Tutoriel sur les serveurs

Ensemble de documents réalisés pour la freeduc−sup.

Table des matières

Liste des illustrations

Chapitre 1. Eléments de cours sur TCP/IP

La suite de protocoles TCP/IP

La suite de protocoles TCP/IP

Le document présente la suite de protocoles TCP/IP.

Ce document sert d'introduction à l'ensemble des cours et TP sur les différents protocoles

1.1. Présentation de TCP/IP

TCP/IP est l'abréviation de Transmission Control Protocol/Internet Protocol. Ce protocole a été développé, en environnement UNIX, à la fin des années 1970 à l'issue d'un projet de recherche sur les interconnexions de réseaux mené par la DARPA (Defense Advanced Research Projects Agency) dépendant du DoD (Department of Defense) Américain.

TCP/IP ,devenu standard de fait, **est actuellement la famille de protocoles réseaux qui gère le routage la plus répandue** sur les systèmes Unix et Windows, et surtout, c'est le protocole de l'Internet.

Plusieurs facteurs contribuent à sa popularité :

Maturité, Ouverture, Absence de propriétaire, Richesse (il fournit un vaste ensemble de fonctionnalités), Compatibilité (différents systèmes d'exploitation et différentes architectures matérielles), et le développement important d'Internet.

La famille des protocoles TCP/IP est appelée **protocoles Internet**, et a donné son nom au réseau du même nom. Leurs spécifications sont définies dans des documents du domaine public appelés RFC (Request For Comments − Appels à commentaires). Ils sont produits par l'IETF (Internet Engineering Task Force) au sein de l'IAB (Internet Architecture Board).

La RFC 826, par exemple, définit le protocole ARP.

Le datagramme correspond au format de paquet défini par le protocole Internet. Les cinq ou six (sixième facultatif) premier mots de 32 bits représentent les informations de contrôle appelées en−tête.

Figure 1−1. datagramme IP

La longueur théorique maximale d'un datagramme IP est de 65535 octets. En pratique la taille maximale du datagramme est limitée par la longueur maximale des trames transportées sur le réseau physique. La fragmentation du datagramme (définie dans le 2ème mot de 32 bits) devient alors nécessaire dès que sa taille ne lui permet plus d'être directement transporté dans une seule trame physique. Les modules internet des équipements prennent en charge le découpage et le réassemblage des datagrammes.

Le protocole Internet transmet le datagramme en utilisant l'adresse de destination contenue dans le cinquième mot de l'en−tête. L'adresse de destination est une adresse IP standard de 32 bits permettant d'identifier le réseau de destination et la machine hôte connectée à ce réseau.

Dans un réseau TCP/IP, on assigne généralement un nom à chaque hôte. Le terme d'hôte est pris dans son sens large, c'est à dire un "noeud de réseau". Une imprimante, un routeur, un serveur, un poste de travail sont des noeuds qui peuvent avoir un nom d'hôte, s'ils ont une adresse IP.

1.2. OSI et TCP/IP

Bien que le protocole TCP/IP ait été développé bien avant que le modèle OSI apparaisse, ils ne sont pas totalement incompatibles. L'architecture OSI est définie plus rigoureusement, mais ils disposent tous deux d'une architecture en couches.

Les protocoles **TCP** et **IP** ne sont que deux des membres de la suite de protocoles TCP/IP qui constituent le **modèle DOD** (modèle en 4 couches). Chaque couche du modèle TCP/IP correspond à une ou plusieurs couches du modèle OSI (*Open Systems Interconnection*) défini par l'ISO (*International Standards Organization*) :

Figure 1−2. OSI et TCP/IP

Des relations étroites peuvent être établies entre la couche réseau et IP, et la couche transport et TCP.

TCP/IP peut utiliser une grande variété de protocoles en couche de niveau inférieur, notamment X.25, Ethernet et Token Ring. En fait, TCP/IP a été explicitement conçu sans spécification de couche physique ou de liaison de données car le but était de faire un protocole adaptable à la plupart des supports.

1.3. La suite de protocoles TCP / IP

Les protocoles TCP/IP se situent dans un modèle souvent nommé "famille de protocoles TCP/IP".

Les protocoles **TCP** et **IP** ne sont que deux des membres de la suite de protocoles IP.

1.3.1. IP (*Internet Protocol*, Protocole Internet)

*IP est un protocole qui se **charge de l'acheminement des paquets** pour tous les autres protocoles de la famille TCP/IP.* Il fournit un système de remise de données optimisé sans connexion. Le terme « optimisé » souligne le fait qu'il ne garantit pas que les paquets transportés parviennent à leur destination, ni qu'ils soient reçus dans leur ordre d'envoi. . La fonctionnalité de somme de contrôle du protocole ne confirme que l'intégrité de l'en−tête IP. Ainsi, seuls les protocoles de niveau supérieur sont responsables des données contenues dans les paquets IP (et de leur ordre de réception).

Le protocole IP travaille en **mode non connecté**, c'est−à−dire que les paquets émis par le niveau 3 sont **acheminés de manière autonome** (datagrammes), sans garantie de livraison.

1.3.2. TCP (*Transmission Control Protocol,* Protocole de contrôle de la transmission)

TCP est probablement le protocole IP de niveau supérieur le plus répandu. ***TCP fournit un service sécurisé de remise des paquets.*** *TCP fournit un protocole fiable, orienté connexion **, au−dessus d'IP** (ou encapsulé à l'intérieur d'IP).* TCP garantit l'ordre et la remise des paquets, il vérifie l'intégrité de l'en−tête des paquets et des données qu'ils contiennent. TCP est responsable de la retransmission des paquets altérés ou perdus par le réseau lors de leur transmission. Cette fiabilité fait de TCP/IP un protocole bien adapté pour la transmission de données basée sur la session, les applications client−serveur et les services critiques tels que le courrier électronique.

La fiabilité de TCP a son prix. Les en−têtes TCP requièrent l'utilisation de bits supplémentaires pour effectuer correctement la mise en séquence des informations, ainsi qu'un total de contrôle obligatoire pour assurer la fiabilité non seulement de l'en−tête TCP, mais aussi des données contenues dans le paquet. Pour garantir la réussite de la livraison des données, ce protocole exige également que **le destinataire accuse réception des données**.

Ces accusés de réception (ACK) génèrent une activité réseau supplémentaire qui diminue le débit de la transmission des données au profit de la fiabilité. Pour limiter l'impact de cette contrainte sur la performance, la plupart des hôtes n'envoient un accusé de réception que pour un segment sur deux ou lorsque le délai imparti pour un ACK expire.

Sur une connexion TCP entre deux machines du réseau, les messages (ou paquets TCP) sont acquittés et délivrés en séquence.

1.3.3. UDP (*User Datagram Protocol*)

UDP est un complément du protocole TCP qui offre un **service de datagrammes sans connexion** qui ne garantit ni la remise ni l'ordre des paquets délivrés. Les sommes de contrôle des données sont facultatives dans le protocole UDP. Ceci permet d'échanger des données sur des réseaux à fiabilité élevée sans utiliser inutilement des ressources réseau ou du temps de traitement. Les messages (ou paquets UDP) sont transmis de manière autonome (sans garantie de livraison.).

Le protocole UDP prend également en charge l'envoi de données d'un unique expéditeur vers plusieurs destinataires.

Ex: TFTP(trivial FTP) s'appuie sur UDP, NT4 utilise UDP pour les Broadcast en TCP−Ip

1.3.4. ICMP (*Internet Control Message Protocol*)

ICMP : protocole de messages de contrôle, est un **protocole de maintenance**. Il permet à deux systèmes d'un réseau IP de partager des informations d'état et d'erreur. Utilisé pour les tests et les diagnostics

La commande **ping** utilise les paquets ICMP de demande d'écho et de réponse en écho afin de déterminer si un système IP donné d'un réseau fonctionne. C'est pourquoi l'utilitaire **ping** est utilisé pour diagnostiquer les défaillances au niveau d'un réseau IP ou des routeurs.

1.3.5. RIP *(Routing Information Protocol)*

RIP est un protocole de routage dynamique qui permet l'échange d'informations de routage sur un inter−réseau. Chaque routeur fonctionnant avec RIP échange les identificateurs des réseaux qu'il peut atteindre, ainsi que la distance qui le sépare de ce réseau *(nb de sauts=nb de routeurs à traverser).* Ainsi chacun dispose de la liste des réseaux et peut proposer le meilleur chemin.

1.3.6. ARP (*Address Resolution Protocol*)

Le protocole ARP permet de déterminer l'adresse physique (ou MAC) d'un noeud à partir de son adresse IP en effectuant une diffusion du type "*qui est X2.X2.X2.X2 ? "*

Figure 1−3. Protocoles TCP/IP et OSI

1.3.7. Fonctionnement général

Pour désigner les informations transmises et leur enveloppe, selon le niveau concerné, on parle de ***message****(ou de flux) entre applications, de* ***datagramme*** *(ou segment) au niveau TCP, de* ***paquet*** *au niveau IP, et enfin, de* ***trames*** *au niveau de l'interface réseau (Ethernet ou Token Ring).*

Les protocoles du niveau application les plus connus sont :

- **HTTP** (Hyper Text Transfer Protocol) permet l'accès aux documents HTML et le transfert de fichiers depuis un site WWW
- **FTP** (File Transfer Protocol) pour le transfert de fichiers s'appuie sur TCP et établit une connexion sur un serveur FTP
- **Telnet** pour la connexion à distance en émulation terminal, à un hôte Unix/Linux.
- **SMTP** (Simple Mail Transfer Protocol) pour la messagerie électronique (UDP et TCP)
- **SNMP** (Simple Network Management Protocol) pour l'administration du réseau
- **NFS** (Network File System) pour le partage des fichiers Unix/Linux.

1.4. Les applications TCP−IP

1.4.1. Modèle client/serveur

Les applications réseaux fonctionnent sur le modèle client/serveur. Sur la machine serveur un processus serveur (daemon) traite les requêtes des clients. Client et serveur dialoguent en échangeant des messages qui contiennent des requêtes et des réponses.

Prenons par exemple telnet.

Figure 1−4. Exemple Telnet

Figure 1−5. Modèle client/serveur

1.4.2. L'adressage des applicatifs : les ports

Une fois le datagramme transmis à l'hôte destinataire, il doit parvenir à l'utilisateur (si le système est multi−utilisateur) et à l'application visée (si le système est multi−tâches).

- sur la machine cliente, l'utilisateur (usager ou programme) effectue une requête vers une machine IP serveur sur le réseau. (par exemple *telnet host* ou *ftp host*). Cela se traduit par la réservation d'un port de sortie TCP ou UDP et l'envoi d'un paquet IP à la machine serveur. Ce paquet contient un message TCP ou UDP avec un numéro de port correspondant à l'application demandée sur le serveur.
- sur le serveur, la requête est réceptionnée par le pilote IP, aiguillée vers TCP ou UDP puis vers le port demandé. Le processus serveur correspondant est à l'écoute des appels sur ce port (par ex: le daemon *telnetd* traite les requêtes *telnet*, le daemon *ftpd* traite les requêtes *ftp*).
- processus client et processus serveur échangent ensuite des messages.

Des numéros de port (entre 0 et 1023) sont réservés pour les applications « standards : les ports « bien connus » (Well Known Ports), ils ont été assignés par l'IANA. Sur la plupart des systèmes ils peuvent être seulement employés par des processus du système (ou root) ou par des programmes exécutés par les utilisateurs privilégiés (liste complète : http://www.iana.org/assignments/port−numbers ou dans le fichier **/etc/services** y compris sous Windows).

D'autres numéros de port sont disponibles pour les applications développées par les utilisateurs (1024 à 65535).

Figure 1−6. Ports applicatifs

*On identifie le protocole de communication entre applications par un **numéro de protocole** et l'application par un **numéro de port**.*

Par exemple, les serveurs HTTP dialoguent de manière traditionnelle par le port 80 :

http ://www.sncf.com/index.htm <=> http :// www.sncf.com:80/index.htm

Les numéros de protocole et de port sont inclus dans le datagramme.

Une fois la connexion établie entre le client et le serveur, ceux−ci peuvent s'échanger des informations selon un protocole défini selon l'applicatif. Le client soumet des requêtes auxquelles répondra le serveur.

Ce mode de communication s'appuie sur la couche "socket". Cette couche est une interface entre la couche présentation et transport. Elle permet la mise en place du canal de communication entre le client et le serveur.

On peut schématiquement dire qu'un socket fourni un ensemble de fonctions. Ces fonctions permettent à une application client/serveur d'établir un canal de communication entre 2 ou plusieurs machines, qui utilisent un protocole de transport (TCP ou UDP) et un port de communication.

1.4.2.1. Les ports prédéfinis à connaître

Service réseau	*N°de Port*	*Type*	*Commentaire*
ICMP	7	TCP/UDP	Commandes Ping
Netstat	15	TCP/UDP	Etat du réseau
FTP	21	TCP	Transfert de fichiers
Telnet	23	TCP	Connexion de terminal réseau
SMTP	25	TCP	Envoi de courrier
DNS	53	TCP/UDP	Serveurs de noms de domaine
HTTP	80	TCP	Serveur Web
Pop3	110	TCP	Réception de courrier
sftp	115	TCP	Transfert de fichiers sécurisé
nntp	119	TCP	Service de news
ntp	123	UDP	Protocole temps réseau
nbname	137	TCP/UDP	Service de Nom Netbios
imap	143	TCP/UDP	Protocole d'accès messagerie Internet
SNMP	161	UDP	Gestion de réseau

Chapitre 2. Eléments de cours sur l'adressage IP

Le document présente l'adressage IP sur un réseau local et en environnement routé

Ce document sert d'introduction à l'ensemble des cours et TP sur les différents protocoles

Mots clés : Adressage physique, Adresse IP, masque, sous−réseau, routage

2.1. Adresses physiques (MAC) et adresses logiques (IP)

2.1.1. Notion d'adresse Physique et de trames

Deux cartes réseaux qui communiquent s'échangent des messages (suite de bits) appelés trames (frame). Tous les postes connectés au même câble reçoivent le message, mais seul celui à qui il est destiné le lit.

Comment sait−il que cette trame lui est adressée ?

Car il reconnaît l'adresse de destination, contenue dans la trame comme étant la sienne.

Comment sait il qui lui a envoyé la trame ?

Car la trame contient aussi l'adresse de l'émetteur.

Au niveau de la couche liaison, les noeuds utilisent une adresse dite « physique » pour communiquer. L'adresse correspond à l'adresse de la carte réseau. On parle **d'adresse physique**, **d'adresse MAC** (Medium Access Control) ou d'adresse de couche 2 (référence au modèle OSI).

Cette adresse est identique pour les réseaux Ethernet, Token Ring et FDDI. **Sa longueur est de 48 bits** soit six octets (par exemple : 08−00−14−57−69−69) définie par le constructeur de la carte. Une adresse universelle sur 3 octets est attribuée par l'IEEE à chaque constructeur de matériel réseau. Sur les réseaux CCITT X.25, c'est la norme X.121 qui est utilisée pour les adresses physiques, qui consistent en un nombre de 14 chiffres.

L'adresse MAC identifie de manière unique un noeud dans le monde. Elle est physiquement liée au matériel (écrite sur la PROM), c'est à dire à la carte réseau.

2.1.2. Notion d'adresse logique et de paquets

L'adresse d'une carte réseau correspond à l'adresse d'un poste et d'un seul. Or les postes sont généralement regroupés en réseau.

