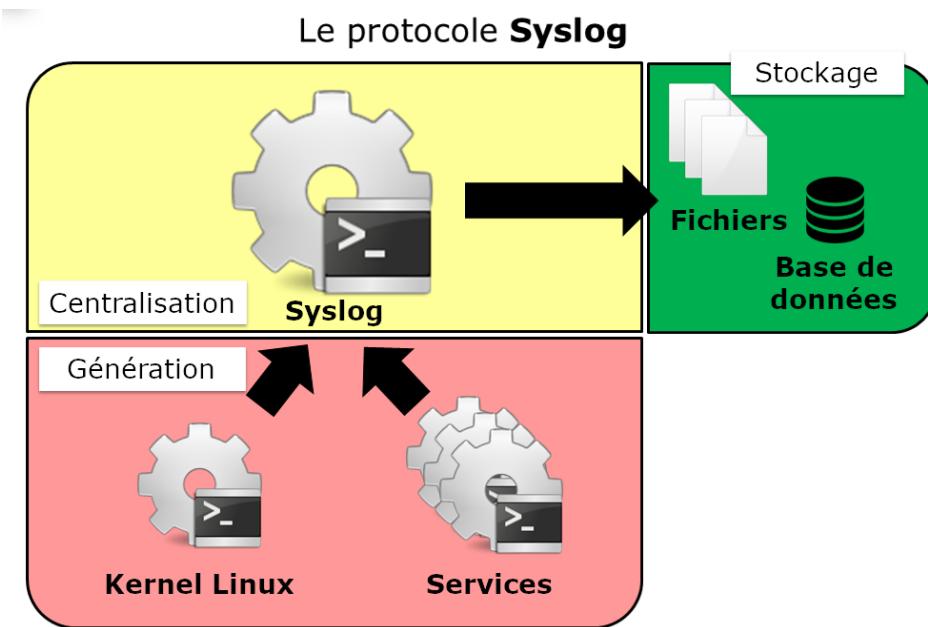


Figure 11.1. Fonctionnement du protocole Syslog

Le serveur Syslog centralise les messages du kernel Linux ou des services dans des fichiers. Des modules existent pour rediriger les logs vers une base de données.

En mode client/serveur, les clients envoient leurs logs vers un serveur syslog sur le port 514. Ce serveur peut ensuite stocker les logs de ses clients vers un serveur de base de données.

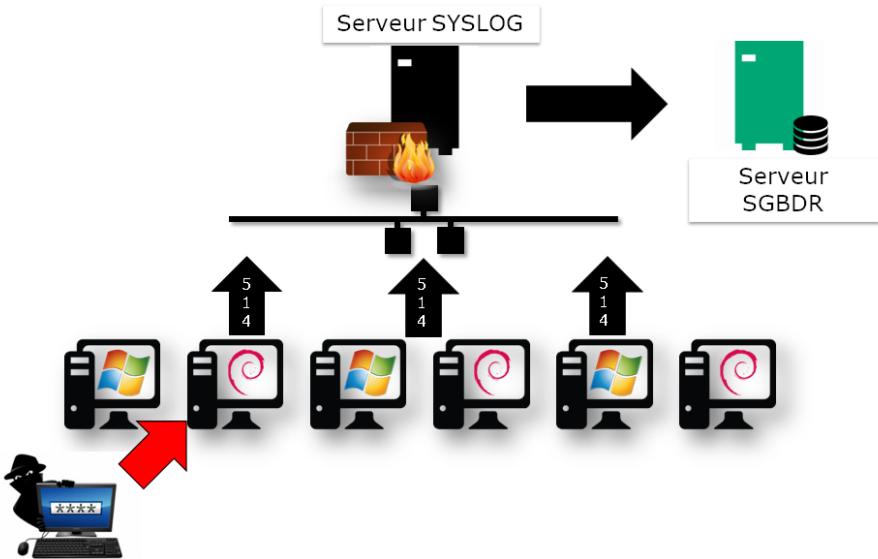


Figure 11.2. Mise en réseau du protocole Syslog

Ainsi, un attaquant ne peut pas effacer ces traces qui sont déportées sur un serveur distant.

11.1.1. Les catégories de messages

Les messages sont orientés selon leur origine et leur gravité (origine.gravité).

