Services Réseaux

TCP/IP

TPC/IP

- Les réseaux sont généralement organisés en «piles protocolaires »
- Chaque couche de la pile offre un niveau d'abstraction supplémentaire à la couche supérieure chaque couche offre un service supplémentaire par rapport à la couche inférieure
- Chaque couche offre un service supplémentaire par rapport à la couche inférieure

TPC/IP

- La couche inférieure est **IP** (Internet Protocol).
- La couche de transport est **TCP** (Transmission Control Protocol) ou **UDP** (User Datagram Protocol).
- Les couches supérieures sont les couches des protocoles applicatifs, par exemple :
  - **NFS** (Network File System) : partage de fichiers à distance.
  - **DNS** (Domain Name System) : association hôte<->IP.
  - **FTP** (File Transfer Protocol) : transfert de fichiers.
  - **TELNET** : émulation d’un terminal de type texte...

Adressage

- **Les Classes**

Il est important de savoir avant l'installation dans quel type de réseau doit s'intégrer le nouveau serveur, TCP/IP bien sûr, mais il faut déjà lui réserver **une adresse IP**, un **hostname** (nom de machine réseau), connaître les diverses passerelles, le nom de domaine, la classe utilisée et le masque de sous-réseau **netmask.**

Adressage

- Une adresse IP (IPV4) est définie sur 32 bits et représentée par quatre nombres séparés par des points : n1.n2.n3.n4.
- Cette adresse est constituée de deux parties qui définissent l'adresse réseau et l'hôte dans le réseau.
- On distinguez cinq classes d'adresses : A, B, C, D et E, mais seules les trois premières nous intéressent.

Adressage

- 4 octets (32 bits)
- 0.0.0.0 ⇔ 255.255.255.255
- Soient 232 adresses possibles (~4.3 milliards)
- n bits pour l’adresse réseau
- (32-n) bits pour l’adresse de l’hôte

Adressage – Les classes

- Classe A: 1.x.x.x à 126.x.x.x
  16777214 hôtes, 127 réseaux
- Classe B: 128.0.x.x à 191.255.x.x
  65534 hôtes, 16382 réseaux.
- Classe C: 192.0.0.x à 223.255.255.x
  254 hôtes, 2097150 réseaux.
- Classe D: Commence par 1110, pour la multidiffusion IP
- Classe E : Commence par 1111, pour expérimentation.

Adressage – Les classes

Les adresses suivantes ne doivent pas être routées sur Internet et sont réservées aux réseaux locaux:

- 10.0.0.0 - 10.255.255.255 (10/8)
- 172.16.0.0 - 172.31.255.255 (172.16/12)
- 192.168.0.0 - 192.168.255.255 (192.168/16)

L’adresse 127.0.0.1 est l’adresse de loopback ou bouclage : elle représente la machine elle-même, ainsi que le sous réseau127.0.0.0/8.

Adressage - Sous-réseaux

- Il est possible de découper ces réseaux en sous-réseaux à l'aide de masques permettant un découpage plus fin des adresses.
- Un **netmask** est un masque binaire qui permet de séparer immédiatement l'adresse du réseau et du sous-réseau de l'adresse de l'hôte dans l'adresse IP globale.
- Les masques prédéfinis sont :
  - Classe A : 255.0.0.0
  - Classe B : 255.255.0.0
  - Classe C : 255.255.255.0

Adressage - Sous-réseaux

- Pour communiquer directement entre eux les hôtes doivent appartenir à un même réseau ou sous-réseau.
- Il est simple de Calculer un sous-réseau. Voici un exemple pour un réseau de classe C.
- Réseau : 192.168.1.0
- Adresse de réseau : 192.168.1.255
- Masque de réseau : 255.255.255.0

Adressage - Masque de sous-réseau

Pour calculer le masque de sous-réseau, il d'abord déterminer combien de machines on souhaite intégrer dans celui-ci.

Un réseau de classe C permet d'intégrer 254 machines (0 et 255 étant réservés).

Si on souhaite de créer des réseaux contenant 60 machines, il faut ajouter 2 à cette valeur pour les adresses réservées (adresse du sous-réseau et adresse de broadcast) ce qui donne 62.

Adressage - Masque de sous-réseau

Une fois le nombre de machines déterminé (62 dans cet exemple), il faut trouver la puissance de deux exacte ou juste supérieure au nombre trouvé.

Pour cet exemple 2 puissance 6 donne 64

Ensuite on écrit le masque en binaire, on place tous les bits du masque de réseau de classe C à 1 et on place à 0 les 6 premiers bits du masque correspondant à la partie machine : 1111111 1 1111111 1 1111111 1 11000000

Ensuite on convertit ce masque en décimal : 255.255.255.192

Adressage - Masque de sous-réseau

- Au final on peut obtenir quatre sous-réseaux de 62 machines, soit 248 machines.

Adressage - Routage

Le masque de réseau permet de déterminer si une machine destinataire est sur le même réseau que notre machine ou non.

Il faut indiquer le chemin que doivent prendre les paquets IP pour rejoindre leur destination.

Si notre machine est un poste client disposant d'une seule carte réseau et que ce réseau ne comporte qu'un seul routeur (cas classique d'une connexion vers Internet) alors il faut créer deux routes:

- La première est celle indiquant quelle carte réseau doivent emprunter les paquets pour accéder au reste du réseau (au sous-réseau),
- la seconde quelle route doivent emprunter les paquets pour sortir du réseau.

