Les Serveurs sous Linux

Toumanari – le 16, 17 et 18 décembre 2010

Plan

- Architecture client/Serveur les Sockets
- TPC Wrappers et Super démon
- Installation des Serveurs
- Configuration d'Apache
- Configuration de DNS

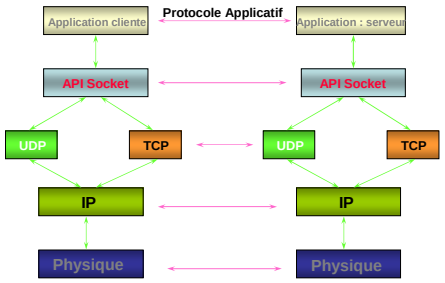
Les sockets

Toumanari – le 16, 17 et 18 décembre 2010

Les sockets

- Les sockets : interface client/serveur utilisée à l'origine dans le monde UNIX et TCP/IP.
- Étendue aujourd'hui du micro (Cf Winsock) au Mainframe.
- fournit les primitives pour le support des communications reposant sur toute suite de protocoles; les protocoles TCP/IP sont à l'origine des développements.
- Les applications cliente et serveur ne voient les couches de communication qu'à travers l'API socket (abstraction):

Les sockets



Sockets : l'abstraction

- associe un descripteur à un socket;
- le concepteur d'application utilise ce descripteur pour référencer la communication client/serveur sous-jacente.
- une structure de données «socket» est créée à l' ouverture de socket;

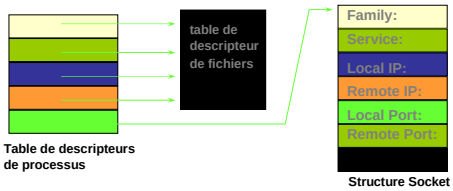
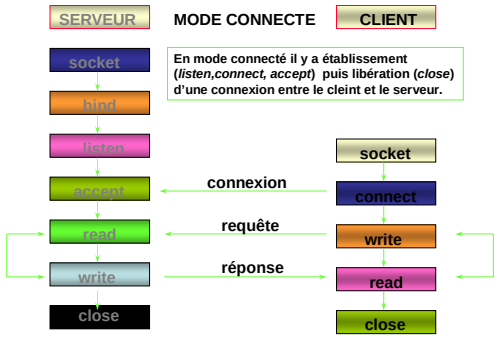


Table de descripteurs de processus

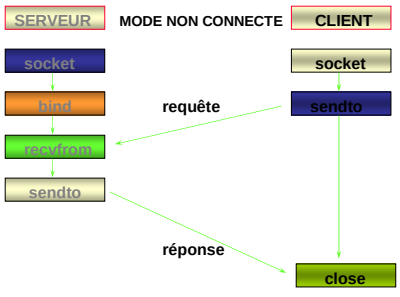
Structure Socket

La primitive socket permet l'ouverture de cette socket; initialement, après l'appel à cette fonction, la structure de données associée au socket est principalement vide, les appels à d'autres primitives de l'interface socket renseigneront ces champs vides.

Les Sockets : Mode connecté



Les Sockets : Mode non connecté



Socket : exemple de serveur itératif

```
int sockfd, newsockfd ;

if ( ( sockfd = socket (.....)) < 0 )      err_sys(«erreur de socket«) ;
if ( bind ( sockfd, ....) < 0 )            err_sys («erreur de bind»)
if ( listen ( sockfd , 5) < 0 ) ;                  err_sys (« erreur de listen» ) ;

for ( ; ; ) {
    newsockfd = accept ( sockfd, .....) ;
    if ( newsockfd < 0)
          err_sys( «erreur de accept») ;

    execute_la_demande( newsockfd ) ;
    close ( newsockfd ) ;
}
```


Socket : exemple de serveur parallèle

```
int sockfd, newsockfd ;

if ( ( sockfd = socket (.....)) < 0 )      err_sys(«erreur de socket«) ;
if ( bind ( sockfd, ....) < 0 )  err_sys («erreur de bind»)
if ( listen ( sockfd , 5) < 0 ) ;          err_sys (« erreur de listen» ) ;

for ( ; ; ) {
    newsockfd = accept ( sockfd, .....) ;
    if ( newsockfd < 0)       err_sys( «erreur de accept») ;
    if ( fork() == 0 ) {
          close ( sockfd ) ;
          execute_la_demande( newsockfd ) ;
          exit (1) ;
          }
    close ( newsockfd ) ;
}
```


Sockets : gestion de noms

- Les primitives gethostname et sethostname
  - gethostname permet aux processus utilisateurs d'accéder au nom de la machine locale.
  - sethostname permet à des processus privilégiés de définir le nom de la machine locale.
- La primitive getsockname
  - Cette primitive rend le nom associé au socket qui est spécifié en paramètre.

Sockets : gestion de noms

- Lorsque ces fonctions sont exécutées sur des machines ayant accès à un serveur de noms de domaines, elles fonctionnent elles-mêmes en mode client/serveur en émettant une requête vers le serveur de nom de domaines et attendent la réponse.
- Lorsqu'elles sont utilisées sur des machines qui n'ont pas accès à un serveur de noms, elles obtiennent les informations à partir d'une base de données (simple fichier) locale.