Comment identifier le réseau auquel appartient le poste ?

Il faut une adresse logique qui soit indépendante de l'adresse physique.

C'est ce que propose le protocole IP et le protocole IPX.

Pourquoi identifier le réseau ?

Pour permettre à 2 postes qui ne sont pas connectés au même réseau de communiquer.

Cela est impossible avec une adresse MAC, il faut une adresse de niveau supérieur, comme nous le verrons un peu plus loin et surtout avec le routage IP.

Le message véhiculé par la trame va contenir une autre adresse destinataire dont un des objectifs sera de définir le réseau destinataire du message. On appelle le message contenu dans une trame un **paquet**.

Ce qu'il nous faut savoir à ce stade, c'est qu'une machine sait que le paquet n'est pas destiné au réseau si l'adresse réseau de destination est différente de la sienne, dans ce cas elle envoie le paquet à une machine spéciale (la passerelle ou routeur) dont le rôle est d'acheminer les paquets qui sortent du réseau.

Cette adresse dite **logique** du noeud (car elle est attribuée par logiciel à un **hôte**, plus précisément à une carte réseau) contenue dans le paquet est l'adresse IP, est définie indépendamment de toute topologie d'ordinateur ou de réseau. Son format reste identique quel que soit le support utilisé.

Les machines (hôtes) d'un réseau TCP/IP sont identifiées par leur adresse IP.

3 − Résolution d'adresses logiques en adresses physiques

Toute machine sur un réseau IP a donc 2 adresses, une adresse MAC et une adresse IP.

Les processus de niveaux supérieurs utilisent toujours l'adresse IP et donc lorsqu'un processus communique avec un autre processus, il lui envoie un message dont l'adresse destinataire est une adresse IP, mais pour pouvoir atteindre la carte réseau du

destinataire, il faut connaître son adresse MAC. Le rôle du protocole ARP (Adress Resolution Protocol) est d'assurer la correspondance entre l'adresse IP et l'adresse MAC.

2.1.3. Attribution d'une adresse IP Internet

Les réseaux connectés au réseau Internet mondial doivent obtenir un identificateur de réseau officiel auprès du bureau de l'Icann de l'Inter−NIC (Network Information Center) afin que soit garantie **l'unicité** des identificateurs de réseau IP sur toute la planète. Une adresse est attribuée au réseau privé dont l'administrateur en fait la demande auprès du NIC (http://www.nic.fr).

Après réception de l'identificateur de réseau, l'administrateur de réseau local doit attribuer des identificateurs d'hôte uniques aux ordinateurs connectés au réseau local. Les réseaux privés qui ne sont pas connectés à Internet peuvent parfaitement utiliser leur propre identificateur de réseau. Toutefois, l'obtention d'un identificateur de réseau valide de la part du centre InterNIC leur permet de se connecter ultérieurement à Internet sans avoir à changer les adresses des équipements en place.

Chaque noeud (interface réseau) **relié à l'Internet doit posséder une adresse IP unique**.

2.2. Adressage IP

2.2.1. Structure des adresses IP

Les **adresses IP** sont des **nombres de 32 bits** qui contiennent 2 champs :

- Un **identificateur de réseau** (NET−ID): tous les systèmes du même réseau physique doivent posséder le même identificateur de réseau, lequel doit être unique sur l'ensemble des réseaux gérés.
- Un **identificateur d'hôte** (HOST−ID): un noeud sur un réseau TCP/IP est appelé hôte, *il identifie une station de travail, un serveur, un routeur ou tout autre périphérique TCP/IP au sein du réseau.*

La concaténation de ces deux champs constitue une **adresse IP unique** sur le réseau.

Pour éviter d'avoir à manipuler des nombres binaires trop longs, les adresses 32 bits sont divisées en 4 octets. Ce format est appelé la **notation décimale pointée,** cette notation consiste à découper une adresse en quatre blocs de huit bits. Chaque bloc est ensuite converti en un nombre décimal.

Chacun des octets peut être représenté par un nombre de 0 à 255.

Ex : 130.150.0.1

Exemple :

L'adresse IP 10010110110010000000101000000001 est d'abord découpée en quatre blocs :

10010110.11001000.00001010.00000001 puis, chaque bloc est converti en un nombre décimal pour obtenir finalement 150.200.10.1

= >4 nombres entiers (entre 0 et 255) séparés par des points.

= >4 octets

L'écriture avec les points est une convention, le codage en machine est binaire.

2.2.2. Classes d'adresses

La communauté Internet a défini **trois classes d'adresses** appropriées à des réseaux de différentes tailles. Il y a, a priori, peu de réseaux de grande taille (classe A), il y a plus de réseaux de taille moyenne (classe B) et beaucoup de réseaux de petite taille (classe C). La taille du réseau est exprimée en nombre d'hôtes potentiellement connectés.

Le premier octet d'une adresse IP permet de déterminer la classe de cette adresse.

Les adresses disponibles (de 0.0.0.0 à 255.255.255.255) ont donc été découpées en plages réservées à plusieurs catégories de réseaux.

Pour éviter d'avoir recours aux organismes NIC à chaque connexion d'un nouveau poste, chaque société se voit attribuer une plage d'adresse pour son réseau. Le nombre d'adresses disponibles dans chaque plage dépend de la taille du réseau de la société. Les grands réseaux sont dits de classe A (IBM, Xerox , DEC, Hewlett−Packard), les réseaux de taille moyenne sont de classe B (Microsoft en fait partie !), et les autres sont de classe C.

Figure 2−1. Classes d'adresses

Par exemple, l'adresse d'un poste appartenant à un réseau de classe A est donc de la forme :

0AAAAAAA.xxxxxxxx.xxxxxxxx.xxxxxxxx, avec A fixé par le NIC et x quelconque.

Exemple

IBM a obtenu l'adresse 9 (en fait, on devrait dire 9.X.X.X, mais il est plus rapide de n'utiliser que la valeur du premier octet). 9 est bien de classe A car 9d=**0**0001001b

Cela signifie que chaque adresse IP du type 00001001.xxxxxxxx.xxxxxxxx.xxxxxxxx, avec x prenant la valeur 0 ou 1, fait partie du réseau d'IBM.

Malgré ces possibilités d'adressage, la capacité initialement prévue est insuffisante et sera mise à défaut d'ici quelques années. L'**IPNG** (*Internet Protocol Next Generation*) ou Ipv6 devrait permettre de résoudre ces difficultés en utilisant un adressage sur 16 octets noté en héxadécimal.

2.2.3. Identification du réseau

L'adresse IP se décompose, comme vu précédemment, en un numéro de réseau et un numéro de noeud au sein du réseau.

Afin de s'adapter aux différents besoins des utilisateurs, la taille de ces 2 champs peut varier.

On définit ainsi les 5 classes d'adresses notées A à E:

Figure 2−2. Classes d'adresses

ex. : Soit l'adresse IP suivante : 142.62.149.4

142 en décimal = 100011102 en binaire

Le mot binaire commence par les bits 102 donc il s'agit d'une adresse de classe B. Ou, plus simple : 142 est compris entre 128 et 191.

S'agissant d'une adresse de classe B, les deux premiers octets (a et b) identifient le réseau. Le numéro de réseau est donc : 142.62.0.0

Les deux derniers octets (c et d) identifient l'équipement hôte sur le réseau.

Finalement, cette adresse désigne l'équipement numéro 149.4 sur le réseau 142.62.

2.2.4. Adresses réservées

Les adresses réservées ne peuvent désigner une machine TCP/IP sur un réseau.

L'adresse d'acheminement par défaut (route par défaut.) est de type **0.X.X.X**. Tous les paquets destinés à un réseau non connu, seront dirigés vers l'interface désignée par **0.0.0.0**.

NB : 0.0.0.0 est également l'adresse utilisée par une machine pour connaître son adresse IP durant une procédure d'initialisation (DHCP).

L'adresse de bouclage (*loopback*): l'adresse de réseau 127 n'est pas attribuée à une société, elle est utilisée comme adresse de bouclage dans tous les réseaux. Cette adresse sert à tester le fonctionnement de votre carte réseau. Un ping 127.0.0.1 doit retourner un message correct. Le paquet envoyé avec cette adresse revient à l'émetteur.

Toutes les adresses de type 127.X.X.X ne peuvent pas être utilisées pour des hôtes. La valeur de 'x' est indifférente. On utilise généralement **127.0.0.1**

L'adresse de réseau est une adresse dont tous les bits d'hôte sont positionnés à 0 (ex 128.10.0.0 adresse de réseau du réseau 128.10 de classe B). Elle est utilisée pour désigner tous les postes du réseau. On utilise cette adresse dans les tables de routage.

Les noms de réseaux de type :

1. X.Y.Z.0 (de 192.0.0.0 à 223.255.255.0) sont dits de classe C

- X.Y.0.0 (de 128.0.0.0 à 191.255.0.0) sont dits de classe B :
- X.0.0.0. (de 1.0.0.0 à 126.255.255.254) sont dits de classe A :

1. de 224.0.0.0 à 254.0.0.0 : adresses réservées pour des besoins futurs
2.

L'adresse de diffusion est *une adresse dont tous les bits d'hôte sont positionnés à 1 (ex : 128.10.255.255 adresse de diffusion du réseau 128 de classe B).*

Elle est utilisée pour envoyer un message à tous les postes du réseau.

Les adresses "privées"

Les adresses suivantes (RFC 1918) peuvent également être librement utilisées pour **monter un réseau privé** :

A 10.0.0.0

B 172.16.0.0 à 172.31.255.255

C 192.168.0.0 à 192.168.255.255

Aucun paquet provenant de ces réseaux ou à destination de ces réseaux, ne sera routé sur l'Internet.

Figure 2−3. Récapitulatif Classes d'adresses

Le rôle du masque de réseau (netmask) est d'identifier précisément les bits qui concernent le N° de réseau d'une adresse (il "masque" la partie hôte de l'adresse).

Un bit à 1 dans le masque précise que le bit correspondant dans l'adresse IP fait partie du **N° de réseau ;** à l'inverse, **un bit à 0** spécifie un bit utilisé pour coder le **N° d'hôte**.

Ainsi, on a un masque dit "par défaut" qui correspond à la classe de ce réseau.

Exemple: dans un réseau de classe A sans sous−réseau, le premier octet correspond à l'adresse du réseau donc le **netmask** commence par 11111111 suivi de zéros soit **255.0.0.0.**

D'où le tableau suivant :

Classe	*Netmask*
A	*255.0.0.0*
B	*255.255.0.0*
C	*255.255.255.0*

Ex : Si mon adresse IP est 149.127.1.110 alors je travaille avec une adresse de classe B. Mon N° de réseau est 149.127.0.0 et mon masque 255.255.0.0.

2.3. Les sous−réseaux

2.3.1. Pourquoi créer des sous réseaux ?

Les avantages de la segmentation en sous−réseau sont les suivants :

1. **Utilisation de plusieurs media** (câbles, supports physiques). La connexion de tous les noeuds à un seul support de réseau peut s'avérer impossible, difficile ou coûteuse lorsque les noeuds sont trop éloignés les uns des autres ou qu'ils sont déjà connectés à un autre media.
2. **Réduction de l'encombrement**. Le trafic entre les noeuds répartis sur un réseau unique utilise la largeur de bande du réseau. Par conséquent, plus les noeuds sont nombreux, plus la largeur de bande requise est importante. La répartition des noeuds sur des réseaux séparés permet de réduire le nombre de noeuds par réseau. Si les noeuds d'un réseau de petite taille communiquent principalement avec d'autres noeuds du même réseau, l'encombrement global est réduit.
3. **Economise les temps de calcul**. Les diffusions (paquet adressé à tous) sur un réseau obligent chacun des noeuds du réseau à réagir avant de l'accepter ou de la rejeter.
4. **Isolation d'un réseau**. La division d'un grand réseau en plusieurs réseaux de taille inférieure permet de limiter l'impact d'éventuelles défaillances sur le réseau concerné. Il peut s'agir d'une erreur matérielle du réseau (une connexion
5. **Renforcement de la sécurité**. Sur un support de diffusion du réseau comme Ethernet, tous les noeuds ont accès aux paquets envoyés sur ce réseau. Si le trafic sensible n'est autorisé que sur un réseau, les autres hôtes du réseau n'y ont pas accès.
6. **Optimisation de l'espace réservé à une adresse IP**. Si un numéro de réseau de classe A ou B vous est assigné et que vous disposez de plusieurs petits réseaux physiques, vous pouvez répartir l'espace de l'adresse IP en multiples sous−réseaux IP et les assigner à des réseaux physiques spécifiques. Cette méthode permet d'éviter l'utilisation de numéros de réseau IP supplémentaires pour chaque réseau physique.

2.3.2. Masque de sous−réseau

Les masques de sous−réseaux (*subnet mask*) permettent de **segmenter un réseau en plusieurs sous−réseaux**. On utilise alors une partie des bits de l'adresse d'hôte pour identifier des sous−réseaux.

L'adressage de sous−réseau permet de définir des organisations internes de réseaux qui ne sont pas visibles à l'extérieur de l'organisation. Cet adressage permet par exemple l'utilisation d'un routeur externe qui fournit alors une seule connexion Internet.

Toutes les machines appartenant à un sous−réseau possèdent le même numéro de réseau.

On utilise le même principe que pour le masque par défaut sur l'octet de la partie hôte auquel on va prendre des bits. Ainsi, le masque de sous−réseau d'une adresse de classe B commencera toujours par 255.255.xx.xx

Pour connaître l'adresse du sous−réseau auquel une machine appartient, on effectue en réalité un **ET logique** entre l'adresse de la machine et le masque.

Adresse : 200.100.40.33 11001000.01100100.00101000.00100001

Masque : 255.255.255.224 11111111.11111111.11111111.11100000

Opération ET 11001000.01100100.00101000.00100000

=> La machine appartient au sous−réseau : 200.100.40.32

Nous voyons dans ce deuxième exemple que nous avons pris 3 bits sur le dernier octet de notre adresse. Ces 3 bits vont nous permettre de construire plusieurs sous−réseaux.

Ex : adresse : 192.0.0.131

Masque : 255.255.255.192

Conversion de l'adresse en binaire : *11000000 00000000 00000000 10000011*

Conversion du masque en binaire : **11111111 11111111 11111111 11000000**

La machine appartient au sous−réseau *192.0.0.192 et a l'adresse 11=3*

Pour des raisons de commodité, on préférera **réserver un octet entier** pour coder le numéro de sous réseau. De même la théorie ne nous oblige pas à prendre les **bits contigus d'un masque**, même si c'est ce que nous utiliserons en pratique.

Important : pour parer à d'éventuels problèmes de routage et d'adressage, tous les ordinateurs d'un réseau logique doivent utiliser le même masque de sous−réseau et le même identificateur de réseau.

2.3.3. Sous−réseaux

2.3.3.1. Nombre de sous−réseaux

Le nombre théorique de sous−réseaux est égal à 2n, n étant le nombre de bits à 1 du masque, utilisés pour coder les sous−réseaux.

Exemple :

Adresse de réseau : 200.100.40.0

Masque : 255.255.255.224

224 = 11100000 donc **3 bits** pour le N° de **sous−réseau** et 5 bits pour l'hôte.

Le nombre de sous−réseau est donc de *: 23 =8.*

Remarque : la RFC 1860 (remplacée par la RFC 1878) stipulait qu'un numéro de sous réseau ne peut être composé de bits tous positionnés à zéro ou tous positionnés à un.