Généralement, on parle de route par défaut quand un seul routeur est présent

Routage des paquets IP

- Pour communiquer, les hôtes transmettent leur paquets IP, des datagrammes, à des « routeurs»

- Les "routeurs" sont aussi des hôtes au sens TCP/IP, avec la particularité
  - d'avoir plusieurs interfaces
  - de faire passer des paquets d'une interface à l'autre

Configuration

- Pour être fonctionnelle, la pile IP n'a besoin que d'une chose :

une adresse (avec son masque)

- Pour sortir du LAN, il faudra aussi une passerelle (gateway, next-hop)
- Pour résoudre les noms en adresse, il faut un serveur DNS

Accès Internet = adresse + next-hop (+ DNS)

Configuration - ifconfig

- **ifconfig** permet d'affecter des adresses IP aux interfaces de la machine **ifconfig** existe sur toutes les distributions Unix

ifconfig [interface] [paramètres]

ifconfig

affiche les informations sur toutes les interfaces

ifconfig eth0

affiche les informations sur l'interface eth0

ifconfig eth0 192.168.0.1 netmask 255.255.255.0

affecte l'adresse 192.168.0.1/24 à l'interface eth0

Adresse IP - ip

- Il faut savoir que ifconfig existe , mais il faut utiliser la commande **ip**
- **ip** gère :
  - l'adressage IP (comme **ifconfig**)
  - la table de routage (comme **route**)
  - la table arp (comme **arp**)
  - les devices (comme **ifconfig**)
- mais apporte aussi du nouveau :
  - le policy routing
  - gestion des autres tables de routage
  - syntaxe cohérente et concision

Adresse IP - ip

- Voir les adresses

ip addr show [dev device]

- Affecter une adresse avec ip

ip addr add addresse/msk dev device

ip addr add addresse netmask msk dev device

Adresse IP - ip

- Supprimer une addresse

ip addr del addresse/msk dev device

ip addr del addresse netmask msk dev device

- Supprimer toutes les adresses

ip addr flush dev device

- Mettre en route/couper une l'interface

ip link set device état

(état : up pour mettre en marche, down pour couper)

Adresse IP - ip route

- Voir la table de routage principale

ip route show

- Ajouter une route

ip route add addresse/msk via passerelle

- Supprimer une route

ip route del addresse/msk

- Supprimer toutes les routes

ip route flush table main

- Obtenir la route pour un subnet

ip route get addresse/msk

- Couper le routage vers un subnet

ip route add blackhole addresse/msk

Outils réseaux - ping

ping [c nombre] [f] [s] adresse

nombre : nombre de paquets à envoyer

adresse : adresse IP destinataire

f : flood (au moins 1 paquet toutes les 10ms)

s : taille du paquet en bytes

fping [c nombre] [g] adresse

nombre**:** nombre de paquets à envoyer

adresse : adresse IP destinataire

g : teste un subnet complet

Outils réseaux - traceroute

Quand on tente d'accéder à un hôte distant depuis votre machine, les paquets IP passent souvent par de nombreuses routes, parfois différentes selon le point de départ et de destination, l'engorgement, etc.

Le trajet passe par de nombreuses passerelles (gateways), qui dépendent des routes par défaut ou prédéfinies de chacune d'elles.

La commande **traceroute** permet de visualiser chacun des points de passage de vos paquets IP à destination d'un hôte donné.

Outils réseaux - whois

Whois permet d'obtenir toutes les informations voulues sur un domaine

Exemple:

whois ubuntu.com

Outils réseaux - Netstat

La commande **netstat** permet d'obtenir une foule d'informations sur le réseau et les protocoles.

Le paramètre -i permet d'obtenir l'état des cartes réseaux, afin de déterminer une éventuelle panne ou un problème de câble.

Avec **netstat –ei** , on obtient le même résultat qu'avec **ifconfig –a**

Le paramètre **-r** permet d'obtenir, comme route, les tables de routage. En ajoutant le paramètre -n pour indiquer les adresses ip à la place des noms.

netstat -rn

Outils réseaux - Netstat

- Le paramètre **-a** permet de visualiser toutes les connexions, pour tous les protocoles, y compris les ports en écoute de la machine

netstat –a | less

Outils réseaux - iptraf

La commande **iptraf** permet de visualiser en temps réel l'activité du réseau via un outil texte interactif ou non (ligne de commande).

On peut se déplacer avec les touches fléchées et les divers raccourcis précisés.

Fichiers généraux

/etc/resolv.conf

- Le fichier **/etc/resolv.conf** est utilisé pour indiquer au système quels serveurs de noms et quels domaines interroger pour résoudre les requêtes DNS clientes. Les API sont incluses dans la bibliothèque et les API standards de Linux (il n'y a pas besoin d'ajouter des outils supplémentaires). On appelle cette bibliothèque le **resolver**.
- En configurant **DHCP** ce fichier est en principe mis automatiquement à jour et ne devrait pas être modifié sauf si on a interdit la configuration DNS sur notre client.

/etc/hosts et /etc/networks

Sans même utiliser de serveur de noms , on peut établir une correspondance entre les adresses IP et les noms grâce au fichier /etc/hosts

Vous pouvez faire de même pour nommer les réseaux (ce qui peut être utile pour les tcp_wrappers ou la commande route) dans le fichier **/etc/networks**.

/etc/nsswitch.conf

- Le fichier **/etc/nsswitch.conf** permet de déterminer l'ordre dans lequel le **resolver** (ou d'autres services) récupère ses informations.
- Le fichier **/etc/hosts** est d'abord lu, puis, si le **resolver** ne trouve pas l'information il passe par une résolution **DNS**.

/etc/services

Le fichier **/etc/services** contient la liste des services réseaux connus de Unix ainsi que les ports et protocoles associés.

Il est utilisé par de nombreux services (dont xinetd) et sous-systèmes comme le firewall de Linux.

Ce fichier est indicatif : c'est un fichier de description et de définition : tous les services de ce fichier ne tournent pas forcément sur la machine.