- gethostbyname spécifie un nom de domaine et retourne un pointeur vers une structure hostent qui contient les informations propres à ce nom de domaine.

- gethostbyaddr permet d'obtenir les mêmes informations à partir de l'adresse spécifiée.

Sockets : fonctions de service

- Les fonctions getprotobyname et getprotobynumber
  - Dans la base de données des protocoles disponibles sur la machine, chaque protocole a un nom officiel, des alias officiels et un numéro de protocole officiel.
  - La fonction *getprotobyname* permet d'obtenir des informations sur un protocole donné en spécifiant son nom; renseigne la structure *protoent*.
  - La fonction *getprotobynumber* permet d'obtenir les mêmes informations en spécifiant le numéro de protocole.
- La fonction getservbyname
  - Certains numéros de ports sont réservés pour les services s'exécutant au-dessus des protocoles TCP et UDP.
  - *getservbyname* retourne les informations relatives à un service donné en spécifiant le numéro du port et le protocole utilisé; renseigne la structure *servent.*

Sockets : Byte ordering

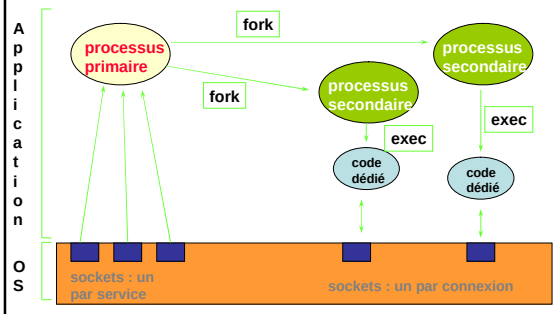
Pour que les applications fonctionnent correctement, elles doivent translater la représentation des données de la machine locale vers le network byte order :

- htonl : host to network long : convertit une valeur sur 32 bits de la représentation machine vers la représentation réseau.
- htons : host to network short : convertit une valeur sur 16 bits de la représentation machine vers la représentation réseau.
- ntohl : network to host long : convertit une valeur sur 32 bits de la représentation réseau vers la représentation machine.
- ntohs : network to host short : convertit une valeur sur 16 bits de la représentation réseau vers la représentation machine.

Sockets : les options

- Une application peut contrôler certains aspects du fonctionnement des sockets:
  - configurer les valeurs des temporisations,
  - l'allocation de la mémoire tampon,
  - vérifier si le socket autorise la diffusion ou la gestion des données hors bande.
- La primitive getsockopt
- Permet à une application d'obtenir les informations relatives au socket. Le système d'exploitation exploite les structures de données internes relatives au socket et renseigne l'application appelante.

Sockets : serveurs multi-services



TCP Wrappers et super-server

Toumanari – le 16, 17 et 18 décembre 2010

Lancement des services réseaux

- Le système offre un contrôle d'accès sécurisé fiable grâce à la mise en oeuvre du firewall IpTables
- Cependant, pour accroître la sécurité, il est recommandé d'ajouter une couche de protection supplémentaire individuelle à chaque service réseau démarré
- Cette protection est offerte sur deux niveaux :
  - Les TCP Wrappers (enveloppeurs réseaux) qui détermine les machines autorisées à se connecter à chaque service
  - Xinetd qui s'intercale entre les TCP Wrappers et le service réseaux et offre un contrôle d'accès plus affiné au service réseau

Les enveloppeurs TCP

- Lorsqu'une tentative de connexion à un service est effectuée
  - L'enveloppeur TCP contrôle l'accès en fonction des fichiers /etc/hosts.allow et /etc/hosts.deny
  - Il enregistre ensuite les informations de connexion dans le fichier de logs /var/log/secure ou /var/log/messages
- Si l'accès est donné, l'enveloppeur TCP n'interfère plus dans le processus de communication entre le serveur et le client
- Les enveloppeurs TCP sont ainsi complètement transparents dans le contrôle d'accès à un serveur

Xinetd : un super-server

- xinetd est un super-service enveloppé dans un enveloppeur TCP contrôlant l'accès à un sous-réseau de services réseaux comme ftpd, telnetd, etc.
- En ce sens, xinetd est un super-service car il centralise l'accès à d'autres services réseaux et permet de les contrôler plus finement
- xinetd permet le contrôle d'accès, la redirection réseau, la gestion des ressources et l'enregistrement de connexion (logging)
- C'est un service très puissant. Cependant, beaucoup de services décident de contrôler eux-même l'accès et se passent de xinetd

Fonctionnement de xinetd

- Xinetd écoute sur certains ports réseaux associés à des services
- Lors d'une tentative de connexion à un service réseau géré par xinetd
  - Une première vérification d'accès est faite par l'enveloppeur TCP
  - Si l'accès est autorisé, xinetd vérifie l'accès et les modalités de démarrage du service en fonction de sa propre configuration pour ce service
  - Si l'accès est autorisé par xinetd, une instance du service est démarrée à qui la connexion est cédée. xinetd n'intervient plus alors dans le processus de communication entre le serveur et le client
  - Xinetd gère les nouvelles tentatives de connexion à ce service, et en fonction des ressources allouées, décide s'il doit lancer une nouvelle instance de ce service