Autrement dit, dans notre exemple, on ne pouvait pas utiliser le sous−réseau 0 et le sous−réseau 224. Le premier nous donnant une adresse de sous−réseau équivalente à l'adresse du réseau soit 200.100.40.0. Le deuxième nous donnant une adresse de sous−réseau dont l'adresse de diffusion se confondrait avec l'adresse de diffusion du réseau. Le nombre de sous−réseaux aurait alors été de seulement *: 2^3−2 =6.*

Il est donc important de savoir quelle RFC est utilisée par votre matériel pour savoir si les adresses de sous−réseau composées de bits tous positionnés à zéro ou tous positionnés à un sont prises en compte ou non.

2.3.3.2. Adresse des sous−réseaux

Il faut donc maintenant trouver les adresses des sous−réseaux valides en utilisant les bits à 1 du masque.

Pour l'exemple précédent, il faut utiliser les 3 premiers bits:

000 00000 = 0

001 00000 = 32

010 00000 = 64

011 00000 = 96

100 00000 = 128

101 00000 = 160

110 00000 = 192

111 00000 = 224

On constate que le pas entre 2 adresses de sous−réseau est de *32 = 25* correspondant *au nombre théorique d'hôtes par sous−réseau.*

2.3.3.3. Adresse de diffusion d'un sous−réseau

Il faut mettre **tous les bits de la partie hôte à 1**.

Cherchons l'adresse de diffusion des sous réseaux précédents.

- Avec le masque 255.255.255.224

Pour le sous−réseau 200.100.40.**32**

32 = **001** 00000 donc l'adresse de diffusion est *001 11111 = 63.*

L'adresse de diffusion complète est donc *200.100.40.63*

Pour le sous−réseau 200.100.40.**64** l'adresse de diffusion est 200.100.40.*95*

...ETC ...

Avec le masque 255.255.255.129

Pour le sous−réseau 200.100.40.1 l'adresse de diffusion est 200.100.40.127

Pour le sous−réseau 200.100.40.128 l'adresse de diffusion est 200.100.40.254

Pourquoi 254 et pas 255 car avec 255 le dernier bit serait à 1 donc on serait dans le sous−réseau 10000001 , en décimal 129.

2.3.3.4. Nombre de postes d'un sous−réseau

Le nombre de postes est égal à **2n**, **n** étant le nombre de **bits à 0** du masque permettant de coder l'hôte. A ce chiffre il faut enlever 2 numéros réservés :

- tous les bits à zéro qui identifie le sous−réseau lui−même.
- tous les bits à 1 qui est l'adresse de diffusion pour le sous−réseau.

Exemples :

Soit le masque 255.255.255.224

224 = 11100000 donc 3 bits pour le N° de sous−réseau et 5 bits pour l'hôte

le nombre de poste est donc de : 2^5 −2 =30 postes.

De même, avec le masque 255.255.255.129 le nombre de postes sera de *2^6−2 = 62 postes*

2.3.3.5. Adresse de poste sur un sous−réseau

L'adresse de poste sur un sous−réseau subnetté " normalement " ne pose pas de problème, elle est comprise dans la fourchette [adresse de sous−réseau + 1, adresse de diffusion du sous−réseau − 1] soit dans l'exemple précédent :

[200.100.400.33,200.100.40.62] pour le sous−réseau 200.100.40.32

[200.100.400.65,200.100.40.94] pour le sous−réseau 200.100.40.64.

Par exemple, au lieu d'allouer un identificateur de réseau de classe B, dans une entreprise comportant 2000 hôtes, InterNic alloue une plage séquentielle de 8 identificateurs de réseau de classe C. Chaque identificateur de réseau de classe C gère 254 hôtes pour un total de 2 032 identificateurs d'hôte.

Alors que cette technique permet de conserver des identificateurs de réseau de classe B, elle crée un nouveau problème.

En utilisant des techniques de routage conventionnelles, les routeurs d'lnternet doivent désormais comporter huit entrées (en RAM) dans leurs tables de routage pour acheminer les paquets IP vers l'entreprise. La technique appelée CIDR (Classless

Inter−Domain Routing) permet de réduire les huit entrées utilisées dans l'exemple précédent à une seule entrée correspondant à tous les identificateurs de réseau de classe C utilisés par cette entreprise.

Soit les huit identificateurs de réseau de classe C commençant par l'identificateur de réseau 220.78.**168.0** et se terminant par l'identificateur de réseau 220.78.**175.0**, l'entrée de la table de routage des routeurs d'lnternet devient :

Identificateur de réseau	*Masque de sous réseau*	*Masque de sous réseau (en binaire)*
220.78.168.0	*255.255.248.0*	11111111 11111111 *1111*000 00000000

En effet 168 en binaire donne : **10101**000

et 175 donne : **10101**111

la partie commune porte bien sur les 5 1ers bits

d'où le masque : **11111**000

Dans l'adressage de sur−réseaux, la destination d'un paquet est déterminée en faisant un ET logique entre l'adresse IP de destination et le masque de sous−réseau de l'entrée de routage. En cas de correspondance avec l'identificateur de réseau, la route est utilisée. Cette procédure est identique à celle définie pour l'adressage de sous−réseaux.

La notation CIDR définit une convention d'écriture qui spécifie le nombre de bits utilisés pour identifier la partie réseau (les bits à 1 du masque).

Les adresses IP sont alors données sous la forme :

142.12.42.145 / **24** <=> 142.12.42.145 255.255.255.0

153.121.219.14 / **20**<=> 153.121.219.14 255.255.240.0

Dans cette écriture les nombres **24** et **20** représentent le nombre de bits consacrés à la codification du réseau (et sous réseau).

Remarque : Les RFC 1518 et 1519 définissent le CIDR *(Classless Inter−Domain Routing).*

2.4. Le routage

2.4.1. Recherche de l'adresse physique

La communication entre machines ne peut avoir lieu que lorsque celles−ci connaissent leurs adresses physiques (MAC). Pour envoyer les paquets IP vers les autres noeuds du réseau, les noeuds qui utilisent les protocoles TCP/IP **traduisent les adresses IP de destination en adresses MAC**. L'application émettrice ajoute son adresse IP au paquet et l'application réceptrice peut utiliser cette adresse IP pour répondre.

Sur les réseaux à diffusion, tels qu'Ethernet et Token−Ring, le **protocole** IP nommé **ARP** (*Address Resolution Protocol*) fait le lien entre les adresses IP et les adresses physiques (ou MAC).

Quand un poste **cherche l'adresse physique** correspondant à l'adresse IP qu'il connaît, le protocole **ARP** se met en oeuvre et réalise les tâches suivantes :

1. réalisation d'un appel broadcast sur le réseau en demandant à qui correspond l'adresse IP à résoudre : il diffuse un paquet ARP qui contient l'adresse IP du destinataire
2. les machines du réseau comparent l'adresse demandée à leur adresse et le noeud correspondant renvoie son adresse physique au noeud qui a émis la requête.
3. stockage de l'adresse physique lorsque le destinataire répond dans le cache ARP de la machine

Pour accélérer la transmission des paquets et réduire le nombre des requêtes de diffusion qui doivent être examinées par tous les noeuds du réseau, chaque noeud dispose d'un **cache de résolution d'adresse**. Chaque fois que le noeud diffuse une requête ARP et reçoit une réponse, il crée une entrée dans une table de correspondance stockée en mémoire cache. Cette entrée assigne l'adresse IP à l'adresse physique.

Lorsque le noeud envoie un autre paquet IP, il cherche l'adresse IP dans son cache. S'il la trouve, il utilise alors l'adresse physique correspondante pour son paquet.

Le noeud diffuse une requête ARP seulement s'il ne trouve pas l'adresse IP dans son cache.

2.4.2. Principe

Le routage dans Internet est similaire au **mécanisme d'adressage du courrier**.

Si vous adressez une lettre à un destinataire aux USA, à Los Angeles, dans l'état de Californie. Le bureau de poste de Belfort reconnaîtra que cette adresse n'est pas locale et transmettra le courrier au bureau français des PTT qui le remettra au service du mail US. Celui−ci s'en remettra à son bureau de la Californie, qui le transmettra au bureau de Los Angeles, qui connaît la localisation qui correspond à l'adresse dans la ville.

Avantages du système :

1. le bureau de poste local n'a pas à connaître toutes les adresses du monde
2. le chemin suivi peut être variable : chaque opérateur sait juste à qui remettre le courrier.

Le routage dans un réseau est identique :

Internet en entier est composé de réseaux autonomes qui s'occupent en interne de l'adressage entre leurs hôtes. Ainsi, tout datagramme arrivant sur un hôte quelconque du réseau destination sera acheminé à bon port par ce réseau seul.

Quand tous les hôtes participent au même réseau, chacun d'eux peut adresser des paquets aux autres sans difficulté. Par contre, si le destinataire est situé sur un autre réseau, le problème est de savoir où et à qui adresser le paquet puisque l'hôte expéditeur ne « voit » pas le destinataire.

On appelle **passerelle** (dans la terminologie TCP/IP) ou **routeur** un équipement qui fait le lien entre différents réseaux ou entre sous−réseaux. Ex de passerelle: **un ordinateur équipé de plusieurs adaptateurs réseau** peut être relié avec chacune d'elle à un réseau physiquement séparé.

Les paquets d'un réseau qui sont adressés à l'autre réseau doivent passer par la passerelle. D'où la nécessité pour chaque hôte de connaître, sur son réseau, l'adresse IP d'un ou de plusieurs routeurs qui servent de passage vers le ou les réseaux qu'ils ne connaît pas.

Mettre en place le routage consiste à configurer chaque hôte du réseau de façon à ce qu'il sache vers quelle adresse de son propre réseau il doit adresser un paquet qui concerne un autre réseau (ou sous−réseau). Ces destinataires intermédiaires sont des routeurs qui prennent en charge le paquet.

Les hôtes pouvant être nombreux, bien souvent chacun ne connaît que l'adresse d'une passerelle (routeur) par défaut et ce sera cette passerelle qui « connaîtra » les adresses des autres routeurs.

2.4.3. Acheminement des paquets TCP−IP

Voici comment un hôte expéditeur se comporte pour adresser un paquet à un destinataire :

1. Il extrait l'adresse de réseau, voire de sous réseau de l'adresse du destinataire et la compare à sa propre adresse de réseau ou de sous réseau. S'il s'agit du même réseau, le paquet est expédié directement au destinataire en mettant en oeuvre ARP.
2. S'il ne s'agit pas du même réseau, l'expéditeur cherche dans sa table de routage une correspondance destinataire final / destinataire intermédiaire (routeur). Il cherche, en quelque sorte, sur son réseau, un hôte capable de servir de facteur vers un autre réseau.
3. L'expéditeur cherche d'abord à trouver dans sa table de routage locale l'adresse IP complète du destinataire,
4. s'il ne la trouve pas il cherche l'adresse du sous réseau du destinataire,
5. s'il ne la trouve pas, il cherche enfin l'adresse du réseau,
6. s'il ne trouve aucune correspondance, l'expéditeur cherche dans sa table l'adresse d'une passerelle à utiliser par défaut, (route 0.0.0.0)
7. s'il échoue là encore, le paquet, décidément bien encombrant, est supprimé.
8. Si l'une de ces recherches aboutit, la machine émettrice construit le paquet avec l'adresse IP du destinataire hors réseau. Elle l'encapsule dans une trame ayant comme adresse MAC de destination l'adresse MAC du routeur. La couche 2 du routeur lit la trame qui lui est adressée et la transmet à la couche 3 IP. Celle−ci récupère le paquet et s'aperçoit que le paquet ne lui est pas adressé, elle consulte sa table de routage, décide sur quelle nouvelle interface réseau le paquet doit être transmis, encapsule le paquet dans une nouvelle trame, et ainsi de suite de passerelle en passerelle jusqu'à destination.

2.4.4. Les tables de routage

Les réseaux IP sont interconnectés par des routeurs IP de niveau 3 (appelés abusivement en terminologie IP des gateways ou passerelles).

Chaque station IP doit connaître le routeur par lequel il faut sortir pour pouvoir atteindre un réseau extérieur, c'est−à−dire avoir en mémoire une table des réseaux et des routeurs. Pour cela elle contient une table de routage locale.

Dans une configuration de **routage statique**, une table de correspondance entre adresses de destination et adresses de routeurs intermédiaires est complétée « à la main » par l'administrateur, on parle de **table de routage**.

Figure 2−4. table de routage

Réseau 1 −−> Routeur 1

Réseau 2 −−> Routeur 1

.....

Réseau n −−> Routeur p

La table de routage comporte les adresses des passerelles permettant d'atteindre les réseaux de destination. La commande **Route** permet de manipuler le contenu de la table de routage.

Exemple de table de routage :

Destination	*Masque de Sous réseau*	*Passerelle*	
127.0.0.1	255.255.255.0	127.0.0.1	voie de bouclage
142.62.10.0	255.255.255.0	142.62.10.99	sortie de la passerelle vers le sous−réseau 10
142.62.20.0	255.255.255.0	142.62.20.99	sortie de la passerelle vers le sous−réseau 20

2.4.5. Acheminement Internet

2.4.5.1. Domaine d'acheminement

Les échanges entre passerelles de chaque domaine de routage font l'objet de protocoles particuliers : EGP (Exterior Gateway Protocol) et BGP (Border Gateway Protocol) plus récent. Ces protocoles envoient les paquets vers des destinations en dehors du réseau local vers des réseaux externes (Internet, Extranet...).

2.4.5.2. Principe du choix d'une voie d'acheminement

1. Si l'hôte de destination se trouve sur le réseau local, les données sont transmises à l'hôte destination
2. Si l'hôte destination se trouve sur un réseau à distance, les données sont expédiées vers une passerelle locale qui route le paquet vers une autre passerelle et ainsi de suite de passerelle en passerelle jusqu'à destination.

La commande **Tracert** permet de suivre à la trace le passage de routeur en routeur pour atteindre un hôte sur le réseau. La commande **Ping** permet de vérifier la fiabilité d'une route donnée.

2.4.6. Routage dynamique

Les **protocoles d'échange dynamique des tables de routage IP** sur un réseau local sont **RIP** (*Routing Information Protocol*) et le protocole **OSPF (Open Shortest Path First).** Dans une configuration de **routage dynamique**, un protocole (RIP ou OSPF) est mis en oeuvre pour construire dynamiquement les chemins entre routeurs.

Le protocole RIP permet à un routeur d'échanger des informations de routage avec les routeurs avoisinants. Dès qu'un routeur est informé d'une modification quelconque de la configuration sur les réseaux (telle que l'arrêt d'un routeur), il transmet ces informations aux routeurs avoisinants. Les routeurs envoient également des paquets de diffusion générale RIP périodiques contenant toutes les informations de routage dont ils disposent. Ces diffusions générales assurent la synchronisation entre tous les routeurs.

Avec un protocole comme RIP, on peut considérer que les tables de routages des routeurs et passerelles sont constituées et mises à jour automatiquement.