Par contre il est conseillé, lorsqu'on rajoutez un service qui n'est pas présent dans ce fichier, de le rajouter à la fin

/etc/protocols

- Le fichier **/etc/protocols** contient la liste des protocoles connus par Unix.

xinetd

Présentation

Le démon **xinetd** est un « super-service » permettant de contrôler l'accès à un ensemble de services, telnet par exemple.

Beaucoup de services réseaux peuvent être configurés pour fonctionner avec **xinetd**, comme les services **ftp, ssh, samba, rcp, http**, etc.

Des options de configuration spécifiques peuvent être appliquées pour chaque service géré.

Présentation

Lorsqu'un hôte client se connecte à un service réseau contrôlé par **xinetd, xinetd** reçoit la requête et vérifie tout d'abord les autorisations d'accès TCP puis les règles définies pour ce service (autorisations spécifiques, ressources allouées, etc.).

Une instance du service est alors démarrée et lui cède la connexion.

À partir de ce moment **xinetd** n'interfère plus dans la connexion entre le client et le serveur

Configuration

Les fichiers de configuration sont :

- **/etc/xinetd.conf** : configuration globale
- **/etc/xinetd.d/*** : répertoire contenant les fichiers spécifiques aux services. Il existe un fichier par service, du même nom que celui précisé dans /etc/services.

xinetd n'est pas installé par défaut, sur ubuntu, il faut l'installer

```
sudo apt-get install -y xinetd
```

Configuration

```
defaults
{
        instances               = 60
        log_type                = SYSLOG authpriv
        log_on_success          = HOST PID
        log_on_failure          = HOST
        cps                     = 25 30
}
includedir /etc/xinetd.d
```

Contenu du fichier **xinetd.conf :**

- **instances** : nombre maximal de requêtes qu'un service xinetd peut gérer à un instant donné
- **log_type** : dans notre cas, les traces sont gérées par le démon syslog
- **log_on_success** : xinetd va journaliser l'événement si la connexion au service réussit.
- **log_on_failure** : idem mais pour les échecs. Il devient simple de savoir quels hôtes ont tenté de se connecter
- cps : xinetd n'autorise que 25 connexions par secondes à un service. Si la limite est atteinte, xinetd attendra 30 secondes avant d'autoriser à nouveau les connexions.

Démarrage et arrêt des services

On distingue deux cas.

- Premier cas, le service **xinetd** est un service comme un autre dont le démarrage ou l'arrêt peut s'effectuer avec la commande **service** ou directement via l'exécution de /etc/init.d/xinetd.
- Second cas, comme **xinetd** gère plusieurs services, l'arrêt de xinetd arrête tous les services associés, et le démarrage de **xinetd** lance tous les services associés

OpenSSH

Présentation

- OpenSSH est un protocole de shell sécurisé, un mécanisme qui permet l'authentification sécurisée, l'exécution à distance et la connexion à distance.
- Il estcapable d'encapsuler des protocoles non sécurisés en redirigeant les ports
- Les packages à utiliser pour un serveur sont openssh, openssl et openssh-client
- L'utilisation la plus commune reste l'accès distant sécurisé à une machine via le client ssh.

Configuration

- La configuration est /etc/ssh/sshd_config. Quelques options sont éventuellement à modifier :
- **Port** : le numéro de port, par défaut 22
- **Protocol** : fixé à 2,1 il autorise SSH1 et SSH2. On préférera SSH2 et donc on laissera la valeur 2 seule
- **ListenAddress** : par défaut ssh écoute sur toutes les IP du serveur. On peut autoriser uniquement l'écoute

sur une interface donnée

- **PermitRootLogin** : ssh autorise les connexions de root. On peut placer la valeur à « no ». Dans ce cas, il faudra se connecter en simple utilisateur et passer par **su**
- **Banner** : chemin d'un fichier dont le contenu sera affiché aux utilisateurs lors de la connexion.

Ssh est un service System V à lancer avec service ou directement par /etc/init.d/sshd.

Utilisation

La commande **ssh** permet d'établir une connexion.

ssh -l login host

ssh login@host

host : adresse ip du serveur ssh ou nom de la machine

Clés et connexion automatique

Il est possible d'établir une connexion automatique vers une autre machine sans saisir de mot de passe. Pour cela, il est nécessaire depuis le compte utilisateur du client (la machine qui va se connecter) de générer une paire de clés, privée et publique.

Du côté du serveur ssh, la clé publique du client doit être placée dans un fichier contenant les clés autorisées à se connecter dans le compte de destination.

Côté client

Générez une clé au format RSA avec la commande ssh-keygen :

ssh-keygen -t rsa

Suite de quoi le répertoire de l'utilisateur contiendre un répertoire .ssh qui contiendra id_rsa et id_rsa.pub.

Le fichier id_rsa.pub contient la clé publique

Côté serveur

- Aller dans le répertoire .ssh du compte auquel on souhaite accéder sur le serveur (le créer s'il n'existe pas)
- Éditer le fichier **authorized_keys2** (le créer s'il n'existe pas) et y copier sur une nouvelle ligne le contenu du fichier **id_rsa.pub** du client.
- Désormais le client peut se connecter au serveur ssh sans mot de passe

Monter un serveur DHCP

Présentation

Le service DHCP (Dynamic Host Configuration Protocol), protocole de configuration dynamique des hôtes, permet aux hôtes d'un réseau de demander et recevoir des informations de configuration (adresse, routage, DNS, etc.).

Il y a en général un seul serveur DHCP par segment de réseau même si plusieurs sont possibles.

Si le serveur est sur un autre segment, on peut utiliser un agent de retransmission DHCP

Présentation

Autrement dit, un client DHCP recherche tout seul un serveur DHCP qui lui communiquera son adresse IP.