Xinetd : fichiers de configuration

- /etc/xinetd.conf : configuration globale de xinetd
- Le répertoire /etc/xinetd.d/ : fichiers de configuration spécifiques au service
  - includedir /etc/xinetd.d (dans xinetd.conf)
  - La plupart des directives de configuration globale sont héritées aux services

/etc/xinetd.conf : configuration globale

- Paramètres généraux lus une seule fois au démarrage de xinetd
  - Lors de changement dans la configuration, nécessaire de redémarrer xinetd
- Exemple :

```
defaults
        {
        // nbre  req max géré xinetd à 1 moment donnée
        instances  = 60
 // log envoyer à syslog via fichier /var/log/xinedlog (authpriv facilite
envoi)
        log_type  = SYSLOG authpriv
              log_on_success  = HOST PID
              log_on_failure  = HOST
//25 conn/s à 1 service si atteint blocage 30s
      cps  = 25 30                    }
includedir /etc/xinetd.d
```


/etc/xinetd.conf : configuration globale

- Instances : nombre de requêtes maximum que xinetd peut gérer
- log_type : les logs sont envoyées à syslogd avec la facilité authpriv. Pour enregistrer directement dans un fichier sans envoyer à syslogd, FILE /var/log/xinetdlog
- log_on_success : engregistrer les connexions réussies. Par defaut, l'adresse IP de la machine et le process ID du server lancé sont enregistrés
- log_on_failure : entregistrer les connexions non-réussies ou non-autorisées
- Cps : nombre de connexion / seconde pour chaque service. Si cette limite est atteinte, le service est inacessible pendant 30s.
- includedir /etc/xinetd.d/ : inclus les options de configurations pour chaque service, sous la forme d'un fichier spécifique

Le répertoire /etc/xinetd.d

- Ce répertoire contient les fichiers de configuration spécifiques à chaque service. Comme xinetd.conf, lus au démarrage de xinetd une fois pour toute.
- Exemple :

```
service telnet
    {
        flags           = REUSE
        socket_type     = stream
        wait            = no
        user            = deamon
        server          = /usr/sbin/in.telnetd
        log_on_failure  += USERID
        disable         = yes
    }
```


Le répertoire /etc/xinetd.d

- Service : nom du service. Correspond en général aux services définis dans /etc/services.
- Flags : définis les attributs de la connexion. REUSE ordonne à xinetd de réutiliser le socket pour une connexion Telnet
- socket_type : définit le type de socket à stream
- Wait : le service est simple-tâche (yes) our multi-tâches (no) ?
- User : sous quel utilisateur le service doit être lancé
- Server : définit le programme à lancer
- log_on_failure : paramètres à enregistrer en cas de connexion réussie, en plus des paramètres par défaut définis dans xinetd.conf
- Disable : définit si le service est active ou pas

Installation des Serveurs

Toumanari – le 16, 17 et 18 décembre 2010

Installation

- - Fichier Binaire
- - Package RPM
- - A partir des sources
  - - décompression et désarchivage
  - - compilation des sources (Structure de Makefile)
  - Démarrage pour les différentes niveaux d'exécutions

Configuration serveur Apache

- **ServerType** standalone // Le serveur s'exécutera seul, sans recourir au super-serveur *xinetd*.
- **ServerRoot** /etc/httpd // Il s'agit du répertoire où le serveur trouvera son répertoire de configuration conf
  On trouve dans */etc/httpd*, un lien vers */var/log/httpd/access_log*, le fichier-journal des accès aux ressource.
- **PidFile** /var/run/httpd.pid // C'est le fichier où le serveur en exécution stocke son premier numéro de processus (PID)
- **DocumentRoot** /var/www/html // fixe la racine du serveur Web, c'est-à-dire le répertoire de base où sont cherchées par défaut les pages html, lorsque l'URL ne comporte pas de chemin de répertoire
- **Port** 80 // Apache écoute sur le port tcp usuel
- **User** apache (dans commonhttpd.conf)
  Group apache // Apache doit être démarré par root, mais par sécurité ses processus auront pour propriétaire l'utilisateur *apache*, sans privilège.
- **ServerAdmin** root@localhost (dans commonhttpd.conf) // S'il a un problème, le serveur écrit un message à cette adresse

Configuration serveur Apache

- **UserDir public_html** // Ce paramètre signifie que l'utilisateur toto peut publier ses pages WEB personnelles dans un sous-répertoire de son répertoire perso, qui doit être nommé public_html, c'est-à-dire dans /home/toto/ public_html. Sa page d'accueil sera alors accessible par l'URL : http://serveur/~toto , où serveur est le nom du serveur ou son adresse IP.
- **DirectoryIndex index.html index.php index.htm ...** // Il est courant d'omettre le nom du fichier de la page d'accueil d'un site ou de l'un de ses sous-répertoires. Pour ne pas retourner systématiquement une erreur 404 signalant une adresse erronnée, le serveur posséde une liste standard de noms de fichiers qu'il s'efforce de trouver dans le répertoire. Cette liste ordonnée est indiquée par la clause DirectoryIndex
- **AccessFileName .htaccess** Cette clause fixe le nom du fichier à trouver dans un répertoire pour que son accès soit protégé, en imposant à l'utilisateur une authentification par nom et mot de passe. Ces comptes sont spécifiques à Apache et n'interfèrent pas avec les comptes Linux.
- **ErrorLog logs/error_log** : Journal d'erreur par défaut