Chapitre 3. Eléments de cours sur ARP

Résumé sur ARP

3.1. Le protocole ARP

L'adresse Ethernet est une adresse unique sur 48 bits (6 octets) associée à la carte Ethernet. Lorsqu'un noeud `N1` du réseau TCP/IP X1.X1.X1.X1 veut émettre un paquet TCP/IP (dans une trame Ethernet) vers une machine `N2` d'adresse IP (X2.X2.X2.X2), il faut qu'il connaisse l'adresse Ethernet (E2.E2.E2.E2.E2.E2). Pour réaliser l'association *@ip / @ Ethernet* l'émetteur `N1` utilise le protocole ARP dont le principe est le suivant :

L'émetteur envoie une trame Ethernet de diffusion (broadcast) (ie @destinataire toute à 1) contenant un message ARP demandant

qui est X2.X2.X2.X2 ?

Figure 3−1. Trame Ethernet contenant une requête ARP

Toutes les machines IP du réseau local reçoivent la requête. `N2` qui a l'adresse X2.X2.X2.X2 se reconnaît, et elle répond à `N1` ie X1.X1.X1.X1 (dans une trame destinée à E1.E1.E1.E1.E1.E1)

Figure 3−2. Trame Ethernet contenant une réponse ARP

Chaque machine maintient en mémoire une table cachée de correspondances *@ip / @ Ethernet* pour éviter trop de requêtes ARP. Chaque entrée de la table à une durée de vie limitée. Voici pour exemple ce que donne le programme tcpdump avec la commande **ping *`192.168.1.2`*** à partir de la machine `uranus` alors que la table ARP de l'hôte `uranus` est vide :

```
13:17:14.490500 arp who−has 192.168.1.2 tell uranus.planete.net
13:17:14.490500 arp reply 192.168.1.2 is−at 0:40:33:2d:b5:dd
13:17:14.490500 uranus.planete.net > 192.168.1.2: icmp: echo request
13:17:14.490500 192.168.1.2 > uranus.planete.net: icmp: echo reply
13:17:15.500500 uranus.planete.net > 192.168.1.2: icmp: echo request
13:17:15.500500 192.168.1.2 > uranus.planete.net: icmp: echo reply
```

Explications :

Ligne 1, `uranus` demande qui est 192.168.1.2 (requête ARP) Le paquet est diffusé à tous les hôtes du réseau.

Ligne 2 réponse ARP : je suis à l'adresse Ethernet 00:40:33:2d:b5:dd

Lignes 3 à 6 : échanges de paquets ICMP entre les 2 hôtes.

Chapitre 4. L'adressage IP v6

> L'adressage IPv4 sur 32 bits se révélant insuffisant (saturation prévue pour 2010) avec le développement d'Internet, l'IETF en 1998 a proposé le standard IPv6 (ou Ipng − ng pour "Next Generation", RFC 2460), afin de permettre l'adressage d'au moins un milliard de réseaux, soit quatre fois plus qu'IPv4.

IPv6 possède un nouveau format d'en−tête IP, une infrastructure de routage plus efficace, et un espace d'adressage plus important. Pour permettre le déploiement d'IPv6 de la manière la plus flexible possible, la compatibilité avec IPv4 est garantie.

4.1. Caractéristiques

− les **adresses IPv6 sont codées sur 128 bits** (1 milliard de réseaux).

− le principe des numéros de réseaux et des numéros d'hôtes est maintenu.

− IPv6 est conçu pour interopérer avec les systèmes IPv4 (transition douce prévue sur 20 ans). L'adresse IPv6 peut contenir une adresse IPv4 : on place les 32 bits de IPv4 dans les bits de poids faibles et on ajoute un préfixe de 96 bits (80 bits à 0 suivis de 16 bits à 0 ou 1)

− IPv6 utilise un **adressage hiérarchique** (identification des différents réseaux de chaque niveau) ce qui permet un routage plus efficace.

− IPv6 prévu pour les systèmes mobiles : auto−configuration, notion de voisinage (neighbor).

− IPv6 permet **l'authentification et le chiffrement** dans l'en−tête des paquets, ce qui permet de sécuriser les échanges. En effet IP v.6 intègre **IPSec** (protocole de création de tunnel IP avec chiffrement), qui garantit un contexte sécurisé par défaut.

− IPv6 intègre la **qualité de service** : introduction de flux étiquetés (avec des priorités)

− IPv6 prend mieux en charge le trafic en temps réel (garantie sur le délai maximal de transmission de datagrammes sur le réseau).

4.2. Types d'adresses

IPv6 supporte 3 types d'adresses: Unicast, Anycast et Multicast.

Anycast est un nouveau type d'adressage. Il identifie qu'un noeud, parmi un groupe de noeuds, doit recevoir l'information. L'interface de destination doit spécifiquement être configurée pour savoir qu'elle est Anycast.

La notion de diffusion (broadcast) disparaît dans IPv6.

4.3. Représentation des adresses

Une adresse IPv6 s'exprime en notation hexadécimale avec le séparateur "deux−points".

Exemple d'adresse :

5800:10C3:E3C3:F1AA:48E3:D923:D494:AAFF

Dans IPv6 les masques sont exprimés en notation CIDR.

Il y a 3 façons de représenter les adresses IPv6

forme préférée :

"x:x:x:x:x:x:x:x" où x représente les valeurs hexadécimales des 8 portions de 16 bits de l'adresse. Exemple:

3ffe:0104:0103:00a0:0a00:20ff:fe0a:3ff7

forme abrégée :

Un groupe de plusieurs champs de 16 bits mis à 0 peut être remplacé par la notation "::".

La séquence "::" ne peut apparaître qu'une seule fois dans une adresse.

Exemple:

5f06:b500:89c2:a100::800:200a:3ff7

ff80::800:200a:3ff7

::1

forme mixte :

Lorsqu'on est dans un environnement IPv4 et IPv6, il est possible d'utiliser une représentation textuelle de la forme: "**x:x:x:x:x:x:d.d.d.d**", où les 'x' sont les valeurs hexadécimales des 6 premiers champs de 16 bits et les 'd' sont les valeurs décimales des 4 derniers champs de 8 bits de l'adresse.

Exemple:

::137.194.168.93

4.4. Allocation de l'espace d'adressage

Le type d'adresse IPv6 est indiqué par les premiers bits de l'adresse qui sont appelés le "Préfixe de Format" (Format Prefix). L'allocation initiale de ces préfixes est la suivante:

Allocation	*Préfixe*	*Usage*
Adresses Unicast pour ISP	010	Adresse d'un hôte sur Internet
Adresses Unicast expérimentales	001	
Adresses "Link Local Use"	1111 1110 10	Un seul réseau, autoconfiguration, « neighbor »
Adresses "Site Local Use"	1111 1110 11	sous−réseaux privés
Adresses Multicast	1111 1111	

15 % de l'espace d'adressage est actuellement alloué. Les 85% restants sont réservés pour des usages futurs. En réalité sur les 128 bits, seulement 64 sont utilisés pour les hôtes (Interface ID).

Chapitre 5. Fichiers de configuration du réseau et commandes de base

Présentation des principaux fichiers de configuration du réseau et des commandes d'administration système et réseau.

5.1. Présentation du document : les outils de l'administrateur réseau

Ce document présente les principaux fichiers de configuration d'une machine en réseau, et les commandes d'administration réseau.

Il est composé de 6 parties:

1. Les fichiers de configuration réseau
2. La commande **ifconfig**
3. La commande **arp**
4. La commande **route**
5. La commande **netstat**
6. La commande **traceroute**

5.2. Les fichiers de configuration

5.2.1. Le fichier `/etc/hosts`

Le fichier `hosts` donne un moyen d'assurer la résolution de noms

Exemple de fichier `host`

```
127.0.0.1  localhost localhost.localdomain
192.168.1.1  uranus.foo.org uranus
```

5.2.2. Le fichier `/etc/networks`

Il permet d'affecter un nom logique à un réseau

```
localnet   127.0.0.0
foo-net 192.168.1.0
```

Cette option permet par exemple d'adresser un réseau sur son nom, plutôt que sur son adresse.

route add *`foo-net`* au lieu de **route add -net *`192.168.1.0`***.

5.2.3. Le fichier `/etc/host.conf`

Il donne l'ordre dans lequel le processus de résolution de noms est effectué. Voici un exemple de ce que l'on peut trouver dans ce fichier :

```
order hosts,bind
```

La résolution est effectuée d'abord avec le fichier `host`, en cas d'échec avec le DNS.

5.2.4. Le fichier `/etc/resolv.conf`

Il permet d'affecter les serveurs de noms.

Exemple

```
Nameserver 192.168.1.1
Nameserver 192.168.1.2
Nameserver 192.168.1.3
```

Ici le fichier déclare le nom de domaine et 3 machines chargées de la résolution de noms.

5.2.5. Les fichiers de configuration des interfaces réseau

Vous trouverez ces fichiers dans `/etc/network/interfaces`. Voici un exemple qui contient 3 interfaces.

```
# /etc/network/interfaces -- configuration file for ifup(8), ifdown(8)
# The loopback interface
# automatically added when upgrading

auto lo eth0 eth1

iface lo inet loopback

iface eth0 inet static
        address 192.168.90.1
        netmask 255.255.255.0
        network 192.168.90.0
        broadcast 192.168.90.255
        gateway 192.168.90.1
```

```
iface eth1 inet static
        address 192.168.0.1
        netmask 255.255.255.0
        network 192.168.0.0
        broadcast 192.168.0.255
```

5.3. Les outils de l'administrateur réseau

5.3.1. La commande ifconfig

La commande **ifconfig** permet la configuration locale ou à distance des interfaces réseau de tous types d'équipements (unité centrale, switch, routeur). La ligne de commande est :

`ifconfig` *`interface adresse`* `[parametres]`.

Exemple : **`ifconfig` *`eth0 192.168.1.2`*** (affecte l'adresse 192.168.1.2 à la première interface physique.

Voici les principaux arguments utilisés :

`interface` logique ou physique, il est obligatoire,

`up` active l'interface

`down` désactive l'interface

`mtu` définit l'unité de transfert des paquets

`netmask` affecter un masque de sous−réseau

`broadcast` définit l'adresse de broadcast

`arp` ou `−arp` activer ou désactiver l'utilisation du cache arp de l'interface

`metric` paramètre utilisé pour l'établissement des routes dynamiques, et déterminer le << coût >> (nombre de sauts ou << hops >>) d'un chemin par le protocole RIP.

`multicast` active ou non la communication avec des machines qui sont hors du réseau.

`promisc` ou `−promisc` activer ou désactiver le mode promiscuité de l'interface. En mode *promiscuous*, tous les paquets qui transitent sur le réseau sont reçus également par l'interface. Cela permet de mettre en place un analyseur de trame ou de protocole.

*Description du résultat de la commande **ifconfig `eth0`** :*

1. eth0 Link encap:Ethernet HWaddr 00:80:C8:32:C8:1E
2. inet addr:192.168.1.1 Bcast:192.168.1.255 Mask:255.255.255.0
3. UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
4. RX packets:864 errors:0 dropped:0 overruns:0 frame:0
5. TX packets:654 errors:0 dropped:0 overruns:0 carrier:0
6. collisions:0
7. Interrupt:10 Base address:0x6100

Explications :

Ligne 1: l'interface est de type Ethernet. La commande nous donne l'adresse MAC de l'interface.

Ligne 2 : on a l'adresse IP celle de broadcast, celle du masque de sous−réseau

Ligne 3 : l'interface est active (UP), les modes broadcast et multicast le sont également, le MTU est de 1500 octets, le Metric de 1

Ligne 4 et 5 : RX (paquets reçus), TX (transmis), erreurs, suppressions, engorgements, collision

Mode d'utilisation :

Ce paragraphe décrit une suite de manipulation de la commande **ifconfig**.

Ouvrez une session en mode console sur une machine.

1 − Relevez les paramètres de votre machine à l'aide de la commande **ifconfig**. Si votre machine n'a qu'une interface physique, vous devriez avoir quelque chose d'équivalent à cela.

```
Lo  Link encap:Local Loopback
inet addr:127.0.0.1  Bcast:127.255.255.255  Mask:255.0.0.0
UP BROADCAST LOOPBACK RUNNING  MTU:3584  Metric:1
RX packets:146 errors:0 dropped:0 overruns:0 frame:0
TX packets:146 errors:0 dropped:0 overruns:0 carrier:0
collisions:0

eth0 Link encap:Ethernet  HWaddr 00:80:C8:32:C8:1E
inet addr:192.168.1.1  Bcast:192.168.1.255  Mask:255.255.255.0
UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
RX packets:864 errors:0 dropped:0 overruns:0 frame:0
TX packets:654 errors:0 dropped:0 overruns:0 carrier:0
collisions:0

Interrupt:10 Base address:0x6100
```

2 − Désactivez les 2 interfaces lo et eth0

ifconfig *lo* down

ifconfig *eth0* down

3 − Taper les commandes suivantes :

ping *localhost*

ping *192.168.1.1*

telnet *localhost*

Aucune commande ne fonctionne, car même si la configuration IP est correcte, les interfaces sont désactivées.

4 − Activez l'interface de loopback et tapez les commandes suivantes :

ifconfig *lo* up /* activation de l'interface de loopback */

ping *localhost* ou telnet localhost /* ça ne marche toujours pas */

route add *127.0.0.1* /* on ajoute une route sur l'interface de loopback */

ping *localhost* ou **telnet *localhost*** /* maintenant ça marche */

ping *192.168.1.1* /* ça ne marche pas car il manque encore une route*/

On peut déduire que :

- − pour chaque interface il faudra indiquer une route au protocole.
- − dans la configuration actuelle, aucun paquet ne va jusqu'à la carte, donc ne sort sur le réseau.

Voici le rôle de l'interface loopback. Elle permet de tester un programme utilisant le protocole IP sans envoyer de paquets sur le réseau. Si vous voulez écrire une application réseau, (telnet, ftp, ou autre), vous pouvez la tester de cette façon.

5 − Activez l'interface eth0 et tapez les commandes suivantes :

ifconfig *eth0* up /* activation de l'interface */

route add *192.168.1.1*

ifconfig /* l'information Tx/Rx de l'interface eth0 vaut 0 */

/* Aucun paquet n'est encore passé par la carte.*/

ping *127.0.0.1*

ifconfig /* on voit que l'information Tx/Rx de lo est modifiée */

/* pas celle de eth0, on en déduit que les paquets */

/* à destination de lo ne descendent pas jusqu'à l'interface physique */

ping *192.168.1.1* /* test d'une adresse locale */

ifconfig /* Ici on peut faire la même remarque. Les paquets ICMP */

/* sur une interface locale, ne sortent pas sur le réseau */

/* mais ceux de l'interface lo sont modifiés*/

ping *192.168.1.2* /* test d'une adresse distante */

ifconfig /* Ici les paquets sont bien sortis. Les registres TX/RX de eth0 */

/* sont modifiés, mais pas ceux de lo */

6 −Réalisez les manipulations suivantes, nous allons voir le comportement de la commande **ping** sur les interfaces.

Sur la machine tapez la commande

192.168.1.1 ifconfig /* relevez les valeurs des registres TX/RX */

192.168.1.2 ping 192.168.1.1

192.168.1.1 ifconfig /* relevez les nouvelles valeurs des registres TX/RX */

/* il y a bien eu échange Réception et envoi de paquets*/

192.168.1.2 ping 192.168.1.3

192.168.1.1 ifconfig /* On voit que le registre Rx est modifié mais */

/* le registre Tx n'est pas modifié. La machine a bien reçu*/

/* paquet mais n'a rien renvoyé */

192.168.1.2 ping 192.168.1.2

192.168.1.2 ifconfig /* aucun registre n'est modifié, donc les paquets */

/* ne circulent pas jusqu'à l'interface physique avec un .*/

/* ping sur l'interface locale */

7 − le MTU (*Message Transfert Unit*) détermine l'unité de transfert des paquets.