L’adresse IP est assignée soit dynamiquement à partir de plages d’adresses prédéfinies, soit statiquement en fonction de l’adresse MAC du demandeur.

Les informations sont valables un laps de temps donné (un bail) qui peut être renouvelé et configurable.

Serveur dhcpd

Démarrage

Le serveur dhcpd est un service (daemon) lancé à l'aide d'un script (/etc/init.d/dhcpd).

Il est configuré à l'aide du fichier /etc/dhcpd.conf

Serveur DNS

Serveur DNS

Le Système de Noms de Domaines DNS (Domain Name System) transforme les noms d'hôte en adresses IP : c'est la résolution de nom. Il transforme les adresses IP en noms d'hôte : c'est la résolution inverse.

Il permet de regrouper les machines par domaines de nom.

Il fournt des informations de routage et de courrier électronique.

Le DNS permet de faire référence à des systèmes basés sur IP (les hôtes) à l'aide de noms (les noms de domaines).

Les noms de domaine sont plus simples à retenir, et si son adresse IP change l'utilisateur ne s'en rend même pas compte.

Serveur DNS

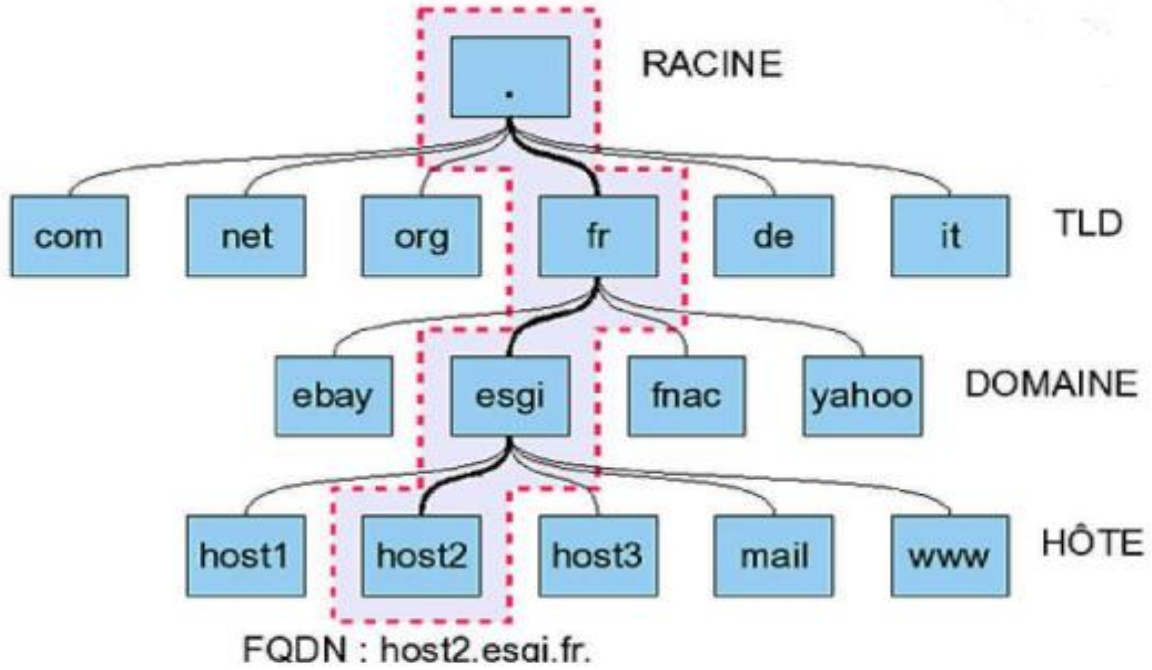
Les noms de domaine sont séparés par des points, chaque élément pouvant être composé de 63 caractères il ne peut y avoir qu'un maximum de 127 éléments et le nom complet ne doit pas dépasser 255 caractères.

Le nom complet non abrégé est appelé FQDN (Fully Qualified Domain Name).

Dans un FQDN, l'élément le plus à droite est appelé TLD (Top Level Domain), celui le plus à gauche représente l'hôte et donc l'adresse IP.

Serveur DNS

Représentation d'une arborescence DNS



Serveur DNS

Une zone est une partie d’un domaine gérée par un serveur particulier. Une zone peut gérer un ou plusieurs sous domaines, et un sous-domaine peut être réparti en plusieurs zones. Une zone représente l’unité d’administration dont une personne peut être responsable.

Une zone est une partie d’un domaine gérée par un serveur particulier. Une zone peut gérer un ou plusieurs sousdomaines, et un sous-domaine peut être réparti en plusieurs zones. Une zone représente l’unité d’administration dont une personne peut être responsable.

Lancement

Le service s’appelle **named**

/etc/init.d/named start

service named start

Configuration de Bind

- Bind (Berkeley Internet Name Daemon) est le serveur de noms le plus utilisé sur Internet.
- Bind9 supporte l'IPv6, les noms de domaine unicode, le multithread et de nombreuses améliorations de sécurité.

Configuration générale

La configuration globale de Bind est placée dans le fichier /etc/named.conf. La configuration détaillée des zones est

placée dans /var/lib/named.

/etc/named.conf est composé de deux parties.

La première concerne la configuration globale des options de Bind.

La seconde est la déclaration des zones pour les domaines individuels.
Les commentaires commencent par un # ou //

Section globale

La configuration globale est placée dans la section options du **/etc/named.conf** .