Configuration serveur Apache

- **Timeout 300** Fixe la durée (en secondes) d'attente maximum du serveur d'une réponse à une requête envoyée à un programme extérieur (comme SGBD)
- **KeepAlive on** Autorise les connexions persistantes d'un client, afin de lui permettre l'envoi de plusieurs requêtes sans déconnexion
- **MaxKeepAliverequests 100** avec un plafond fixé pour un client, pour servir aussi d'éventuels autres clients
- **KeepAliveTimeout 15** et un temps d'attente maxi de la requête suivante provenant du même client.
- **ServerName www** Fixe un nouveau nom public pour le serveur, auquel on pourra s'adresser par les URL http://www/. www doit être connu du DNS ou du fichier hosts local.
- **MinSpareServers 4 et MaxSpareServers 20** Nombres maximum et minimum de processus serveurs devant être en permanence disponibles, en attente de nouvelles connexions clientes
- **StartServers 4** Nombre de processus serveurs démarrés à l'initialisation, en plus du processus père. Ceci explique pourquoi la requête ps aux|grep httpd renvoie 5 PID.
- **MaxClients 150** Nombre maximum de processus qu'Apache peut lancer et gérer simultanément. Ce nombre ne peut pas excéder 254
- **MaxRequestsPerChild 500** Nombre maximum de requêtes HTTP traitées par un processus enfant avant qu'il ne soit éliminé.

paramétrage des permissions d'accès

Il est préférable d'être restrictif à la racine. Politique par défaut : accès interdit à tous à partir de / sauf permissions à expliciter après.

```
<Directory />
order deny, allow
deny from all
Options None
AllowOverride None
</Directory>
```

Attention, contrairement aux permissions Linux, les clauses s'appliquent AUSSI à TOUS les sous-répertoires si une directive <Directory rep> spécifique à l'un des sous-répertoires ne s'impose pas.

Permettre accès à la racine

Pour la racine du serveur WEB, il faut bien permettre l'accès

```
<Directory /var/www/html>
Options Indexes Includes FollowSymLinks
# AllowOverride = All pour donner la priorité aux fichiers .htaccess
AllowOverride All
order allow,deny
# allow from = all pour permettre à tout le monde d'accéder aux
    documents
allow from all
</Directory>
```


Exemple de Contrôle d'accès

soit à autoriser tout le réseau 172.16.0. sauf 172.16.0.25. le bon ordre ?

```
allow from 172.16.0.0/255.255.255.0
deny from 172.16.0.25
```

Accéder par l'alias doc aux documents HTML du serveur Linux du /usr/share/doc. On réserve cette consultation aux machines du réseau local. Ordre?

```
Alias /doc /usr/share/doc
<Directory /usr/share/doc>
order deny,allow
deny from all
allow from localhost, 127.0.0.1
allow from .ensa-agadir.ac.ma
Options Indexes FollowSymLinks
</Directory>
```


Serveur web virtuel

Soit le serveur www (adresse IP 192.168.1.11), nous allons créer les hôtes virtuels genux.esta.ac.ma et senux.esta.ac.ma qui vont pointer chacun vers un endroit différent du disque. éditer le fichier /etc/httpd/conf/vhosts/Vhosts.conf et rajouter :

```
NameVirtualHost 192.168.1.11
<VirtualHost 192.168.1.11>
ServerAdmin webmaster@www.esta.ac.ma
DocumentRoot /home/httpd/html/genux
ServerName genux.esta.ac.ma
ErrorLog logs/genux-error_log
CustomLog logs/genux-error_log common
</VirtualHost>

<VirtualHost 192.168.1.11>
ServerAdmin webmaster@www.esta.ac.ma
DocumentRoot /home/httpd/html/senux
ServerName senux.esta.ac.ma
ErrorLog logs/error_log
CustomLog logs/error_log common
</VirtualHost>
```

common défini par logformat

customLog défini format utilisé par fich journal

Authentification des utilisateurs

La protection d'une page pour l'utilisateur ahmed se fait de manière très simple, tous les fichiers à accès limité devant être concentré dans un même répertoire. Dans ce répertoire, il suffit de créer un fichier nommé .htaccess contenant :

```
AuthUserFile auth/ahmed.users
AuthName "Acces Restreint"
AuthType Basic
<Limit GET POST>
require valid-user
</Limit>
```

- Le fichier ahmed.users doit contenir la liste des utilisateurs habilités à accéder au répertoire où se trouve .htaccess. A noter que le fichier .htaccess peut être nommé différemment en utilisant la directive AcessFileName.
- Pour créer ce fichier il suffit de taper :

```
htpasswd -c /etc/httpd/auth/ahmed.users ahmed
```

- L'option -c correspondant à la création du fichier.