Tableau 11.1. Origine des messages Syslog

Code	Mot-clé	Description
0	kern	kernel messages
1	user	user-level messages
2	mail	mail system
3	daemon	system daemons
4	auth	security/authorization messages
5	syslog	messages generated internally by syslogd
6	lpr	line printer subsystem
7	news	network news subsystem
8	uucp	UUCP subsystem

Code	Mot-clé	Description
9		clock daemon
10	authpriv	security/authorization messages
11	ftp	FTP daemon
12	-	NTP subsystem
13	-	log audit
14	-	log alert
15	cron	clock daemon
16	local0	local use 0 (local0)
17	local1	local use 1 (local1)
18	local2	local use 2 (local2)
19	local3	local use 3 (local3)
20	local4	local use 4 (local4)
21	local5	local use 5 (local5)
22	local6	local use 6 (local6)
23	local7	local use 7 (local7)

Tableau 11.2. Gravité des messages syslog

Code	Gravité	Mot-clé	Description
0	Emergency	emerg (panic)	Système inutilisable.
1	Alert	alert	Une intervention immédiate est nécessaire.
2	Critical	crit	Erreur critique pour le système.
3	Error	err (error)	Erreur de fonctionnement.
4	Warning	warn (warning)	Avertissement (une erreur peut intervenir si aucune action n'est prise).
5	Notice	notice	Événement normal méritant d'être signalé.

Code	Gravité	Mot-clé	Description
6	Informational	info	Pour information.
7	Debugging	debug	Message de mise au point.

11.2. Client Syslog

La configuration du client et du serveur rsyslog est centralisée dans le fichier **/etc/rsyslog.conf**.

Après toute modification il faut redémarrer le service :

```
service rsyslog restart
```

La commande logger génère une ligne de log.

Syntaxe de la commande logger.

```
logger texte
```

Exemple :

```
logger "====> Marqueur"
```

Fichier **/var/log/messages** après exécution de la commande logger.

```
Nov 23 08:30:00 centos6 stagiaire =====> Marqueur
```

Il est possible de rediriger les logs du client vers le serveur :

Modification du fichier **/etc/rsyslog.conf** pour envoyer les logs vers le réseau.

```
*.* @IPServer:514
```

```
service rsyslog restart
logger test
```

Le message test est envoyé vers le serveur.



@IPServeur = UDP @IPServeur = TCP

Pour différencier une redirection en TCP d'une redirection en UDP, il faudra doubler l'arobase présent devant l'adresse IP du serveur.

Par exemple :

```
mail.err* @@172.16.96.203
```

Après avoir ajouté cette ligne, le service syslog enverra les logs de la catégorie mail d'un niveau de gravité supérieur à erreur vers le serveur syslog 172.16.96.203 en TCP.

11.2.1. La commande logwatch

La commande logwatch effectue une synthèse journalière des logs et l'envoie par message.

Installation :

```
yum install logwatch
```

LogWatch analyse pour vous quotidiennement les logs pour en extraire les informations du jour, les trie et vous envoie une synthèse quotidienne.

Les logs des services étant généralement très copieux, un outil tel Logwatch (couplé avec la redirection des mails) est nécessaire pour rester informé en un seul coup d'œil.

Voici un exemple de rapport :

```
##### Logwatch 7.3.6 (05/19/07) #####
Processing Initiated: Fri Oct 23 10:10:04 2015
Date Range Processed: yesterday
( 2015-Oct-22 )
Period is day.
Detail Level of Output: 0
Type of Output: unformatted
Logfiles for Host: srv-instructeurs.formatux.lan
```

```
#####
----- Selinux Audit Begin -----
----- Selinux Audit End -----
----- Automount Begin -----
----- Automount End -----
----- Cron Begin -----
----- Cron End -----
----- httpd Begin -----
Requests with error response codes
 403 Forbidden
    /: 1 Time(s)
 404 Not Found
    /favicon.ico: 2 Time(s)
----- httpd End -----
----- Init Begin -----
----- Init End -----
----- Named Begin -----
Received control channel commands
  reload: 8 Time(s)
  stop: 7 Time(s)
----- Named End -----
```

```
----- pam_unix Begin -----
```

su-1:

 Authentication Failures:
 Sessions Opened:
 pupitre -> root: 1 Time(s)

sudo:

 Authentication Failures:

```
----- pam_unix End -----
```

```
----- Postfix Begin -----
```

3.957K	Bytes accepted	4,052
3.957K	Bytes delivered	4,052
=====	=====	=====

4	Accepted	100.00%
-----	-----	-----
4	Total	100.00%
=====	=====	=====

4	Removed from queue
2	Sent via SMTP
2	Forwarded
-----	-----
6	Postfix start
6	Postfix stop
1	Postfix waiting to terminate

```
----- Postfix End -----
```

```
----- Connections (secure-log) Begin
```

New Users:
 postgres (26)

New Groups:

```
postgres (26)
```

```
groupadd: group added to /etc/group: name=postgres, GID=26: 1 Time(s)
groupadd: group added to /etc/gshadow: name=postgres: 1 Time(s)
webmin: Successful login as pupitre from 172.16.96.232: 1 Time(s)
```

```
----- Connections (secure-log) End -----
```

```
----- SSHD Begin -----
```

```
----- SSHD End -----
```

```
----- Sudo (secure-log) Begin -----
```

```
----- Sudo (secure-log) End -----
```

```
----- yum Begin -----
```