Voici un détail de quelques options importantes :

- **directory "filename"** : emplacement des fichiers contenant les données des zones.
- **forwarders { adresse-ip }** : si le serveur **bind** ne peut résoudre lui-même la requête, elle est renvoyée à un serveur DNS extérieur, par exemple celui du fournisseur d'accès.
- **listen-on-port 53 {127.0.0.1 adresse-ip }** : port d'écoute du **DNS** suivi des adresses d'écoute. On indique ici les adresses IP des interfaces réseau de la machine. Il ne faut pas oublier 127.0.0.1.
- **allow-query { 127.0.0.1 réseau }** : machine(s) ou réseau(x) autorisés à utiliser le service DNS. Si la directive est absente, tout est autorisé.
- **allow-transfer { 192.168.1.2 }** : machine(s) ou réseau(x) autorisés à copier la base de données dans le cas d'une relation maître et esclave. Par défaut aucune copie n'est autorisée.
- **notify no** : on notifie ou non les autres serveurs DNS d'un changement dans les zones ou d'un redémarrage du serveur

Section de zones

Pour chaque domaine ou sous-domaine, on définit deux sections zone. La première contient les informations de résolution de nom (nom vers IP) et la seconde les informations de résolution inverse (IP vers Nom). Dans chacun des cas, la zone peut être maître **Master** ou esclave **Slave**

Section de zones

- **Master** : le serveur contient la totalité des enregistrements de la zone dans ses fichiers de zone. Lorsqu'il reçoit une requête, il cherche dans ses fichiers (ou dans son cache) la résolution de celle-ci.
- **Slave** : le serveur ne contient par défaut aucun enregistrement. Il se synchronise avec un serveur maître duquel il récupère toutes les informations de zone. Ces informations peuvent être placées dans un fichier.

Dans ce cas l'esclave stocke une copie locale de la base. Lors de la synchronisation, le numéro de série de cette copie est comparé à celui du maître. Si les numéros sont différents, une nouvelle copie a lieu, sinon la précédente continue à être utilisée

Zone de résolution

Elle est généralement appelée **zone**. Pour chaque domaine ou sous-domaine, elle indique dans quel fichier sont placées les informations de la zone (c'est-à-dire et entre autres les adresses IP associées à chaque hôte), son type (maître ou esclave), si on autorise ou non la notification, l'adresse IP du serveur DNS maître dans le cas d'un esclave, etc.

Le nom de la zone est très important puisque c'est lui qui détermine le domaine de recherche. Quand le DNS reçoit une requête, il recherche dans toutes les zones une correspondance.

Zone de résolution inverse

Pour chaque réseau ou sous-réseau IP (ou plage d'adresses) on définit une zone de résolution inverse dont le fichier

contient une association IP vers nom de machine. C'est en fait presque la même chose que la zone de résolution sauf que l'on doit respecter une convention de nommage :

Zone de résolution inverse

- Le nom de la zone se termine toujours par une domaine spécial .in-addr.arpa.
- On doit tout d'abord déterminer quel réseau la zone doit couvrir (cas des sous-réseaux). Pour nous : un

réseau de classe C 192.168.1.0 soit 192.168.1/24.

- On inverse l'ordre des octets dans l'adresse : 1.168.192.
- On ajoute in-addr.arpa. Notre nom de zone sera donc 1.168.192.in-addr.arpa.
- Pour le reste, les mêmes remarques que pour la zone de résolution s'appliquent.

Diagnostic des problèmes de configuration

La commande **named-checkconf** vérifie la syntaxe du fichier **named.conf**.

Elle reçoit en paramètre le fichier.

La sortie indiquera les lignes posant problème.

Interrogation dig et host

Le programme **dig** est un outil d'interrogation avancé de serveur de noms, capable de restituer toutes les informations des zones.

Exemple

dig ubuntu.com

Par défaut **dig** ne restitue que l'adresse de l'hôte passé en paramètre. En cas de réussite, le statut vaut NOERROR, le nombre de réponses est indiqué par ANSWER et la réponse se situe en dessous de la section ANSWER.

Pour une résolution inverse

dig -x AdrIP

Interrogation dig et host

L’outil **host** fournit le même résultat de manière peut-être un peu plus simple.

Exemple:

host ubuntu.com

Service HTTP Apache

Présentation

Apache 2 est le serveur HTTP le plus utilisé actuellement sur les serveurs Web. Sa configuration et sa flexibilité en font un serveur incontournable.

Lorsqu'un serveur Apache reçoit des requêtes, il peut les redistribuer à des processus fils. La configuration permet de lancer des processus de manière anticipée et d'adapter dynamiquement ce nombre en fonction de la charge.

Apache est modulaire. Chaque module permet d'ajouter des fonctionnalités au serveur. Le module le plus connu est probablement celui gérant le langage PHP, « mod_php ».

Présentation

Chaque module s’ajoute via les fichiers de configuration, et il n’y a même pas besoin de relancer le serveur Apache : on lui donne juste l’ordre de relire sa configuration.

Apache peut gérer plusieurs sites Web en même temps, ayant chacun leur nom, à l’aide des hôtes virtuels

Arrêt/Relance

Le nom du service dépend de la distribution. Il est souvent intitulé **apache** ou **httpd** (apache2 pour la distribution ubuntu)

Suivant la distribution on peut lancer le service via la commande service ou directement par son nom **/etc/init.d/apache**.

Configuration

La configuration principale est stockée dans /etc/httpd/conf/httpd.conf (/etc/apache2/apache2.conf pour ubuntu)

Elle contrôle les paramètres généraux du serveur Web, les hôtes virtuels et les accès.

La configuration des différents modules est placée

dans /etc/httpd/conf.d.

Les modules sont présents dans /etc/httpd/modules/.

Par défaut la racine du serveur, celle où sont placées les pages du site, est dans **/var/www** ou **/srv/www**. Cette position dépend de la directive DocumentRoot dans les fichiers de configuration.

Partage de fichiers

NFS - Lancement

Le partage de fichier NFS (Network File System) ou système de fichiers réseau permet de partager tout ou partie de son système de fichiers à destination de clients NFS, bien souvent d'autres Unix. Dans sa version de base c'est un système simple et efficace.