Restriction accès aux fichiers

Si vous voulez vous assurez que personne ne puisse consulter les fichiers .htaccess de vos utilisateurs, rajoutez dans le fichier httpd.conf, la directive suivante :

```
<files ~ "/\.ht">
  order deny,allow
  deny from all
  </files>
```


Service DNS

Toumanari – le 16, 17 et 18 décembre 2010

Ancienne solution: hosts.txt

- Un fichier centralisé distribué à toutes les machines sur l'Internet
- Cette fonctionalité existe toujours
  - `/etc/hosts` [Linux/Unix]
  - `c:\windows\system32\drivers\etc\hosts` [Windows]

```
192.168.13.9        poste22
192.168.133.7        genux
192.168.194.33     ensa5
```

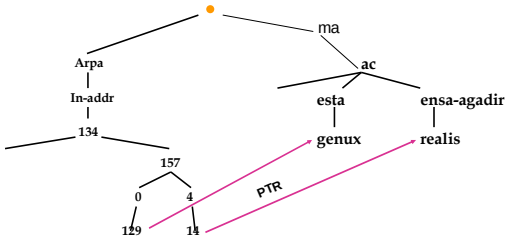

hosts.txt est inadapté à grande échelle

- ✗ Fichier volumineux
- ✗ Nécessite d'être copié ec.maéquemment sur toutes les machines
- ✗ Pas uniforme
- ✗ Pas d'unicité des noms
- ✗ Un seul point d' administration

Le DNS

- DNS est une base de donnée distribuée pour faire correspondre des noms aux adresses IP(et autres informations)
- Distribuée:
  - Administration partagée
  - Charge partagée
- Robustesse et performance à travers:
  - La duplication
  - Le cache

DNS est Hiérarchisé



DNS est Hiérarchisé (2)

- Donne des noms globalement uniques
- Administré en "zones" (parties de l'arbre)
- Vous pouvez donner ("déléguer") le contrôle d'une partie de l'arbre sous votre autorité
- Exemple:
  - esta.ac.ma est sur un ensemble de serveurs
  - tcc.esta.ac.ma sur un ensemble différent
  - tm.esta.ac.ma sur un autre ensemble

Utilisation du DNS

- Un nom de domaine (comme www.ensa-agadir.ac.ma) est une clé de recherche d'informations
- Le resultat est un ou plusieurs enregistrements de ressources (ER)
- Il y a différents ER pour différents types d'informations
- Vous pouvez rechercher un type spécifique, ou rechercher "tous" les ER associés à un nom de domaine

ER courants

- A (adresse IP): associe les noms aux adresses IP
- PTR (pointer): associe les adresses IP aux noms
- MX (mail exchanger): où délivrer les courriers pour utilisateur@*domaine*
- CNAME (canonical name):associe des alias au nom réel
- TXT (text):n'importe quel texte descriptif
- NS (Name Server), SOA (Start Of Authority): Utilisés pour les délégations et le fonctionnement du DNS

Exemple simple

- Requête: www.ensa-agadir.ac.ma
- Type de requête : A
- Resultat:

```
www.ensa-agadir.ac.ma.   IN   A   212.74.101.10
```

- Dans ce cas, un seul ER a été trouvé, mais en général, plusieurs ER peuvent être retournés.

Résultats possibles

- Positif
  - 1 ou plusieurs EE trouvés
- Négatif
  - Définitivement aucun ER ne correspond à la requête
  - Définitivement le nom recherché n'existe pas
- Echec de serveur
  - Ne peut contacter "quelqu'un" qui connait la réponse

Recherche inverse?

- Convertir l'adresse IP au format décimal(A.B.C.D)
- Inverser les quatre parties
- Ajouter ".in-addr.arpa" à la fin (domaine spécial réservé à cette fin)
- e.g. Pour trouver le nom de 212.74.101.10

```
10.101.74.212.in-addr.arpa.
          → PTR www.ensa-agadir.ac.ma.
```

Le DNS est une application Client-Serveur

- Basé sur les sockets
- Requêtes et réponses sont normalement envoyées dans des paquets UDP, port 53
- Utilise occasionnellement TCP, port 53
  - Pour les transferts de zones du maître aux esclaves et pour les grandes requêtes, e.g. > 512 octets

Les types de Serveurs de Noms

- Resolver !
  - pas de résolution des noms des ressources locales
  - résolution des noms des ressources distantes
- Serveur secondaire
  - l'administration des ressources locales est assurée par un tiers
- Serveur primaire
  - administration des ressources locales
  - autorité sur ces informations
- Serveur cache
  - mémorise les requêtes précédentes
  - aucune table locale
- Serveur "forwarding"
  - enrichi le cache d'un (ou plusieurs) autre(s) NS

LE RESOLVER

- Un morceau de logiciel qui formate une requête DNS dans un paquet UDP, l'envoie au serveur cache et décode le resultat
- Généralement une librairie partagée (e.g. `libresolv.so` sous Linux) parce que beaucoup d'applications en ont besoin

Comment le resolver trouve-t-il le serveur cache?