Vous allez, sur la machine 192.168.1.1 modifier le MTU par défaut à 1500, pour le mettre à 300, avec la commande :

ifconfig *eth0 mtu 300*

Sur la machine d'adresse 192.168.1.2, vous allez ouvrir une session ftp et chronométrer le temps de transfert d'un fichier de 30 MO. Relevez le temps et le nombre de paquets transmis ou reçus (commande **ifconfig**, flags TX/RX).

Restaurez le paramètre par défaut sur la première machine.

Refaites le même transfert et comparez les chiffres. La différence n'est pas énorme sur le temps car le volume de données est peu important. Par contre la différence sur le nombre de paquets, elle, est importante.

5.3.2. La commande arp

Description de la commande

La commande **arp** permet de visualiser ou modifier la table du cache de l'interface. Cette table peut être statique et (ou) dynamique. Elle donne la correspondance entre une adresse IP et une adresse Ethernet.

À chaque nouvelle requête, le cache ARP de l'interface est mis à jour. Il y a un nouvel enregistrement. Cet enregistrement à une durée de vie (ttl ou *Time To Leave*).

Voici un exemple de cache ARP obtenu avec la commande **arp −va** :

```
? (192.168.1.2) at 00:40:33:2D:B5:DD [ether] on eth0
>Entries: 1      Skipped: 0      Found: 1
```

On voit l'adresse IP et l'adresse MAC correspondante. Il n'y a qu'une entrée dans la table. Voici les principales options de la commande **arp** :

arp −s (ajouter une entrée statique), exemple : **arp −s *192.168.1.2 00:40:33:2D:B5:DD***

arp −d (supprimer une entrée), exemple : **arp −d *192.168.1.2***

Voir la page **man** pour les autres options.

La table ARP et le fonctionnement d'un proxy ARP.

Cela est réalisé par la configuration de tables ARP statiques.

Le proxy est une machine qui est en interface entre un réseau et l'accès à Internet. Il fait office de passerelle et de cache à la fois.

- Passerelle, parce que tous les accès à Internet passent par le Proxy,
- Cache, parce que le Proxy conserve en mémoire cache (sur disque), une copie des pages consultées par les utilisateurs du réseau. Cela évite de télécharger à nouveau la même page sur le site d'origine, si un utilisateur revient fréquemment dessus.

Si un hôte du réseau demande l'adresse d'un noeud distant situé sur un autre réseau, et que cet hôte passe par un proxy, le proxy va renvoyer à l'hôte sa propre adresse Ethernet. Une fois cette opération réalisée, tous les paquets envoyés par l'hôte seront à destination de l'adresse Ethernet du proxy. Le proxy aura en charge de transmettre ces paquets à l'adresse effective du noeud distant.

Pour les réponses, un processus identique est mis en place. Le site consulté, ne retourne les réponses qu'au serveur proxy. Le serveur proxy se charge de ventiler les pages au bon destinataire du réseau local.

Voir, pour le fonctionnement des serveurs cache et la configuration des navigateurs avec ce type de serveur, le document sur le W3 et les scripts CGI.

Mode d'utilisation :

Attention à certaines interprétations de ce paragraphe. Il dépend de votre configuration. Soit vous êtes en réseau local avec une plage d'adresse déclarée, soit vous utilisez une carte d'accès distant.

Première partie :

1. Affichez le contenu de la table ARP avec la commande **`arp -a`**,
2. Supprimez chaque ligne avec la commande **`arp -d @ip`**, où *`@ip`* est l'adresse IP de chaque hôte apparaissant dans la table,
3. La commande **`arp -a`** ne devrait plus afficher de ligne,
4. Faites un **ping**, sur une station du réseau local,
5. **`arp -a`**, affiche la nouvelle entrée de la table,
6. Ouvrez une session sur Internet, puis ouvrez une session ftp anonyme sur un serveur distant en utilisant le nom, par exemple `ftp.cdrom.com`. Utilisez une adresse que vous n'avez jamais utilisée, supprimez également tout gestionnaire de cache.
7. Affichez le nouveau contenu de la table avec **`arp -a`**. Le cache ARP ne contient pas l'adresse Ethernet du site distant, mais celle de la passerelle par défaut. Cela signifie que le client n'a pas à connaître les adresses Ethernet des hôtes étrangers au réseau local, mais uniquement l'adresse de la passerelle. Les paquets sont ensuite pris en charge par les routeurs.
8. Refaites une tentative sur le site choisi précédemment. Le temps d'ouverture de session est normalement plus court. Cela est justifié, car les serveurs de noms ont maintenant dans leur cache la correspondance entre le nom et l'adresse IP.

Deuxième partie :

La commande **arp** permet de diagnostiquer un dysfonctionnement quand une machine prend l'adresse IP d'une autre machine.

1. Sur la machine 192.168.1.1, faites un **ping** sur 2 hôtes du réseau 192.168.1.2 et 192.168.1.3,
2. À l'aide de la commande **arp**, relevez les adresses MAC de ces noeuds,
3. Modifiez l'adresse IP de la machine 192.168.1.2 en 192.168.1.3
4. relancez les 2 machines en vous arrangeant pour que la machine dont vous avez modifié l'adresse ait redémarré la première,
5. Sur la machine d'adresse 192.168.1.1, remettez à jour les tables ARP.
6. Quel est le contenu, après cela de la table ARP ?

Conclusion : vous allez avoir un conflit d'adresses. Vous allez pouvoir le détecter avec la commande **arp**. Autre problème, si vous faites un **telnet** sur 192.168.1.3, il y a de fortes chances pour que ce soit la machine qui était d'adresse 192.168.1.2, qui vous ouvre la session. Nous sommes (par une action volontaire bien sûr) arrivés à mettre la pagaille sur un réseau de 3 postes. Cette pagaille pourrait tourner vite au chaos sur un grand réseau, d'où la nécessité pour un administrateur de faire preuve d'une grande rigueur.

Où en suis−je ?

Exercice 1 :

Vous êtes sur un réseau d'adresse 192.168.1.0 avec une interface d'adresse MAC 00:40:33:2D:B5:DD,

Vous n'avez aucun fichier host sur votre machine,

Il n'y a pas de DNS

La passerelle par défaut est 192.168.1.9

Vous faites un **`ping 195.6.2.3`** qui a une interface d'adresse MAC 00:45:2D:33:C2 est localisée sur Internet

Le réseau fonctionne parfaitement et tout est parfaitement configuré

Cochez la bonne réponse:

A − On a dans la table arp ? (192.168.1.2) at 00:40:33:2D:B5:DD [ether] on eth0

B − On a dans la table arp ? (192.168.1.2) at 00:45:2D:33:C2 [ether] on eth0

C − On a dans la table arp ? (195.6.2.3) at 00:40:33:2D:B5:DD [ether] on eth0

D − On a dans la table arp ? (195.6.2.3) at 00: 00:45:2D:33:C2 [ether] on eth0

E − Il faut un fichier `host`, ou DNS pour réaliser l'opération **ping** demandée

F − Il n'est pas possible dans la configuration actuelle d'atteindre l'hôte 195.6.2.3

Réponse F, car la plage d'adresse 192.168.1.1 à 192.168.1.254 n'est pas routée sur l'Internet, sinon vous auriez l'adresse de la passerelle par défaut dans le cache ARP.

Exercice 2 :

Vous êtes sur un réseau d'adresse 192.5.1.0 avec une interface d'adresse MAC 00:40:33:2D:B5:DD,

Vous n'avez aucun fichier host sur votre machine,

Il n'y a pas de DNS,

La passerelle par défaut est 192.5.1.9

Vous faites un **`ping www.existe.org`** dont l'adresse IP est 195.6.2.3, et qui a une interface d'adresse MAC 00:45:2D:33:C2

Le réseau fonctionne parfaitement et tout est parfaitement configuré

Cochez la bonne réponse:

A − On a dans la table arp ? (192.5.1.0) at 00:40:33:2D:B5:DD [ether] on eth0

B − On a dans la table arp ? (192.5.1.0) at 00:45:2D:33:C2 [ether] on eth0

C − On a dans la table arp ? (195.6.2.3) at 00:40:33:2D:B5:DD [ether] on eth0

D − On a dans la table arp ? (195.6.2.3) at 00: 00:45:2D:33:C2 [ether] on eth0

E − Il faut un fichier `host`, ou DNS pour réaliser l'opération **ping** demandée

F − Il n'est pas possible dans la configuration actuelle d'atteindre l'hôte 195.6.2.3

Réponse E, car la résolution de noms ne peut être effectuée

Exercice 3 :

Vous êtes sur un réseau d'adresse 192.5.1.0, sur une machine d'adresse 192.5.1.1, et une interface d'adresse MAC 00:40:33:2D:B5:DD,

Vous n'avez aucun fichier `host` sur votre machine,

Il n'y a pas de DNS

La passerelle par défaut est 192.5.1.9, d'adresse MAC 09:44:3C:DA:3C:04

Vous faites un **`ping 195.6.2.3`**, et qui a une interface d'adresse MAC 00:45:2D:33:C2

Le réseau fonctionne parfaitement et tout est parfaitement configuré

Cochez la bonne réponse:

A − On a dans la table arp ? (192.5.1.0) at 00:40:33:2D:B5:DD [ether] on eth0

B − On a dans la table arp ? (192.5.1.0) at 00:45:2D:33:C2 [ether] on eth0

C − On a dans la table arp ? (195.6.2.3) at 00:40:33:2D:B5:DD [ether] on eth0

D − On a dans la table arp ? (192.5.1.9) at 09:44:3C:DA:3C:04 [ether] on eth0

E − Il faut un fichier `host`, ou DNS pour réaliser l'opération **ping** demandée

F − Il n'est pas possible dans la configuration actuelle d'atteindre l'hôte 195.6.2.3

Réponse D, l'hôte a bien été trouvé, la table ARP a été mise à jour avec l'adresse IP de la passerelle par défaut et son adresse Ethernet.

5.3.3. La commande route

La commande **route** a déjà été entrevue un peu plus haut, avec la commande **ifconfig**. Le routage définit le chemin emprunté par les paquets entre son point de départ et son point d'arrivée. Cette commande permet également la configuration de pc, de switchs de routeurs.

Il existe 2 types de routages :

− le routage statique

− le routage dynamique.

Le routage statique consiste à imposer aux paquets la route à suivre.

Le routage dynamique met en oeuvre des algorithmes, qui permettent aux routeurs d'ajuster les tables de routage en fonction de leur connaissance de la topologie du réseau. Cette actualisation est réalisée par la réception des messages reçus des noeuds (routeurs) adjacents.

Le routage dynamique permet d'avoir des routes toujours optimisées, en fonction de l'état du réseau (nouveaux routeurs, engorgements, pannes)

On combine en général le routage statique sur les réseaux locaux au routage dynamique sur les réseaux importants ou étendus.

Un administrateur qui dispose par exemple de 2 routeurs sur un réseau, peut équilibrer la charge en répartissant un partie du flux sur un port avec une route, et une autre partie sur le deuxième routeur.

Exemple de table de routage :

```
Kernel IP routing table
Destination     Gateway         Genmask         Flags  Metric  Ref Use  Iface
192.168.1.0     *               255.255.255.    0U     0       0   2    eth0
127.0.0.0       *               255.0.0.0       U      0       0   2    lo
default         192.168.1.9     0.0.0.0         UG     0       0   10   eth0
```

Commentaire généraux :

Destination : adresse de destination de la route

Gateway : adresse IP de la passerelle pour atteindre la route, * sinon

Genmask : masque à utiliser.

Flags : indicateur d'état (U − Up, H − Host − G − Gateway, D − Dynamic, M − Modified)

Metric : coût métrique de la route (0 par défaut)

Ref : nombre de routes qui dépendent de celle−ci

Use : nombre d'utilisation dans la table de routage

Iface : interface eth0, eth1, lo

Commentaire sur la 3ème ligne :

Cette ligne signifie que pour atteindre tous les réseaux inconnus, la route par défaut porte l'adresse 192.168.1.9. C'est la passerelle par défaut, d'où le sigle UG, G pour gateway.

Ajout ou suppression d'une route :

route add [net | host] addr [gw passerelle] [métric coût] [netmask masque] [dev interface]

− *net ou host* indique l'adresse de réseau ou de l'hôte pour lequel on établit une route,

− adresse de destination,

− adresse de la passerelle,

− valeur métrique de la route,

− masque de la route à ajouter,

− interface réseau à qui on associe la route.

Exemples :

route add *127.0.0.1 lo* /* ajoute une route pour l'adresse 127.0.0.1 sur l'interface lo */

route add −net *192.168.2.0 eth0* /* ajoute une route pour le réseau 192.168.2.0 sur l'interface eth0 */

route add *saturne.foo.org* /* ajoute une route pour la machine machin sur l'interface eth0 */

route add default gw *ariane* /* ajoute ariane comme route par défaut pour la machine locale */

/* ariane est le nom d'hôte d'un routeur ou d'une passerelle */

/* gw est un mot réservé */

route add *duschmoll* netmask *255.255.255.192*

/* Encore un qui a créé des sous réseaux., Il s'agit ici d'une classe c */

/* avec 2 sous réseaux, il faut indiquer le masque. */

Suppression d'une route :

route del −net *192.168.1.0*

route del −net *toutbet−net*

> Attention, si on utilise des noms de réseau ou des noms d'hôtes, il faut qu'à ces noms soient associés les adresses de réseau ou des adresses IP dans le fichier `/etc/networks` pour les réseaux, et `/etc/hosts` ou DNS pour les noms d'hôtes.

Vous pouvez également voir l'atelier sur la mise en place d'un routeur logiciel.

Petite étude de cas :

Première partie − réalisation d'une maquette

On dispose de 2 réseaux (A et B) reliés par une passerelle. Le réseau A est également relié à Internet par un routeur. Le réseau A dispose d'un serveur de noms. Chaque réseau a deux machines.

```
Réseau  Nom du réseau   Machine         Nom des machines
A       metaux−net      192.3.2.2       platine
                        192.3.2.3       uranium
                        192.3.2.4       mercure (serveur de noms)
B       roches−net      130.2.0.2       quartz
                        130.2.0.3       silex
```

La passerelle entre le réseau A et B à 2 interfaces :

− eth0 192.3.2.1

− eth1 130.2.0.1

Le réseau A, a une passerelle par défaut pour Internet 130.2.0.9, qui est l'interface d'un autre routeur.

On veut :

− que les stations de chaque réseau puissent accéder à Internet,

− que les stations de chaque réseau puissent communiquer entre−elles,

− que les stations du réseau B, utilisent le serveur de noms le moins possible.

On demande :

1 − d'expliquer comment seront configurés les postes du réseau B,

2 − de donner la configuration des fichiers suivants pour chaque machine (`hosts`, `resolv.conf`, fichier de configuration de carte).

3 − de donner la liste des routes à mettre :

− sur les postes du réseau B,

− sur les postes du réseau A,

− sur la passerelle qui relie les 2 réseaux,

− sur le routeur du réseau A.

5.3.4. La commande netstat

La commande **netstat**, permet de tester la configuration du réseau, visualiser l'état des connexions, établir des statistiques, notamment pour surveiller les serveurs.