NFS s'appuie sur le portmapper (portmap), le support nfs du noyau et les services rpc.nfsd et rpc.mountd.

Pour lancer le service NFS, portmap et nfs doivent être lancés

NFS – Partage Côté Serveur

La liste des systèmes de fichiers à exporter se trouve dans /etc/exports.
Il contient un partage par ligne.

Exemple:

```
# Rep exportes Autorisations d'acces
/          poste1(rw)   poste2(rw,no_root_squash)
/projects      *.mondomaine.org(rw)
/home/joe    poste*.mondomaine.org(rw)
/pub 192.168.1.0/255.255.255.0(ro)
```

NFS – Partage Côté Serveur

Chaque ligne est composée de deux parties. La première est le chemin du répertoire exporté. La seconde contient les autorisations d'accès.

L'autorisation d'accès est composée de paires hôtes/permissions selon le format suivant :

host(permissions)

L'hôte peut être : un nom d'hôte unique, un domaine, un réseau ou un sous-réseau, une combinaison de l'ensemble avec des caractères de substitution (*, ?).

Les permissions peuvent être : ro : lecture seule, rw : lecture écriture, no_root_squash : le root distant équivaut au root local.

NFS – Montage Côté Serveur

Le support NFS est inclus sous forme de module du noyau. Il est automatiquement chargé à l'utilisation d'un accès NFS.

Dans **/etc/fstab** on a les modifications :

server1:/pub /mnt/pub nfs defaults 0 0

Le périphérique est remplacé par le chemin du partage sous la forme serveur:chemin.

Le système de fichiers est nfs.

C'est identique avec la commande mount

FTP

FTP

Le serveur FTP (File Transfer Protocol) le plus courant est vsftpd (Very Secure FTP Daemon).

Il a l'avantage d'être très petit, performant et rapide tout en étant tout de même très configurable (moins toutefois que Proftpd ou d'autres).

Il convient dans la quasi-totalité des situations. C'est un service qui peut aussi bien être lancé par xinetd.

FTP

Deux niveaux de sécurité sont utilisables :

- Anonyme : tout le monde peut se connecter au serveur FTP en tant que utilisateur ftp ou anonymous.

l’environnement FTP est chrooté.

- Utilisateur : les utilisateurs qui existent sur le serveur peuvent se connecter avec leur mot de passe et ont un

accès complet à leurs données dans leur répertoire personnel.

Samba

Configuration du service Samba

Samba est un service permettant de partager des répertoires et imprimantes entre des stations Linux et des stations Windows.

Pour la configuration de ce service le principal fichier à modifier est **smb.conf** qui se trouve généralement dans **/etc** ou **/etc/samba** selon la distribution.

Le fichier comporte des sections : la configuration globale commence par**[global]** puis ensuite la configuration de chaque partage est aussi définie dans une section dont le nom est entre crochets**[]**

Configuration du service Samba

- **Exemple de fichier de configuration smb.conf**

```
[global]
workgroup = maison
server string = Serveur Samba sur %h
encrypt passwords = true
log file = /var/log/samba/log.%m
max log size = 1000
socket options = TCP_NODELAY
```

La section [global] contient les options communes à tous les répertoires partagés.

Voici quelques options utilisables :

Configuration du service Samba

workgroup

Le nom du groupe de travail. Les ordinateurs du même groupe de travail se retrouvent côte à côte dans le voisinage réseau de Windows.

server string

La description du serveur, qui apparaitra à coté de son nom dans l'explorateur Windows. Si la description contient le terme %h, il sera remplacé par le nom d'hôte de la machine.

Configuration du service Samba

encrypt passwords

Détermine si les mots de passe doivent être cryptés avant d'être transmis. C’est fortement recommandé et tous les systèmes Windows à partir de 98 et NT4 SP3 utilisent cette fonctionnalité par défaut.

log file

Le nom du fichier qui contiendra le journal des activités du serveur. On peut avoir un journal par machine client en utilisant %m dans le nom du fichier. Le %m sera remplacé par le nom de la machine client.

Configuration du service Samba

max log size

Taille maximale du fichier journal

socket options

Indique les options à mettre sur les sockets comme par exemple TCP_NODELAY pour que le système envoi immédiatement les petits paquets sans attendre d'en avoir plusieurs.

Configuration du partage des répertoires

Les partages Samba sont décrits dans des sections ayant la forme suivante :

```
[<nom du partage>]
<option> = <valeur>
...
```

Les paramètres principaux sont les suivants :

Configuration du partage des répertoires

comment

La description du répertoire partagé.

path

Le chemin du répertoire partagé. C'est le contenu du répertoire indiqué qui sera partagé.

read only

Détermine si les clients pourront écrire ou non dans le répertoire partagé.

Configuration du partage des répertoires

public

Autoriser ou non les connexions sans mot de passe.

valid users

Liste des seuls utilisateurs autorisés à se connecter séparés par des espaces. Si on veut autoriser tous les utilisateurs il ne faut pas mettre cette option.

browseable

Détermine si le partage apparaitra dans la liste des partages du serveur.

Configuration du partage des répertoires

La section [**homes**] est un partage particulier. Elle définit le partage des répertoires utilisateur des comptes Linux de la machine.

De nombreuses autres options sont disponibles. Elles sont détaillées dans la page de man de smb.conf

Configuration du partage des répertoires

Exemple

```
[cdrom]
comment = Samba server's CD-ROM
read only = yes
locking = no
path = /cdrom
guest ok = yes
```

```
[partage]
path = /media/d/partage
available = yes
browsable = yes
public = yes
writable = yes
```

Protéger les répertoires partagés

Il est possible de rendre privé un répertoire et d'autoriser ou non des utilisateurs à y accéder.