- Doit être configuré explicitement (statique, ou via DHCP etc)
- Doit être configuré avec l'adresse IP du cache
- C'est une bonne idée de configurer plus d'un cache, dans le cas où le premier n'est pas disponible

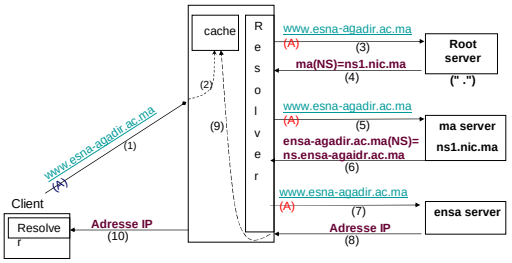
Exemple: Configuration d'un resolver unix

- `/etc/resolv.conf`

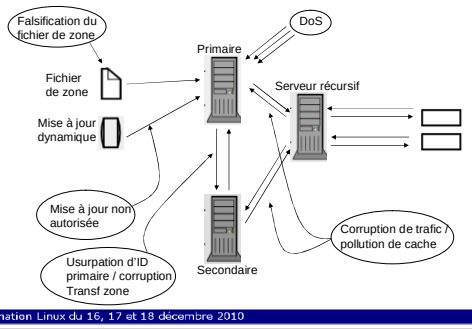
```
Search ensa-agadir.ac.ma
nameserver 196.216.0.21
```

- C'est le minimum dont vous avez besoin pour configurer un resolver

La résolution DNS



Attaque dns



Comment initialiser la cache ?

- Chaque serveur cache est doté d'une liste de serveurs racines

`/usr/local/etc/named.conf`

```
zone "." {
        type hint;
        file "named.ca";
};
```

`named.ca`

```
.                       3600000      NS    A.ROOT-SERVERS.NET.
A.ROOT-SERVERS.NET.     3600000      A     198.41.0.4

.                       3600000      NS    B.ROOT-SERVERS.NET.
B.ROOT-SERVERS.NET.     3600000      A     128.9.0.107

.                       3600000      NS    C.ROOT-SERVERS.NET.
C.ROOT-SERVERS.NET.     3600000      A     192.33.4.12
;... etc
```

Quand a lieu la duplication?

- Les esclavent scrutent le maître périodiquement pour vérifier s'il y a de nouvelles données
  - Seul mécanisme au départ
- Avec les nouveaux logiciels, le maître peut informer les esclaves si les données ont changé (notify)
  - Des mises à jour plus rapides
- Cette notification n'est pas fiable (e.g. Le réseau peut perdre un paquet).Ainsi nous avons toujours besoin de vérifier à "l'intervalle régulier"

Les RR

objet	TTL	CLASSE	TYPE	RDATA f(TYPE, CLASSE)
Nom de Domaine (implicite)	Nb entier (secondes)	IN	A	@IP (32 bits)
			PTR	Nom_Dom.
			SOA	
			NS	Nom_host
			MX	Nom_host
	durée de vie dans le cache		CNAME	Nom_host
			HINFO	Texte
			WKS	Services
			...	

Enregistrement : SOA

- SOA = Start of Authority
- Spécifie que ce serveur de nom a autorité sur le domaine

```
@           IN  SOA     ns.ensa-agadir.ac.ma        root.ensa-agadir.ac.ma.  (
                64      ; serial number
                3600    ; reec.maesh
                600     ; retry
                86400   ; expire
                3600    ) ; minimum TTL
```


Enregistrement : NS

- spécifie les serveurs de nom ayant autorité sur ce domaine

```
;
; Zone NS records
;

ensa-agadir.ac.ma   IN    NS      ns
ns                  IN    A       194.172.2.2
```

Enregistrements : A

hub3	IN	A	193.148.20.16
Hub5	IN	A	193.148.20.17
labo-reseau	IN	A	193.148.80.3
MODEM1	IN	A	193.148.80.4
MODEM2	IN	A	193.148.80.5
Genux	IN	A	193.148.20.1
next	IN	A	193.48.184.3
Ntserv	IN	A	193.148.60.2
ROUTEUR1	IN	A	193.48.184.250
Serv_sun	IN	A	193.148.20.2
Sun_station1	IN	A	193.148.20.3

Enregistrements : CNAME

ftp	IN	CNAME	intranet
gopher	IN	CNAME	intranet
mail	IN	CNAME	intranet
www	IN	CNAME	intranet

Enregistrements : PTR

194.20.148.193.in-addr	IN	PTR	sunstation8.ensa-agadir.ac.ma.
194.20.148.193.in-addr	IN	PTR	sunstation9.ensa-agadir.ac.ma.
194.20.148.193.in-addr	IN	PTR	sunstation10.ensa-agadir.ac.ma.
194.20.148.193.in-addr	IN	PTR	ultra1.ensa-agadir.ac.ma.
194.20.148.193.in-addr	IN	PTR	suntx1.ensa-agadir.ac.ma.
194.20.148.193.in-addr	IN	PTR	sunserv.ensa-agadir.ac.ma.
194.20.148.193.in-addr	IN	PTR	sunstation1.ensa-agadir.ac.ma.
194.20.148.193.in-addr	IN	PTR	sunstation2.ensa-agadir.ac.ma.
194.80.148.193.in-addr	IN	PTR	intranet.ensa-agadir.ac.ma.
194.80.148.193.in-addr	IN	PTR	labo-reseau.ensa-agadir.ac.ma.
194.80.148.193.in-addr	IN	PTR	MODEM1.ensa-agadir.ac.ma.
194.80.148.193.in-addr	IN	PTR	MODEM2.ensa-agadir.ac.ma.