Liste des paramètres utilisables avec **netstat** :

Sans argument, donne l'état des connexions,

`-a` afficher toutes les informations sur l'état des connexions,

`-i` affichage des statistiques,

`-c` rafraîchissement périodique de l'état du réseau,

`-n` affichage des informations en mode numérique sur l'état des connexions,

`-r` affichage des tables de routage,

`-t` informations sur les sockets TCP

`-u` informations sur les sockets UDP.

État des connexions réseau avec **netstat**, dont voici un exemple :

```
 Proto  Recv-Q  Send-Q  Local Address           Foreign Address  State
 Tcp    0       126     uranus.planete.n:telnet 192.168.1.2:1037 ESTABLISHED
 Udp    0       0       uranus.plan:netbios-dgm  *:*
 Udp    0       0       uranus.plane:netbios-ns  *:*

Active UNIX domain sockets (w/o servers)
 Proto  RefCnt  Flags        Type        State         I-Node Path
 unix   2       [ ]          STREAM      1990     /dev/log
 unix   2       [ ]          STREAM CONNECTED 1989
 unix   1       [ ]          DGRAM        1955
```

Explications sur la première partie qui affiche l'état des connexions :

Proto : Protocole utilisé

Recv−q : nbre de bits en réception pour ce socket

Send−q : nbre de bits envoyés

LocalAdress : nom d'hôte local et port

ForeignAdress : nom d'hôte distant et port

State : état de la connexion

Le champ state peut prendre les valeurs suivantes:

Established : connexion établie

Syn snet : le socket essaie de se connecter

Syn recv : le socket a été fermé

Fin wait2 : la connexion a été fermée

Closed : le socket n'est pas utilisé

Close wait : l'hôte distant a fermé la connexion; Fermeture locale en attente.

Last ack : attente de confirmation de la fermeture de la connexion distante

Listen : écoute en attendant une connexion externe.

Unknown : état du socket inconnu

Explications sur la deuxième partie qui affiche l'état des sockets (IPC − *Inter Processus Communication*) actifs :

Proto : Protocole, en général UNIX,

Refcnt : Nombre de processus associés au socket

Type : Mode d'accès datagramme (DGRAM), flux orienté connexion (STREAM), brut (RAW), livraison fiable des messages (RDM)

State : Free, Listening, Unconnected, connecting, disconnecting, unknown

Path : Chemin utilisé par les processus pour utiliser le socket.

Affichage et état des tables de routage avec netstat : ***netstat −nr*** *ou* ***netstat −r***

```
Kernel IP routing table
Destination   Gateway   Genmask         Flags   MSS    Window  irtt  Iface
192.168.1.0   *         255.255.255.0   U       1500   0       0     eth0
127.0.0.0     *         255.0.0.0       U       3584   0       0     lo
```

Explications sur la commande ***netstat −r***

Destination : adresse vers laquelle sont destinés les paquets

Gateway : passerelle utilisée, * sinon

Flags : G la route utilise une passerelle, U l'interface est active, H on ne peut joindre qu'un simple hôte par cette route)

Iface : interface sur laquelle est positionnée la route.

Affichage de statistiques avec ***netstat −i***

```
Iface MTU  Met RX−OK RX−ERR RX−DRP RX−OVR TX−OK TX−ERR TX−DRP TX−OVR Flags
Lo    3584 0   89    0      0      0      89    0      0      0      BLRU
eth0  1500 0   215   0      0      0      210   0      0      0      BRU
```

Explications sur la commande ***netstat −i***

RX−OK et TX−OK rendent compte du nombre de paquets reçus ou émis,

RX−ERR ou TX−ERR nombre de paquets reçus ou transmis avec erreur,

RX−DRP ou TX−DRP nombre de paquets éliminés,

RX−OVR ou TX−OVR recouvrement, donc perdus à cause d'un débit trop important.

Les Flags (B adresse de diffusion, L interface de loopback, M tous les paquets sont reçus, O arp est hors service, P connexion point à point, R interface en fonctionnement, U interface en service)

Exercices :

On donne les résultats de 3 commandes netstat ci−dessous, extraites de la même machine :

$ netstat −nr

Kernel IP routing table

Destination Gateway Genmask Flags MSS Window irtt Iface

198.5.203.0 0.0.0.0 255.255.255.0 U 1500 0 0 eth0

127.0.0.0 0.0.0.0 255.0.0.0 U 3584 0 0 lo

0.0.0.0 198.5.203.3 0.0.0.0 UG 1500 0 0 eth0

$ netstat

Active Internet connections (w/o servers)

Proto Recv−Q Send−Q Local Address Foreign Address State

Tcp 0 127 uranus.toutbet:telnet 194.206.6.143:1027 ESTABLISHED

$ netstat −i

Iface MTU Met RX−OK RX−ERR RX−DRP RX−OVR TX−OK TX−ERR TX−DRP TX−OVR Flags

Lo 3584 0 764 0 0 764 89 0 0 0 BLRU

eth0 1500 0 410856 0 0 33286 210 0 0 0 BRU

On demande :

1. Quels sont les noms et adresse de la machine consultée ?
2. Quel type de session est−elle en train de supporter ?
3. À quoi correspond l'adresse 198.5.203.3 ?
4. Pourquoi une interface porte−t−elle les Flags BLRU et l'autre BRU ?
5. Quelle est la taille des paquets utilisée par la passerelle par défaut ?

5.3.5. La commande traceroute

La commande **traceroute** permet d'afficher le chemin parcouru par un paquet pour arriver à destination. Cette commande est importante, car elle permet d'équilibrer la charge d'un réseau, en optimisant les routes.

Voici le résultat de la commande **traceroute *www.nat.fr***, tapée depuis ma machine.

```
 traceroute to sancy.nat.fr (212.208.83.2), 30 hops max, 40 byte packets
  1  195.5.203.9 (195.5.203.9)  1.363 ms  1.259 ms  1.270 ms
  2  194.79.184.33 (194.79.184.33)  25.078 ms  25.120 ms  25.085 ms
  3  194.79.128.21 (194.79.128.21)  88.915 ms  101.191 ms  88.571 ms
  4  cisco−eth0.frontal−gw.internext.fr (194.79.190.126)  124.796 ms[]
  5  sfinx−paris.remote−gw.internext.fr (194.79.190.250)  100.180 ms[]
  6  Internetway.gix−paris.ft.NET (194.68.129.236)  98.471 ms       []
  7  513.HSSI0−513.BACK1.PAR1.inetway.NET (194.98.1.214)  137.196 ms[]
  8  602.HSSI6−602.BACK1.NAN1.inetway.NET (194.98.1.194)  101.129 ms[]
  9  FE6−0.BORD1.NAN1.inetway.NET (194.53.76.228)  105.110 ms       []
 10  194.98.81.21 (194.98.81.21)  175.933 ms  152.779 ms  128.618 ms[]
 11  sancy.nat.fr (212.208.83.2)  211.387 ms  162.559 ms  151.385 ms[]
```

Explications :

Ligne 0 : le programme signale qu'il n'affichera que les 30 premiers sauts, et que la machine www du domaine nat.fr, porte le nom effectif de `sancy`, dans la base d'annuaire du DNS du domaine nat.fr. Cette machine porte l'adresse IP 212.208.83.2. Pour chaque tronçon, on a également le temps maximum, moyen et minimum de parcours du tronçon.

Ensuite, on a pour chaque ligne, l'adresse du routeur que le paquet a traversé pour passer sur le réseau suivant.

Ligne 4 et 5, le paquet a traversé 2 routeurs sur le même réseau 194.79.190.

Ligne 4, 5, 6, 7, 8, 9, 11, on voit que les routeurs ont un enregistrement de type A dans les serveurs de noms, puisqu'on voit les noms affichés.

Conclusion : depuis ma machine, chaque requête HTTP passe par 11 routeurs pour accéder au serveur www.nat.fr.

L'accès sur cet exemple est réalisé sur Internet. Un administrateur, responsable d'un réseau d'entreprise sur lequel il y a de nombreux routeurs, peut, avec cet outil, diagnostiquer les routes et temps de routage. Il peut ainsi optimiser les trajets et temps de réponse.

5.3.6. La commande dig

La commande **dig** remplace ce qui était la commande **nslookup**. Cette commande sert à diagnostiquer des dysfonctionnements dans la résolution de nom. (Service DNS).

Utilisation simple de **dig** :

```
$ dig any freenix.org
; <<>> DiG 9.2.2 <<>> any freenix.org
;; global options:  printcmd
;; Got answer:
```

```
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 21163
;; flags: qr rd ra; QUERY: 1, ANSWER: 4, AUTHORITY: 3, ADDITIONAL: 3

;; QUESTION SECTION:
;freenix.org.                   IN      ANY

;; ANSWER SECTION:
freenix.org.            92341   IN      SOA     ns2.freenix.org.\
                                             hostmaster.freenix.org.\
                                                        2003042501\
                                                             21600\
                                                              7200\
                                                           3600000\
                                                            259200\

freenix.org.            117930  IN      NS      ns2.freenix.fr.
freenix.org.            117930  IN      NS      ns.frmug.org.
freenix.org.            117930  IN      NS      ns6.gandi.net.

;; AUTHORITY SECTION:
freenix.org.            117930  IN      NS      ns2.freenix.fr.
freenix.org.            117930  IN      NS      ns.frmug.org.
freenix.org.            117930  IN      NS      ns6.gandi.net.

;; ADDITIONAL SECTION:
ns2.freenix.fr.         16778   IN      A       194.117.194.82
ns.frmug.org.           40974   IN      A       193.56.58.113
ns6.gandi.net.          259119  IN      A       80.67.173.196

;; Query time: 197 msec
;; SERVER: 213.36.80.1#53(213.36.80.1)
;; WHEN: Tue May 27 15:16:23 2003
;; MSG SIZE  rcvd: 248
```

retourne les informations sur le domaine concerné.

Il est ensuite possible d'intérroger sur tout type d'enregistrement : SOA, MX, A, CNAME, PTR...

5.3.7. La commande host

La commande **host** interroge les serveurs de coms. Elle peut par exemple être utilisée pour détecter des dysfonctionnement sur un réseau (serveurs hors services). Attention, n'utilisez pas cette commande sur des réseaux dont vous n'avez pas l'administration.

Chapitre 6. Installation d'un serveur Telnet et FTP

Émulation VTx et transfert de fichier.

Émulation VTx et transfert de fichier.

Le document décrit l'installation d'un service de transfert de fichier, la configuration du démon inetd et quelques premiers aspects touchant à la sécurité des services sous GNU/Linux.

Mots clés : Telnet, FTP, ssh, sftp, scp, TCP−Wrapper

6.1. Description et objectifs de la séquence

Vous devriez à la fin pouvoir :

- utiliser le service FTP du serveur à partir d'un client quelconque du réseau
- bénéficier du service ftp anonyme ou authentifié,
- pouvoir filtrer l'accès provenant de tout ou partie du réseau avec TCP−Wrapper.

6.2. Présentation des concepts importants

1) Telnet:

Telnet est un protocole qui permet l'émulation de terminal VTx à distance sur un serveur Unix/Linux.

2) FTP:

FTP est un protocole de communication qui permet le transfert de fichiers entre plusieurs machines.

3) Le daemon inetd:

Toute application fonctionnant sous TCP/IP est basée sur le modèle client/serveur. Par exemple quelqu'un se connectant via telnet à un hôte distant « active » chez l'hôte le service serveur telnetd.

Chaque serveur est sur une machine en attente d'une connexion sur un port particulier. Dans les premières versions d'Unix−TCP/IP chaque application (telnet, ftp,...) avait son propre serveur qui était lancé au démarrage de chaque machine comme un "daemon". Cette stratégie encombrait inutilement la table des processus (autant de serveurs que de services). Ces services sont dits fonctionnant en mode « autonome » ou « standalone ».

Le daemon INETD est un « super » serveur, à l'écoute sur plusieurs ports et qui se charge de recevoir les demandes de connexion de plusieurs clients (telnet, ftp,...) et de lancer le serveur correspondant à la demande. A son démarrage il consulte les fichiers:

− /etc/services qui contient la liste générale des services TCP/IP avec leur numéro de port et le protocole de transport associé.

− /etc/inetd.conf qui contient la liste des services activés sur une machine donnée

Dans les distributions récentes (Mandrake 10.x, RedHat 9.x...), inetd a été remplacé par xinetd. Le principe est très similaire, à la seule différence que, dans /etc/etc/xinetd.d, chaque service (telnet, ftp, pop3...) dispose de son propre fichier de configuration.

Certains services utilisables avec inetd ou xinetd comme telnet, ftp, pop3... sont difficilement sécurisables car les mots de passe transitent en clair sur le réseau. Ce problème sera vu ultérieurement avec les TPs sur la métrologie. Si ces services sont utilisables en l'état sur des petits réseaux isolés, il faudra éviter de les utiliser sur des réseaux reliés à Internet ou dans des environnements peu sûrs. Cependant, la tendance est au cryptage de ces services, grâce à SSL notamment. Il existe une version sécurisée de telnet, nommée telnet−ssl.

6.3. Extrait de /etc/services :

/etc/services :

```
ftp  21/tcp
telnet  23/tcp
smtp  25/tcp  mail
pop3  110/tcp  # Post Office
```

etc...

6.4. Extrait de /etc/inetd.conf

```
ftp      stream  tcp     nowait  root    /usr/sbin/ftpd          ftpd
#shell   stream  tcp     nowait  root    /usr/sbin/rshd          rshd
#login   stream  tcp     nowait  root    /usr/sbin/rlogind       rlogind
#exec    stream  tcp     nowait  root    /usr/sbin/rexecd        rexecd
```

Ici, il n'y a que le service ftp qui est activé par le serveur inetd. Les autres lignes sont en commentaires.

Ces services sont dits fonctionnant en mode « parallèle ».

6.5. Configuration avec xinetd

Le principe est similaire, à la différence que vous avez un fichier de configuration global "/etc/xinetd.conf", et un fichier de configuration par service, en général dans le répertoire "/etc/xinetd.d/".

```
#
# Le fichier xinetd.conf
#
# Some defaults, and include /etc/xinetd.d/

defaults
{
        instances               = 60
        log_type                = SYSLOG authpriv
        log_on_success          = HOST PID
        log_on_failure          = HOST
        cps                     = 25 30
}

includedir /etc/xinetd.d
```

Le fichier /etc/xinetd.d/wu−ftpd

```
# default: on
# description: The wu-ftpd FTP server serves FTP connections. It uses \
#       normal, unencrypted usernames and passwords for authentication.
service ftp
{
        disable = no
        socket_type             = stream
        wait                    = no
        user                    = root
        server                  = /usr/sbin/in.ftpd
```

```
        server_args             = −l −a
        log_on_success          += DURATION USERID
        log_on_failure          += USERID
        nice                    = 10
}
```

Le paramètre "disable", permet d'activer/désactiver le service.

le programme "in.ftpd", indique bien que le service est pris en charge par TCPWrapper (C'est le in qui l'indique, sinon, le binaire s'appellerait ftpd).

Les commentaires en haut du fichier indiquent que ce service ne prend pas en charge l'encryptage des mots de passe.

6.6. TCP−Wrapper

TCP−Wrapper est un outil de sécurité réseau qui permet de contrôler les accès, les tentatives de connexion sur une machine donnée. Il permet à tout instant de savoir (par journalisation syslogd) qui essaie d'accéder sur un ordinateur mais également de filtrer les accès. On peut par exemple sur une machine A interdire les connexions telnet venant d'une machine B tout en autorisant les connexions FTP venant de cette même machine B.