Pour cela, pour chaque répertoire partagé ajoutez les options:

```
public = no
valid users = <nom des utilisateurs autorisés à accéder aux
répertoires>
```

Protéger les répertoires partagés

Pour chaque nom utilisateur saisi dans cette partie, il faut ajouter l'utilisateur samba avec

```
smbpasswd -a <nom de l'utilisateur>
```

Un compte Linux du même nom doit exister. Si ce n'est pas le cas, il faut le créer avec la commande adduser .

Lancement et arrêt du service

- Pour lancer le service
  /etc/init.d/samba start
- Pour l'arrêter
  /etc/init.d/samba stop
- Pour le relancer
  /etc/init.d/samba restart

Les modifications du fichier **smb.conf** sont prises en compte pour chaque nouvelle connexion. Pour les rendre effectives sur les connexions déjà établies, il faut relancer Samba.

Accès aux répertoires

Pour accéder au partage sous Windows, il suffit d'ouvrir le voisinage réseaux d'une station Windows et de vérifier si la machine y est.

Pour se connecter en ligne de commande à un partage à partir de Linux, on peut utiliser la commande:

smbclient //nomduserveur/nomdupartage – U utilisateur

Il est également possible de monter un partage Samba avec

smbmount //nomduserveur/nomdupartage repertoirelocal

Administration réseau sous Linux/
TCP Wrapper

Le service **TCP wrappers** permettent de contrôler et de restreindre l'accès à certains services réseau. Il utilise le démon **tcpd** qui intercepte les demandes de connexion à un service et vérifie dans les fichiers **hosts.allow** et **hosts.deny** si le client est autorisé à utiliser ce service. Sur les versions de linux actuelles, il est installé par défaut. Par contre il n’est pas actif dans sa partie contrôle d'accès.

TCP Wrappers est un élément à mettre en oeuvre pour sécuriser une machine sous linux, il ne peut toutefois pas remplacer complètement un vrai FireWall.

Principe

Lorsque vous souhaitez vous connecter sur une machine distante avec la commande telnet par exemple, le démon **inetd** intercepte votre demande de connexion et vérifie dans le fichier **inetd.conf** si le service telnet est utilisable.

Si la réponse est positive, votre demande est passée à **tcpd** qui vérifie dans les fichiers **hosts.allow** et **hosts.deny** si vous avez le droit de vous connecter en telnet, si cela est le cas votre demande de connexion sera autorisée, sinon vous serez rejeté.

Dans tous les cas de figure **TCP_wrappers tcpd** transmettra à **syslogd** (démon de log) votre demande (cette demande se retrouvera logger dans le fichier **/var/log/securite**).

L’installation

Par défaut il est installé avec la plupart des distributions, mais au cas, le paquet à installer est : tcp_wrappers-x.....rpm.

TCP_wrappers utilise les fichiers suivants : **tcpd**, **inetd**, **inetd.conf**, **hosts.allow**, **hosts.deny**, **tcpdchk**, **tcpdmatch**.

Configuration : le fichier inetd.conf

Ce fichier se trouve dans le répertoire /etc.

Vous pouvez activer ou désactiver ici des services, en plaçant un # devant la ligne ou en l'enlevant, puis en obligeant la relecture du fichier avec la commande **killall -HUP inetd**.

Il est possible d'ajouter d'autres services dans ce fichier.

Configuration : le fichier inetd.conf

Voici un exemple d'une ligne qui concerne le service ftp dans inetd.conf:

```
ftp stream tcp nowait root /usr/sbin/tcpd in.ftpd
```

avec:

- **ftp**: le nom du service tel qu'il est déclaré dans /etc/services
- **stream**: type de service de transport de données (il existe stream pour tcp, dgram pour udp, raw pour IP)
- **tcp**: nom du protocole tel qu'il existe dans /etc/protocols

Configuration : le fichier inetd.conf

- **wait**: état d'attente, si l'état est wait inetd doit attendre que le serveur ait restitué la socket avant de se remettre à l'écoute. On utilise wait plutôt avec les types dgram et raw. L'autre possibilité est nowait qui permet d'allouer dynamiquement des sockets à utiliser avec le type stream.
- **root**: Nom de l'utilisateur sous lequel le daemon tourne
- **/usr/sbin/tcpd in.ftpd** Chemin d'accès au programme in.ftpd lancé par inetd (il estpossible ici d'ajouter les options de démarrage du programme.

Hosts.allow et Hosts.deny

Vous trouverez ces deux fichiers dans le répertoire /etc.

Le premier fichier lu est **hosts.allow** , puis hosts.deny. Si une requête est autorisée dans le fichier **hosts.allow** alors elle est acceptée, quelque soit le contenu du fichier **hosts.deny.**

Si une requête ne satisfait aucune règle, que ce soit dans **hosts.allow** ou **hosts.deny** alors elle est autorisée.

Si on ne met rien dans **hosts.deny**, alors vous n'avez rien fait.

hosts.allow

Voici un exemple d'un fichier **hosts.allow**

```
# hosts.allow
ALL : LOCAL
in.ftpd : 192.168.0.,10.194.168.0/255.255.255.0, 192.168.1.1
in.telnetd : .iut.u-clermont1.fr
```

On autorise tout les ports depuis un accès local, et on autorise ftp pour les machines venant du réseau 192.168.0.0, ainsi que les machines du réseau 10.194.168.0 avec une autre notation et enfin la seule machine qui a pour adresse 192.168.1.1

hosts.deny

Voici un exemple d’un fichier **hosts.deny**

```
#hosts.deny
ALL:ALL
```

Le fichier **hosts.deny** est simple à comprendre, il interdit tout par défaut.