Enregistrement MX

- MX = Mail eXchanger
- L'enregistrement MX est consulté par les mailers (SMTP client)
- Tient compte des priorités; exemple

@	IN	MX	8	relais1. ensa-agadir.ac.ma
@	IN	MX	99	relais2. ensa-agadir.ac.ma

Exemple ensa-hosts

```
ensa-agadir.ac.ma        IN    SOA    genux.ensa-agadir.ac.ma
                                      root.ensa-agadir.ac.ma(
                         2003021618;
                         28800;
                         7200;
                         604800;
                         3600;)
                         IN    NS     genux.ensa-agadir.ac.ma
                         IN    NS     senux.ena-agaidr.ac.ma
www                      IN    A      192.168.1.1
ensa-agadir.ac.ma        IN    MX     10     relais.ensa-agadir.ac.ma
ensa-agadir.ac.ma        IN    MX     30     relais2.ensa-agadir.ac.ma
```


Structure named.conf

Declaration ["*<nom_declaration>*"] [*<class_declaration>*]
{
<option-1>;
<option-2>;
<option-N>;
};

Déclaration acl

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

any — Correspond à toutes les adresses IP.
localhost —toute adresse IP utilisée par le système local.
localnets —toute adresse IP sur tout réseau auquel le système local est connecté.
IP – une IP
IP; IP; IP – liste IPs

exemple

```
acl liste_noire {
10.0.2.0/24;
192.168.0.0/24;
};
acl liste_1 {
10.0.1.0/24;
};
options {
 blackhole { liste_noire; };
 allow-query { liste_1; };
 allow-recursion { liste_1; };
 }
```

Déclaration options

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

- allow-query — Spécifie les hôtes autorisés à interroger ce serveur de noms. Par défaut, tous les hôtes sont autorisés à interroger le serveur de noms. Il est possible d'utiliser ici une liste de contrôle d'accès ou un ensemble d'adresses IP ou de réseaux afin de n'autoriser que des hôtes particuliers à interroger le serveur de noms.
- allow-recursion — Semblable à allow-query, cette option s'applique à des demandes récursives. Par défaut, tous les hôtes sont autorisés à effectuer des demandes récursives sur le serveur de noms.
- blackhole — Spécifie les hôtes qui ne sont pas autorisés à interroger le serveur de noms.

Déclaration options

- directory — Change le répertoire de travail named pour une valeur autre que la valeur par défaut, /var/named/.
- forward — Contrôle le comportement de retransmission d'une directive forwarders.

  Les options suivantes sont acceptées :

  - first — Établit que les serveurs de noms spécifiés dans la directive forwarders soient interrogés avant que named ne tente de résoudre le nom lui-même.
  - only — Spécifie que named ne doit pas tenter d'effectuer lui-même une résolution de nom dans le cas où des demandes vers les serveurs de noms spécifiés dans la directive forwarders échouent.

Directive options

- forwarders : Spécifie une liste d'adresses IP valides correspondant aux serveurs de noms vers lesquels les requêtes devraient être envoyées pour la résolution.
- listen-on : Spécifie l'interface réseau sur laquelle named prend note des requêtes. Par défaut, toutes les interfaces sont utilisées.

  De cette manière, si le serveur DNS sert également de passerelle, BIND peut être configuré de telle sorte qu'il réponde seulement aux requêtes en provenance de l'un des réseaux.:

  options { listen-on { 10.0.1.1; }; };

Directive options

- notify — Établit si named notifie les serveurs esclaves lorsqu'une zone est mise à jour. Les options suivantes sont acceptées :

  yes — Notifie les serveurs esclaves.

  no — Ne notifie pas les serveurs esclaves.

  explicit — Notifie seulement les serveurs esclaves spécifiés dans une liste notify à l'intérieur d'une déclaration de zone.
- pid-file — Spécifie l'emplacement du fichier de processus ID créé par named.
- statistics-file — Spécifie un autre emplacement des fichiers de statistiques. Par défaut, les statistiques named sont enregistrées dans le fichier /var/named/named.stats

Déclaration zone

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

- allow-query — Spécifie les clients qui sont autorisés à demander des informations à propos de cette zone. Par défaut toutes les requêtes d'informations sont autorisées.
- allow-transfer — Spécifie les serveurs esclaves qui sont autorisés à demander un transfert de zone. Par défaut toutes les requêtes de transfert sont autorisées.
- allow-update — Spécifie les hôtes qui sont autorisés à mettre à jour dynamiquement les informations dans leur zone. Par défaut aucune requête de mise à jour dynamique n'est autorisée.