Principe de fonctionnement:

Exemple: Si inetd reçoit une demande de connexion sur le port 23 il va lancer telnetd.

Tcpwrapper sert d'enveloppe. Il vient « s'intercaler » entre le daemon inetd et le serveur à démarrer. Quand une demande de service TCP/IP (en réalité TCP ou UDP) arrive sur un port donné, inetd va lancer TCPD (daemon correspondant à Tcpwrapper) au lieu d'activer directement le service demandé (telnetd, ftpd, pop3...).

Tcpd prend en charge la requête et met en place ses mécanismes de contrôle. Il peut par exemple vérifier que les accès depuis la machine cliente sont autorisés. Une fois le traitement terminé il va (s'il y a autorisation) lancer son propre service in.telnetd, in.ftpd, in.imapd....

6.7. Éléments de configuration

Sous Linux, tcpd est installé par défaut. On peut voir en consultant le fichier /etc/inetd.conf comment inetd active tcpd.

6.7.1. Extrait de /etc/inetd.conf

```
ftp     stream  tcp     nowait  root    /usr/sbin/tcpd  in.ftpd −l −a
telnet  stream  tcp     nowait  root    /usr/sbin/tcpd  in.telnetd
```

6.7.2. TCP Wrapper

L'administrateur réseau va pouvoir utiliser 2 fichiers: /etc/hosts.allow et /etc/hosts.deny pour filtrer les accès à sa machine.

/etc/hosts.deny: on indique dans ce fichier les services et les hôtes pour lesquels l'accès est interdit.

/etc/hosts.allow: on indique dans ce fichier les services et les hôtes pour lesquels l'accès est autorisé.

Exemple:

```
# Fichier /etc/hosts.deny
# interdit tous les accès ftp à la machine
in.ftpd:ALL

# Fichier /etc/hosts.allow
# autorise les accès ftp venant de cli1
in.ftpd :cli1.archinet.edu
```

TCP−Wrapper utilise l'algorithme suivant :

```
Si une règle est applicable dans hosts.allow, alors cette règle est appliquée, sinon, Si une règle est applicab
```

Ce mode de fonctionnement induit la stratégie de sécurité à adopter :

1. décrire toutes les règles pour les couples (services/clients) qui sont autorisés,
2. interdire systématiquement tout le reste. Mettre par défaut ALL:ALL dans hosts.deny.

Les tentatives d'accès depuis des machines extérieures sont toutes enregistrées dans des fichiers particuliers. Ces enregistrements sont effectués par le processus syslogd qui, à son démarrage, lit le fichier /etc/syslog.conf pour trouver dans quel(s) fichier(s) il doit enregistrer les différentes tentatives d'accès.

6.8. Extrait de /etc/syslog.conf

```
# Log anything (except mail) of level info or higher.
# Don't log private authentication messages

# The authpriv file has restricted access.
authpriv.*                                      /var/log/auth.log
auth,authpriv.none;                             /var/log/syslog
```

6.9. Extrait de /var/log/syslog

```
Feb  3 18:02:52 ns1 ftpd[1051]: FTP session closed
Feb  3 18:03:31 ns1 syslogd 1.3-3: restart.
Feb  3 18:07:34 ns1 in.ftpd[1057]: refused connect from cli1.archinet.edu
Feb  3 18:07:46 ns1 in.ftpd[1058]: connect from ns1.archinet.edu
Feb  3 18:10:57 ns1 login[1063]: LOGIN ON ttyp3 BY mlx FROM puce
```

Remarques:

La commande *kill −HUP pid* de sysklogd permet de redémarrer ce processus avec prise en compte des paramètres se trouvant dans /etc/syslog.conf. Il est aussi possible d'invoquer le script de lancement du service en lui passant l'argument restart : /etc/init.d/syslogd restart

6.10. Consignes pour le processus d'installation et de configuration

Ouvrez le fichier /etc/inetd.conf, vérifiez que les lignes qui activent les démons telnet et ftp sont décommentées.

Vérifier qu'il n'y a aucune règle active dans les fichiers /etc/hosts.allow et /etc/hosts.deny (mettez les règles éventuelles en commentaire)

Pour l'activer manuellement utilisez la commande : /etc/init.d/inetd stop | start

6.11. Procédure de tests

1 − Créez un compte d'utilisateur.

2 − Sur la console, ouvrez une session sous le compte root.

3 − Vous devez pouvoir utiliser les commandes :

```
«#> ftp localhost » ou « ftp `hostname` » en vous authentifiant avec le compte que vous avez créé
«#> telnet localhost » ou « telnet `hostname` » en utilisant le compte que vous avez créé.
où `hostname` indique le nom d'hôte de votre machine.
```

Si ces commandes fonctionnent sur le serveur, réaliser les opérations à partir d'un client distant.

Vous pouvez vérifier le fonctionnement de « tcpwraper »

1 − Interdisez tout dans le fichier /etc/hosts.deny (mettre ALL:ALL à la fin du fichier). Attention, mettez plusieurs retours chariots (CR/LF) à la fin du fichier sinon la dernière ligne n'est pas lue. Vous avez des exemples dans "man 5 hosts_access" ou man "hosts.allow".

2 − Vérifiez que l'accès ftp est maintenant refusé, vérifiez les messages dans /var/log/syslog et /var/log/auth.log

Vous devriez voir, dans les fichiers de log (journaux) les demandes ftp rejetées par TCPWrapper.

Remettez le fichier hosts.deny dans son état initial.

6.12. Problèmes que vous pourrez rencontrer

Q − je ne peux pas accéder au serveur en utilisant le compte root.

R − Vous pouvez réaliser cette opération sur la console, mais par mesure de sécurité cette opération n'est pas possible à distance. Avec telnet, ouvrez une session sur un compte d'utilisateur standard, puis la commande « su ».

Q − Je n'arrive pas ouvrir de session Telnet ou FTP.

R − Vérifier le fichier de configuration « /etc/inetd.conf », puis que le serveur inet est bien lancé.

Q − J'ai modifié les fichiers host.allow et host.deny. Les modifications n'ont pas l'air d'être prises en compte.

R − Vérifiez la syntaxe des instructions utilisées dans ces fichiers, normalement la modification des règles est prise en compte dynamiquement sans avoir besoin de relancer le service. Insérez quelques lignes vides à la fin de ces fichiers.

Attention : les services telnet et ftp n'offrent aucune solution de sécurité sur les réseaux (transmission des données en clair). Sur un réseau qui n'est pas sûr vous ne devrez pas utiliser ces services.

Il y a d'autres fichiers de configuration qui permettent de sécuriser le service FTP. Ces fichiers, dans /etc, sont indépendants de TCP Wrapper. Regardez ftpaccess, ftpgroup, ftphosts, ftpusers et leurs pages de manuel. Avec ftpusers, vous pouvez autoriser/interdire l'accès pour un compte en particulier.

Sources de documentation complémentaires

Les pages du manuel de TCPWrapper. man syslog.conf ou man syslogd pour plus de renseignements.

Chapitre 7. TP Unix − Gestion des Utilisateurs

Mode d'utilisation du serveur Unix/Linux. Environnement BTS Informatique deuxième année.

7.1. Gestion des Utilisateurs

Objectif : Mettre à disposition un environnement à des utilisateurs, gérer les accès aux fichiers, les droits, l'environnement de travail, gérer les groupes..

Contexte : pour cet exercice, on imagine que l'on travaille dans une entreprise dont deux services sont connectés au réseau : le service Commercial, et le service Comptable. L'ambiance au sein du service de comptabilité est assez conviviale, et bien que toutes les informations traitées soient absolument secrètes (et donc interdites en accès à toute autre personne qu'un comptable), ces personnes ont l'habitude de partager leurs données. Concernant les commerciaux, au contraire, tous leurs travaux sont absolument secrets (échanges commerciaux, ristournes et remise, chiffre d'affaires, arrangements divers...)

Pré−requis : Documentation et exercice de Jean Gourdin : Document sur la site linux−keops.com

7.2. Documentation technique

La séquence de log : lorsqu'un utilisateur se logue, tout est géré par le programme **login**. Ce programme va obéir aux PAM (Pluggable Authentification Module), qui vont lui indiquer une suite d'étapes à valider. Ce point, même si il est très intéressant, ne sera pas étudié ici. Disons simplement que le fichier `/etc/passwd` sera certainement lu (si on utilise une authentification standard et locale, et pas du NIS, ou du LDAP), et que diverses informations concernant l'utilisateur seront ainsi récupérées : son nom, son répertoire home, son programme de connexion.

Le contenu du fichier `/etc/profile` sera exécuté (réglages génériques pour tous les utilisateurs). Selon le shell de connexion, d'autres scripts seront utilisés : si il s'agit d'un shell de login, et que le shell est bash, alors `~/.bash_profile` ou `~/.bash_login` ou `~/.profile` seront lus. A la déconnexion, `~/.bash_logout` sera exécuté.

Si il s'agit d'une connexion distante, d'un sous−shell, ou d'un xterm, alors `/etc/bash.bashrc` puis `~/.bashrc` seront exécutés.

7.2.1. Exercices

Créez un schéma explicatif des fichiers lus, dans les 2 cas de figures suivants : connexion bash en local, et connexion bash autres (distantes, **xterm**, etc).

Décodez le contenu de ces fichiers sur votre machine.

Expliquez pourquoi la Debian n'offre pas la visualisation en couleurs des fichiers (**ls**). Dans votre **bash**, lancez un nouveau **bash**. Testez le **ls**.. Que se passe−t−il ?

7.3. Amélioration du bash

Peut être avez vous remarqué le fichier `/etc/bash_completion`.

Vous connaissez la complétion, mais celle−ci est encore supérieure : vous le découvrirez dans l'exercice suivant

7.3.1. Exercices

Connectez−vous sur deux sessions (afin de comparer la différence), et sur l'une d'elles, activez la complétion étendue (tapez **. /etc/bash_completion** . Attention au point tout seul : très important pour que le script demandé s'exécute dans l'environnement courant, et non dans un fils (les réglages seraient alors perdus).

Testez la complétion étendue avec **cd** (tab), avec des commandes (**apt−get i**(tab)), **ssh** (tab), **man ls**(tab).

Que remarquez −vous ?

Comment faire pour bénéficier de tout ceci ?

Consultez vos propres fichiers de connexion (`.bash_profile`, et `.bashrc`). Modifiez `.bash_profile` afin qu'il exécute `.bashrc` (ainsi, `.bashrc` est exécuté toujours, en shell de login ou autre). Puis, dans `.bashrc`, dé commentez l'accès à `/etc/bash_completion`.

Testez...

7.4. /etc/skel (profil par défaut)

Extension de ces réglages à tous les nouveaux utilisateurs (ou 'du bon usage de `/etc/skel`')

Dans le répertoire /etc/skel, vous trouverez le squelette de tous les nouveaux comptes : tout ce qui y est présent sera recopié par défaut dans le répertoire de chaque nouvel utilisateur

7.4.1. Exercice

Quel est le contenu de ce répertoire ?

Modifiez le contenu de `/etc/skel` comme vous venez de le faire pour vous. Créez un nouvel utilisateur (**adduser toto**). Connectez vous avec toto, et vérifiez ses réglages.

Imaginons maintenant que nous voulons que tous les nouveaux utilisateurs se retrouvent avec une structure de home standard (par exemple, avec un fichier 'mode d'emploi du réseau' et un sous répertoire `public_html` déjà prêt pour la publication de pages web :

Créez ce répertoire et un texte 'mode_d_emploi_du_reseau.txt' dans le répertoire `/etc/skel`.

Créez deux nouveaux utilisateurs (foo et bar)

Observez leurs homes.

7.5. Droits par défaut

Gestion des droits : Vous avez certainement remarqué la commande **umask**. Cette commande définit les droits standards dont seront affublés vos fichiers. Les droits normaux sont 666 pour un fichier, et 777 pour un répertoire. **Umask** vient en soustraction pour le calcul des droits. L'emploi d'**umask** seul permet d'afficher la valeur d'umask, et umask XXXX permet de définir l'umask à XXXX.

7.5.1. Exercice

Définissez votre umask à 0000 : créez des fichiers et des répertoires et vérifiez les droits obtenus. Définissez votre umask pour être le seul à pouvoir voir vos fichiers et répertoires... Vérifiez.

7.6. Ajout de comptes

Dans le contexte décrit, on peut proposer de résoudre le problème par la création de deux groupes (commerciaux et comptables). Il semble préférable de créer le home de chaque utilisateur dans `/home/NomDuGroupe`, dans un souci de bonne gestion.

Le comportement par défaut de création des comptes est piloté par `/etc/adduser.conf`.

Selon la taille de la population d'utilisateurs à gérer, on pourra modifier ce fichier pour adapter la gestion à nos besoins. Dans notre cas, on utilisera des groupes : on créera des groupes, et on créera des utilisateurs dans ces groupes.

Modification du fichier `/etc/adduser.conf`

Ce fichier définit le fonctionnement par défaut, il est suffisamment documenté pour que vous puissiez vous débrouiller seul. On peut y définir le shell de connexion proposé par défaut, le nom du répertoire contenant les home directories, l'endroit des squelettes, etc...

7.6.1. Exercices

Expliquez ce que sont les LETTERHOMES, le rôle des directives commençant par FIRST. Comment faire pour que les homes soient créées dans un sous−répertoire de home portant le nom du groupe ? (tous les homes des comptables dans un sous répertoire `/home/comptables`)

L'ajout de groupes et d'utilisateurs se fait respectivement par les commandes **addgroup** et **adduser**. Consultez le man de ces commandes. Faites le réglage du adduser.conf correspondant, et testez l'ajout de deux groupes (testprofs et testetudiants), puis de quelques utilisateurs (profs et étudiants)

Testez ensuite le bon fonctionnement, en vous connectant en tant que certains de ces utilisateurs.

Supprimez ensuite tous ces utilisateurs, ainsi que leurs répertoires (**man userdel**)

7.7. Droits d'accès, et multigroupes

Les commandes **chmod**, **chown** et **chgrp** permettent d'attribuer, de modifier des droits sur les objets du file system (fichiers et répertoires)

(faire un **man**)

D'autre part, un utilisateur peut appartenir à plusieurs groupes (un groupe principal, et d'autres additionnels)

Cela permet une certaine souplesse dans les droits, bien que seule l'utilisation des ACLs puisse permettre de tout gérer (au prix d'une dangereuse complexité).

Pour ajouter un utilisateur dans un groupe additionnel, utilisez **adduser user groupAdditionnel**.

Vous pourrez alors donner des droits à ce groupe, et l'OS évaluera les droits de chaque utilisateur par rapport à l'ensemble de ses groupes

7.7.1. Exercice

Créez l'ensemble des comptes selon les régles définies en début de cet exercice : Les commerciaux (Bill, Bob, Carlos, Richard, Laura) et les comptables (Raymond, Georgette, Carlotta, Paula).

Ces utilisateurs peuvent avoir un site web (pensez à créer le `public_html`). Le système de gestion de courrier demande à avoir un répertoire MailDir dans chacun des homes.

Les chefs de services (Bill et Raymond) ont la possibilité d'alimenter le site web (création de pages...) tandis que les utilisateurs ne peuvent que consulter.