Autorisation et interdiction de service

hosts.allow indique les services qu'on veut autoriser (Le nom du service doit être identique au nom qui se trouve dans inetd.conf). la syntaxe est la suivante :

```
deamon[,deamon,...] : client[,client,...] [ : option : option ...]
```

Cette syntaxe est identique dans les deux fichiers, hosts.allow et **hosts.deny**.

Les utilitaires de tcp wrappers

- tcpdchk -av : permet de voir la configuration de **tcp wrappers**
- tcpdmatch in.ftpd localhost pour simuler une connexion sur in.ftpd

Tcpdump

Présentation

Dans un réseau ethernet relié par hub, chaque machines reçoit tous
les paquets qui circulent sur le réseau. En fonctionnement normal, les cartes réseau ne réceptionnent que les paquets qui leur sont destinés, mais on peut faire en sorte qu'elles transmettent tous les paquets au système.
Les hub sont de moins en moins utilisés. Ils sont généralement remplacés par des switch qui savent déterminer (en fonction de l'adresses MAC) sur quel
câble il faut envoyer un paquet.
Les machines ne reçoivent donc généralement que les paquets qui leur sont destinés.
L'utilitaire **tcpdump** permet d'inspecter les paquets qui sont reçus et transmis par une carte réseau.

Filtrage

Il est possible de sélectionner les paquets à "écouter" en fonction d'expressions.

Ainsi, ne seront affichées / traitées que les informations pour lesquelles le résultat de l'expression est vérifié.

Une expression est composée de **primitives** et **d'opérateurs logiques**.

Filtrage

Une **primitive** est un identifiant précédé de mots clés qui indiquent le type de l'identifiant.

Par exemple la primitive **src port 21** contient les éléments suivants :

- le mot clé **src** qui indique que l'identifiant ne porte que sur la source du paquet
- le mot clé **port** qui indique que l'identifiant est le port du paquet
- l'identifiant **21**

La primitive correspond donc au port source 21.

Filtrage

Les primitives les plus courantes sont les suivantes

src <adresse> l'adresse source est <adresse>

dst <adresse> l'adresse destination est <adresse>

host <adresse> l'adresse source ou destination est <adresse>

port <port> le port source ou destination est <port>

src port <port> le port source est <port>

dst port <port> le port destination est <port>

portrange <port1>-<port2> le port est compris entre <port1> et <port2>. On peut préciser l'origine avec les mots clés src ou dst et le protocole avec les mots clés tcp ou udp.

Filtrage

Les primitives peuvent être reliées avec les opérateurs logiques **and**, **or** et **not**. Par exemple l'expression suivante va trouver tous les paquets en provenance de tiny mais dont le port n'est pas le port ssh :

```
src tiny and not port ssh
```

Filtrage

Plusieurs options permettent de modifier le comportement de tcpdump :

-i <interface> sélectionne l'interface réseau sur laquelle tcpdump écoute. Par défaut il prend la première active (sauf lo).

-x affiche également les données contenues dans les paquets trouvés, sous forme

hexadécimale

-X affiche les données des paquets sous forme ASCII

-s <nombre> par défaut, seuls les 68 premiers octets de données sont affichés. Ce paramètre permet de modifier ce nombre.

Filtrage

Exemple

tcpdump src 192.168.0.1

Ici, les seuls paquets affichés sont ceux en provenance de 192.168.0.1. Nous pouvons également préciser nos préférences en ajoutant un critère:

tcpdump src 192.168.0.1 and port 80

Là, le seul port qui nous intéresse est 80 (http).

UFW (uncomplicated Firewall)

UFW

L'outil de configuration de pare-feu par défaut pour Ubuntu est **ufw**. Développé pour faciliter la configuration du pare-feu iptables, ufw fournit un moyen simple de créer un pare-feu.

Fonctionnement d’UFW

L’établissement des règles par défaut fait que le système accepte tout sur l’interface de la boucle locale (loopback) et en sortie.

En entrée, toujours en ACCEPT par défaut, la règle est configurable et n’accepte que quelques services (ICMP, udp).

es paquets jetés sont enregistrés (LOG) avec une limite.

Syntaxe et exemple

ufw allow|deny [proto <protocole>]

[from <addresse> [port <port>]] [to <addresse> [port <port>]]

Une fois ufw lancé avec les règles par défaut, un ping venant d'une autre machine fonctionne mais pas d'autres services.

Autrement dit, si vous avez par exemple un service Apache (WEB), il n'est plus accessible à partir d'autres postes. Il faut le rétablir:

ufw allow proto tcp from 192.168.3.1 to 192.168.3.134 port 80

Syntaxe et exemple

ufw allow proto tcp from 192.168.3.1 to 192.168.3.134 port 80

Ce qui se traduit par : accepter pour le protocole TCP les demandes faites par la machine d'IP 192.168.3.1 pour 192.168.3.134 (IP du serveur WEB) sur le port 80.

Un exemple plus simple **: ufw allow ssh**
Dans ce cas, la permission est donnée pour une connexion SSH à pa rtir de n'importe quel poste externe. Voir le manuel en ligne d'ufw pour plus d'exemples. On eput remplacer la commande par

ufw allow to any port 22 from any

Vérification

La vérification de ces règles se fait par la commande :

ufw status

Cette commande permet d'afficher l'ensemble des règles ufw configurées sur votre serveur.

La commande

ufw status numbered

Permet d'afficher les règles configurées sur le serveur numérotées.

Destruction d’une règle

La suppression d’une règle simple dans sa syntaxe de base, nécessite de connaître la règle exacte précédemment utilisée.

Pour cela on utilise la commande

ufw status numbered

Pour déterminer le numéro de la règle qu’on veut effacer, ensuite on utilise la commande ufw delete:

ufw delete numero_regle