Déclaration zone

- file — Spécifie le nom du fichier qui figure dans le répertoire de travail named et qui contient les données de configuration de la zone.
- masters — Spécifie les adresses IP à partir desquelles demander des informations sur la zone faisant autorité. Cette option ne doit être utilisée que si la zone est définie en tant que type slave.
- notify — Détermine si named notifie les serveurs esclaves lorsqu'une zone est mise à jour. Cette directive accepte les options suivantes :
  yes — Notifie les serveurs esclaves.
  no — Ne notifie pas les serveurs esclaves.
  explicit — Notifie seulement les serveurs esclaves spécifiés dans une liste also-notify à l'intérieur d'une déclaration de zone.

Déclaration zone

- type — Définit le type de zone. Les types énumérés ci-dessous peuvent être utilisés. Ci-après figure une liste des options valides :
  forward — Retransmet toutes les requêtes d'informations concernant cette zone vers d'autres serveurs de noms

  hint — Représente un type spécial de zone utilisé pour diriger des transactions vers les serveurs de noms racines qui résolvent des requêtes lorsqu'une zone n'est pas connue autrement. Aucune configuration autre que la valeur par défaut n'est nécessaire avec une zone hint.

  master — Désigne le serveur de noms faisant autorité pour cette zone. Une zone devrait être configurée comme maître (master) si les fichiers de configuration de la zone se trouvent sur le système.

  slave — Désigne le serveur de noms comme serveur esclave (slave) pour cette zone. Cette option spécifie également l'adresse IP du serveur de noms maître pour cette zone.

déclaration contrôle

controls {
inet 127.0.0.1 allow { localhost; } keys { *<key-name>*; }; };

écoute TCP 953 par défaut de l'adresse inversée et doit autoriser les commandes rndc provenant de l'hôte local. Le *<key-name>* fait référence à la déclaration key, qui se trouve dans le fichier /etc/named.conf. L'exemple suivant illustre une déclaration key.

key "*<key-name>*" {
algorithm hmac-md5;
secret "*<key-value>*";
};

```
controls { inet 127.0.0.1 allow { localhost; } keys { ma_cle; };};

Key “ma_cle“ {
algorithm hmac-md5;
secret « ae/euffd++df/sdefe25ef8epeue++erfe==“; };

options {
Directory /var/named;
{ listen-on { 134.157.1.2; }; };};

zone "." {
        type hint;
        file "named.root";};
zone "ensa-agadir.ac.ma" {
        type master;
        file "ensa/ensa";};
zone "gii.ensa-agadir.ac.ma" {
        type master;
        file "ensa/gii";};
zone "gpee.ensa-agadir.ac.ma" {
        type master;
        file "ensa/gpee";};
```


DNS : named.conf

```
zone "esta.ac.ma" {
        type slave;
        file "slave/esta";
        masters {
                194.10.2.1;
        };
};

zone "fcs.ac.ma" {
        type slave;
        file "slave/fcs";
        masters {
                194.157.3.6;
        };
};
```


DNS : named.conf

```
zone "0.0.127.in-addr.arpa" {
        type master;
        file « ensa/0.0.127.localhost";
};

zone "157.194.in-addr.arpa" {
        type master;
        file « ensa/157.134.ensa";
};

zone "0.157.194.in-addr.arpa" {
        type master;
        file « ensa/00.gii";
};

zone "1.157.194.in-addr.arpa" {
        type master;
        file « ensa/01.gpee";
};
```


Tests DNS avec "dig"

- "dig" est un programme qui fait simplement des requêtes DNS et affiche les résultats
  - Mieux que "nslookup" et "host" pour le débogage, parce qu'il montre les messges DNS au complet

```
dig genux.ensa-agadir.ac.ma.
  - Par défaut recherche le type "A"
dig ensa-agadir.ac.ma. mx
  - specifier le type recherché
dig @212.74.112.66 ensa-agadir.ac.ma. mx
  - Envoie la requête à un cache spécifique
    (outrepasse /etc/resolv.conf)
```

```
# dig www.ensa-agadir.ac.ma. a
; <<>> DiG 9.3.0 <<>> www.ensa-agadir.ac.ma a
;; global options: printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 2462
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 2, ADDITIONAL: 2
;; QUESTION SECTION:
;www.ensa-agadir.ac.ma                   IN   A

;; ANSWER SECTION:
www.ensa-agadir.ac.ma.          86400   IN   CNAME  waib.ensa-agadir.ac.ma.
waib.ensa-agadir.ac.ma.         86400   IN   A      81.91.232.2

;; AUTHORITY SECTION:
ensa-agadir.ac.ma.              86400   IN   NS     ns.amd.ma.
ensa-agadir.ac.ma.              86400   IN   NS     ns.ensa-
agadir.ac.ma.


;; ADDITIONAL SECTION:
ns.ensa-agadir.ac.ma.       86400   IN   A      81.91.232.1
ns.amd.ma.      18205   IN   A      81.91.225.1

;; Query time: 200 msec
;; SERVER: 212.74.112.67#53(212.74.112.67)
;; WHEN: Tue Dec 28 19:50:01 2004
;; MSG SIZE  rcvd: 237
```


Autre Services

- openldap
- Squid
- NFS
- SAMBA
- SSH
- FTP
- SNMPD
- ….