Packages Installed:

```
postgresql-libs-8.4.20-3.el6_6.x86_64
postgresql-server-8.4.20-3.el6_6.x86_64
postgresql-8.4.20-3.el6_6.x86_64
1:mod_ssl-2.2.15-47.el6.centos.x86_64
1:net-snmp-libs-5.5-54.el6_7.1.x86_64
policycoreutils-python-2.0.83-24.el6.x86_64
```

```
----- yum End -----
```

```
----- Disk Space Begin -----
```

Filesystem	Size	Used	Avail	Use%	Mounted on
/dev/mapper/vg_root-lv_root	27G	3.7G	22G	15%	/
/dev/sda1	485M	34M	426M	8%	/boot
/dev/sdb1	488M	154M	310M	34%	/BoiteAOutils

```
----- Disk Space End -----
```

```
##### Logwatch End #####
```

11.3. Serveur Syslog

L'architecture de rsyslog est modulaire. Pour activer son mode serveur, il faut charger le module UDP ou TCP et le mettre en écoute sur le port 514 sans oublier de relancer le service.

Activer les serveurs UDP et TCP rsyslog.

```
vim /etc/rsyslog.conf
$ModLoad imudp
$UDPServerRun 514

$ModLoad imtcp
$InputTCPServerRun 514
```

```
service rsyslog restart

netstat -tagn | grep 514
udp  0  0  0.0.0.0:514  0.0.0.0:*    LISTEN  3172/rsyslog
```

11.3.1. Stocker les logs dans des fichiers différenciés

Les modifications à apporter au fichier /etc/rsyslog.conf sont les suivantes :

```
## Rules
$template syslog, "/var/log/%fromhost%.log"

mail.* ?syslog
```

Explications :

- **\$template** : définir un template qui s'appellera **syslog**
- la variable **%fromhost%** contient le nom du client à l'origine du message

- **mail.*** correspond à tout les messages d'origine mail qui seront redirigés vers le template que nous avons appelé syslog (?syslog)

11.4. Stockage en base de données

Il est particulièrement intéressant de stocker les enregistrements syslog en base de données. Il est ensuite possible de visualiser les logs dans des interfaces web spécialisées (ici LogAnalyzer) :

Date	Entity	Severity	Host	Syslogtag	ProcessID	Message-type
2013-09-04 02:00:01	SECURITY	INFO	logalyzer-demo	CRON	8362	Syslog [pm_anti(cronsession)] session opened for user test by (id ...)
2013-09-04 02:24:48		SYSLOG	INFO	logalyzer-demo	8326	Syslog [pm_anti(cronsession)] session closed for user root
2013-09-04 02:27:01	SECURITY	INFO	logalyzer-demo	CRON	8327	Syslog [root] CRON (0 / & /run-parts -- report /etc/cron.hourly)
2013-09-04 02:27:01	SECURITY	INFO	logalyzer-demo	CRON	8326	Syslog [pm_anti(cronsession)] session opened for user root by (id ...)
2013-09-04 02:27:01	SECURITY	INFO	logalyzer-demo	CRON	8327	Syslog [root] CRON (0 / & /run-parts -- report /etc/cron.hourly)
2013-09-04 02:28:01	SECURITY	INFO	logalyzer-demo	CRON	8293	Syslog [pm_anti(cronsession)] session opened for user root by (id ...)
2013-09-04 02:28:01	SECURITY	INFO	logalyzer-demo	CRON	8294	Syslog [root] CRON (0 / & /run-parts -- report /etc/cron.hourly)
2013-09-04 02:28:01	SECURITY	INFO	logalyzer-demo	CRON	8293	Syslog [pm_anti(cronsession)] session opened for user test by (id ...)
2013-09-04 02:28:01	SECURITY	INFO	logalyzer-demo	CRON	8293	Syslog [pm_anti(cronsession)] session closed for user root
2013-09-04 02:28:01	SECURITY	INFO	logalyzer-demo	CRON	8182	Syslog [root] CRON (0 / & /run-parts -- report /etc/cron.hourly)
2013-09-04 02:28:01	SECURITY	INFO	logalyzer-demo	CRON	8181	Syslog [pm_anti(cronsession)] session opened for user root by (id ...)
2013-09-04 02:28:01	SECURITY	INFO	logalyzer-demo	CRON	8140	Syslog [pm_anti(cronsession)] session closed for user test
2013-09-04 02:28:01	SECURITY	INFO	logalyzer-demo	CRON	8140	Syslog [root] CRON (0 / & /run-parts -- report /etc/cron.hourly)
2013-09-04 02:28:01	SECURITY	INFO	logalyzer-demo	CRON	8140	Syslog [pm_anti(cronsession)] session opened for user test by (id ...)
2013-09-04 02:28:01	SECURITY	INFO	logalyzer-demo	CRON	8037	Syslog [pm_anti(cronsession)] session closed for user root
2013-09-04 02:28:01	SECURITY	INFO	logalyzer-demo	CRON	8037	Syslog [root] CRON (0 / & /run-parts -- report /etc/cron.hourly)
2013-09-04 02:28:01	SECURITY	INFO	logalyzer-demo	CRON	8037	Syslog [pm_anti(cronsession)] session opened for user root by (id ...)
2013-09-04 02:28:01	SECURITY	INFO	logalyzer-demo	CRON	7995	Syslog [root] CRON (0 / & /run-parts -- report /etc/cron.hourly)
2013-09-04 02:28:01	SECURITY	INFO	logalyzer-demo	CRON	7995	Syslog [pm_anti(cronsession)] session opened for user root by (id ...)
2013-09-04 02:28:01	SECURITY	INFO	logalyzer-demo	CRON	7992	Syslog [root] CRON (0 / & /run-parts -- report /etc/cron.hourly)
2013-09-04 02:28:01	SECURITY	INFO	logalyzer-demo	CRON	7991	Syslog [pm_anti(cronsession)] session opened for user root by (id ...)
2013-09-04 02:28:01	SECURITY	INFO	logalyzer-demo	CRON	7948	Syslog [pm_anti(cronsession)] session closed for user test
2013-09-04 02:30:01	SECURITY	INFO	logalyzer-demo	CRON	7948	Syslog [pm_anti(cronsession)] session closed for user root

Figure 11.3. Interface du logiciel LogAnalyzer

Installer le module Mysql :

```
yum install rsyslog-mysql
```

Configurer le module dans /etc/rsyslog.conf :

```
$ModLoad MySQL
*. * > @IPServer,base,USERMYSQL,PWD MYSQL
```



La création de la base Mysql sort du cadre de ce support.

```
service rsyslog restart
```

Glossaire

BASH

Bourne Again SHELL

BIOS

Basic Input Output System

CIDR

Classless Inter-Domain Routing

Daemon

Disk And Execution MONitor

DHCP

Dynamic Host Control Protocol

DNS

Domain Name Service

FQDN

Fully Qualified Domain Name

LAN

Local Area Network

NTP

Network Time Protocol

nsswitch

Name Service Switch

POSIX

Portable Operating System Interface

POST

Power On Self Test

SHELL

En français "coquille". À traduire par "interface système".

SMB

Server Message Block

SMTP

Simple Mail Tranfer Protocol

SSH

Secure SHell

TLS

Transport Layer Security, un protocole de cryptographie pour sécuriser les communications IP.

TTY

teletypewriter, qui se traduit téléscripteur. C'est la console physique.

UEFI

Unified Extensible Firmware Interface

Index

A

AAAA, 16
active, 126
Apache,
apachectl, 61

B

BIND, 8
bounce, 126

C

CA, 149
cacertdir_rehash, 148
ccze, 108
certtool, 146
CIFS, 26
cleanup, 108, 124

D

deferred, 126
dig, 22
DIT, 132
DN, 132
DNS, 7
Dovecot, 119

E

ESMTP, 98

F

FQDN, 7

G

gnutls-utils, 146

H

HTTP, 43

I
IMAP, 98
incoming, 126

K

KeepAlive, 56

L

LDA, 101
LDAP, 131
ldapadd, 136
ldapdelete, 136
ldapmodify, 136, 141
ldappasswd, 136
ldapsearch, 136
LDIF, 135
LMTP, 98
local, 109, 125

M

MAA, 101
Maildir, 103
maildrop, 111, 126
mailx, 109
mbox, 103
MDA, 101
mod_proxy, 91
mod_proxy_balancer, 94
MTA, 97
MUA, 100
MX, 16, 100

N

NetBIOS, 26
NetworkManager, 21
NFS, 1
NSCD, 23
NSS, 132
nsupdate, 19

O

OLC, 133
OpenLDAP, 131
OU, 144

P

pickup, 108, 111, 123
pipe, 126
POP, 99
postconf, 113
Postfix,
postmap, 119

Q

qmgr, 109, 125

R

rdnc, 9
relayhost, 115
rndc, 19
RootDN, 139
RootPW, 139
RPC, 1
RR, 15, 16

S

Samba, 25
SASL, 133

SELinux, 53
Sendmail, 97
Sieve, 99
slapcat, 141
SMB, 26
SMTP, 97
smtp, 125
smtpd, 108, 124

SOA, 17
startTLS, 132
starttls, 146
swaks, 113

T

TLD, 8
TLS, 146
trivial-rewrite, 109, 124
TTL, 10

U

URL, 44
UUCP, 101