Chapitre 8. Travaux pratiques : Telnet et FTP

8.1. Quelques remarques

- Relevez dans /etc/services les ports utilisés par les services telnet, ftp, pop3, dns, smtp, http.
- Installez et testez les services telnet et ftp à partir de votre poste puis à partir d'un autre poste. Utilisez les traces dans les journaux pour identifier les problèmes.

  ```
  tail −f /var/log/syslog
  ```
- Utilisez TcpWrapper pour autoriser/interdire le service telnet, le service ftp, tous les services. Vous testerez l'accès à partir de votre poste, d'un autre poste.

Attention : Pensez à relancer un service serveur chaque fois que vous avez modifié son fichier de configuration, ceci est vrai pour tous les services et ne sera plus répété. En général utilisez la manipulation suivante "/etc/init.d/NomDuService start | stop | status | restart"

Il est possible que le client ftp soit remplacé par un autre programme comme "lftp" par exemple. C'est ce programme qui sera utilisé dans le TP, vous adapterez si vous utilisez autre chose. Fondamentalement ça ne changera rien, mais lftp est beaucoup plus riche fonctionnellement que les clients ftp standard. Il supporte 6 méthodes d'accès ftp, ftps, http, https, hftp et fish.

La freeduc−sup est configurée pour ne pas supporter les transactions et protocoles "non−sûrs". Si vous utilisez un client autre comme une knoppix par exemple,vous devrez installer le support ssl.

```
apt−get install telnetd−ssl
```

8.2. Configuration de telnet

1. Relevez le port utilisé par telnet dans le fichier /etc/services
2. Décommentez la ligne qui concerne telnet dans /etc/inetd.conf et relancez le service.
3. Vérifiez que le port est bien ouvert avec la commande netstat :

   ```
   #> netstat −atup | grep LISTEN
   ```
4. Vérifiez que rien dans TCP−Wrapper n'interdit l'accès au service telnet.

   ```
   # Mettre dans /etc/hosts.allow
   ALL:ALL
   ```

   Testez l'accès au service telnet.

8.3. Configuration de TCP−Wrapper

1. Interdisez tous les accès dans TCP−Wrapper, testez.

```
# Commentez toutes les lignes dans /etc/hosts.allow
# Mettez dans /etc/hosts.deny
ALL:ALL
```

2. En vous aidant des exemples donnés dans "man hosts.allow", autorisez l'accès pour une machine du réseau sur le service telnet, interdisez−le pour toutes les autres, testez.

8.4. Test de l'accès ftp authentifié

L'accès authentifié est simple à mettre en oeuvre.

1. Relevez les ports utilisés par ftp dans /etc/services.
2. Activez le service dans inetd.conf et lancez le service.
3. Vérifiez que le port est bien ouvert avec la commande netstat
4. Vérifiez que rien n'interdit l'accès au service ftp dans les fichiers hosts.allow et hosts.deny
5. Faites un test en utilisant un compte système existant, par exemple "lftp localhost −u util" si util est votre compte.

Normalement c'est terminé, tout doit fonctionner. Si cela ne fonctionne pas, vérifiez que vous n'avez rien oublié, vérifiez aussi les fichiers de log (/var/log/daemon, /var/log/syslog, /var/log/messages)

8.5. Configuration d'un service ftp anonyme

La mise en place d'un service ftp anonyme demande plus de manipulations. Vous allez mettre l'environnement dans /home.

1. Vérifiez que le fichier /etc/passwd dispose bien d'un compte ftp :

```
knoppix@master:~/tmp$ grep ftp /etc/passwd
ftp:x:1003:1003:Compte ftp anonyme:/home/ftp:/bin/true
```

   sinon modifie les fichier "/etc/passwd" pour ajouter la ligne. Attention, prenez un "UID" libre.
2. Créez un compte de groupe dans le fichier /etc/group :

```
knoppix@master:~/tmp$ grep ftp /etc/group
ftp:x:1003:
```

3. Utilisez la commande "pwconv" pour mettre à jour le fichier shadow.

```
Remarque si vous avez des problèmes d'accès sur le serveur
ftp anonyme par la suite :
Il s'agit peut être d'un problème de définition du compte
dans le fichier "/etc/shadow". Vous pouvez pour cela utiliser
plusieurs options :
A) Première option
1 − taper "pwunconv"
2 − modifier le fichier "/etc/passwd" comme indiqué ci−dessus
3 − taper "pwconv" pour "recacher" les mots de passe.

B) Deuxième option
1 − utiliser la commande "adduser ftp" qui va mettre les fichiers
"/etc/passwd" et "/etc/shadow" à jour. Mettez un mot de passe bidon.
2 − taper "pwunconv" et supprimer le mot de passe du compte dans
"/etc/passwd" (mettre "*" par exemple)
3 − remplacer aussi le shell "/bin/bash" par "/bin/true", enregistrer.
4 − taper "pwconv" pour "recacher" les mots de passe.
5 − supprimer /home/ftp (rm −rf /home/ftp)
6 − continuer la procédure ci−dessous.
```

4. Sous le compte root vous allez créer l'environnement pour le service ftp :

```
cd /home && mkdir −p ftp/lib ftp/bin ftp/pub ftp/incoming ftp/etc
```

   On copie le programme ls dans ~ftp/bin. Vous pouvez en mettre d'autres, mais soyez prudent.

```
cp /bin/ls  ftp/bin
```

   Il reste à créer les comptes dans un fichier local passwd et group. On y met juste les comptes nécessaires pour l'utilisation des programmes mis dans ~ftp/bin.

```
root@master:/home# grep ftp /etc/group > ftp/etc/group
root@master:/home# grep root /etc/passwd > ftp/etc/passwd
root@master:/home# grep ftp /etc/passwd > ftp/etc/passwd
```

   Vérifiez que vous avez bien les informations dans les fichiers ftp/passwd et ftp/group.

   On change les permissions

```
chmod −R 111 ftp/bin ; chmod 111 ftp/etc; chmod 444 ftp/etc/*;\
```

```
chmod 555 ftp/pub; chmod 1733 ftp/incoming
```

On rajoute dans ~ftp/lib les librairies utilisées par les programmes mis dans ~ftp/bin. Vous avez, vous le programme "ls". Les librairies utilisées par le programme ls sont visibles avec la commande "ldd".

Si vous ajoutez d'autres programmes, il faudra y mettre également les bonnes librairies.

```
root@master:/home# ldd /bin/ls
        librt.so.1 => /lib/librt.so.1 (0x4001f000)
        libc.so.6 => /lib/libc.so.6 (0x40031000)
        libpthread.so.0 => /lib/libpthread.so.0 (0x40141000)
        /lib/ld-linux.so.2 => /lib/ld-linux.so.2 (0x40000000)
```

Il faut donc copier toutes ces libraires dans ~ftp/lib.

```
cp /lib/librt.so.1 /lib/libc.so.6 /lib/libpthread.so.0 /lib/ld-linux.so.2 ftp/lib
```

Voici donc ce que vous devriez avoir à la fin:

```
root@master:/home# ls -alR ftp
root@mr:/home# ls -alR ftp
ftp:
total 28
drwxr-sr-x    7 root     root         4096 2003-09-19 12:19 .
drwxrwsr-x    7 root     root         4096 2003-09-19 12:21 ..
d--x--x--x    2 root     root         4096 2003-09-19 12:22 bin
d--x--x--x    2 root     root         4096 2003-09-19 12:19 etc
drwx-wx-wt    2 root     root         4096 2003-09-19 12:19 incoming
drwxr-sr-x    2 root     root         4096 2003-09-19 12:21 lib
dr-xr-xr-x    2 root     root         4096 2003-09-19 12:19 pub

ftp/bin:
total 76
d--x--x--x    2 root     root         4096 2003-09-19 12:22 .
drwxr-sr-x    7 root     root         4096 2003-09-19 12:19 ..
---x--x--x    1 root     root        64428 2003-09-19 12:22 ls

ftp/etc:
total 16
d--x--x--x    2 root     root         4096 2003-09-19 12:19 .
drwxr-sr-x    7 root     root         4096 2003-09-19 12:19 ..
-r--r--r--    1 root     root           12 2003-09-19 12:19 group
-r--r--r--    1 root     root           87 2003-09-19 12:19 passwd

ftp/incoming:
total 8
drwx-wx-wt    2 root     root         4096 2003-09-19 12:19 .
drwxr-sr-x    7 root     root         4096 2003-09-19 12:19 ..

ftp/lib:
total 1296
drwxr-sr-x    2 root     root         4096 2003-09-19 12:21 .
drwxr-sr-x    7 root     root         4096 2003-09-19 12:19 ..
-rwxr-xr-x    1 root     root        82456 2003-09-19 12:22 ld-linux.so.2
-rwxr-xr-x    1 root     root      1104040 2003-09-19 12:22 libc.so.6
-rw-r--r--    1 root     root        81959 2003-09-19 12:22 libpthread.so.0
-rw-r--r--    1 root     root        26592 2003-09-19 12:22 librt.so.1

ftp/pub:
total 8
dr-xr-xr-x    2 root     root         4096 2003-09-19 12:19 .
drwxr-sr-x    7 root     root         4096 2003-09-19 12:19 ..
```

5. C'est normalement terminé.

8.6. Test de l'accès ftp et sécurisation du service

1. Activez le service dans inetd.conf et lancez le service inetd si ce n'est pas déjà fait.
2. Vérifiez que le port est bien ouvert avec la commande netstat

```
root@mr:/home# netstat -atup | grep LISTEN
tcp        0      0 *:ftp                   *:*                     LISTEN      879/inetd
```

3. Vérifiez que rien n'interdit l'accès au service ftp dans les fichiers hosts.allow et hosts.deny
4. Commentez le fichier /etc/ftpusers comme ci-dessous :

```
# /etc/ftpusers: list of users disallowed ftp access. See ftpusers(5).
root
#ftp
#anonymous
```

5. Testez et vérifiez le bon fonctionnement de l'accès ftp anonyme (en utilisant le compte "ftp" ou "anonymous" avec la commande :

```
lftp localhost -u anonymous
```

```
ou
lftp localhost -u ftp
```

Si la configuration est correcte, vous devriez avoir le résultat suivant :

```
root@master:/home# lftp localhost -u ftp
Mot de passe: #Il n'y a pas de mot de passe, faites ENTREE
lftp ftp@localhost:~> ls
total 20
d--x--x--x    2 0        0            4096 May  5 03:35 bin
d--x--x--x    2 0        0            4096 May  5 03:35 etc
drwx-wx-wt    2 0        0            4096 May  5 03:04 incoming
dr-xr-xr-x    2 0        0            4096 May  5 03:32 lib
dr-xr-xr-x    2 0        0            4096 May  5 03:04 pub
```

6. Commentez la ligne anonymous dans le fichier /etc/ftpusers et refaites un essai de connexion.
7. Faites un test en utilisant un compte système existant, par exemple "lftp localhost -u util" si util est votre compte.
8. Restaurez l'état initial du fichier /etc/ftpusers.
9. Interdisez l'accès ftp avec TCP-Wrapper. Testez avec l'accès anonyme et en utilisant un compte authentifié.
10. Que peut-on conclure des deux méthodes de protection ?

8.7. telnet, ftp et la sécurité

On désactive en général ces services sauf cas très particulier, car les transactions ne sont pas cryptées. On préfère utiliser les services ssh, scp et sftp. Vous devez avoir un service sshd actif sur le serveur.

Exemple d'utilisation ssh :

```
[root@uranus etc]# grep ssh /etc/services
ssh             22/tcp                          # SSH Remote Login Protocol
ssh             22/udp                          # SSH Remote Login Protocol
ssh -l NomUtilisateur Machine
ssh -l mlx localhost
```

Remarque : Sur la version Live-On-CD, vous devez lancer le démon, car il n'est pas actif par défaut : /etc/init.d/sshd start. Le lancement de ce serveur permet l'utilisation de ssh et de scp à partir de clients.

Exemple d'utilisation de sftp :

```
[root@uranus etc]# grep sftp /etc/services
sftp            115/tcp
sftp            115/udp

[root@uranus etc]# sftp
usage: sftp [-1vC] [-b batchfile] [-osshopt=value] [user@]host[:file [file]]
[root@uranus etc]# sftp  mlx@localhost
Connecting to localhost...
mlx@localhost's password:

Vous pouvez ensuite envoyer ou récupérer des fichiers entre les 2 machines.
```

Exemple d'utilisation de scp :

```
SYNOPSIS
     scp [-pqrvC46] [-S program] [-P port] [-c cipher] [-i identity_file]
         [-o option] [[user@]host1:]file1 [...] [[user@]host2:]file2

# Exporte le fichier "unfichierlocal"
 scp -S ssh unfichierlocal  mlx@hotedistant:/un/chemin/distant/unfichierdistant
# Importe de "hotedistant", le fichier distant "unfichierdistant"
 scp -S ssh mlx@hotedistant:/un/chemin/distant/unfichierdistant unfichierlocal
```

La première ligne exporte un fichier, la deuxième importe. Le compte utilisé est mlx. La transaction est encryptée avec ssh.

Chapitre 9. scp, sftp et les tunnels avec ssh

Approche de ssh, des tunnels et des services mandataires

9.1. Présentation

Un des principaux risques sur les réseaux provient de "l'écoute" possible puisque toutes les données transitent, si rien n'est fait, en clair. C'est à dire qu'elles ne sont pas chiffrées.

Il est ainsi possible de récupérer sans difficulté les mots de passe des personnes utilisant le réseau, leur messages, et d'espionner toutes leurs transactions, y compris celles passées sur des serveurs HTTP. Ceci est lié à la méthode d'accès des réseaux. Le principe de la commutation par switch permet de limiter un peu les risques mais n'est pas imparable.

Il existe des solutions permettant de sécuriser un minimum les transactions. Nous allons voir rapidement quelques modes d'utilisations de ssh et d'autres produits comme scp, sftp, unisson et rsync afin de voir comment évoluer dans un environnement plus sûr.

Il sera fait référence à la maquette suivante :

Figure 9−1. Schéma maquette

Qu'est ce que ssh ?

En fait ssh Secure Shell, propose un shell sécurisé pour les connexions à distance et se présente dans ce domaine comme le standard "de fait". Mais ce n'est pas que cela. Nous allons essayer de voir quelques modes d'utilisation de ce produit.

Dans une transaction traditionnelle, un client (personne ou programme) émet une requête TCP vers un serveur. Il y a un processus serveur utilisant un port et un processus client utilisant également un port. Par exemple pop3. Il y a donc un processus serveur et processus client.

Avec ssh, il sera possible d'établir un tunnel chiffré entre le client et le serveur.

Il faut bien comprendre ce dont il s'agit et des processus mis en oeuvre.

Sur le serveur vous allez avoir 2 processus serveurs. Le serveur pop3 et le serveur SSH. Sur le client, vous allez également avoir 2 processus. Le client pop3 et le client ssh. Le client pop3 se connecte au tunnel (le client ssh local). Le serveur pop3, est relié au serveur ssh distant. Les transactions passent dans le tunnel.

Le client ssh devient un serveur mandataire (proxy) pour le protocole tunnelé. Le serveur ssh devient le client proxy pour le serveur.

Figure 9−2. Schéma du fonctionnement

Sur le diagramme, le canal entre le port tcp de l'application cliente et le port client du tunnel n'est pas chiffré. Il en est de même entre le port tcp de l'application serveur et le port serveur du tunnel. Seul, le canal entre les 2 ports du tunnel est chiffré.

L'application cliente, n'utilise plus le port par défaut que lequel écoute normalement le serveur.

L'utilisation d'un tunnel ssh impose des contraintes. Par exemple, dans l'exemple ci−dessus, si vous voulez utiliser l'application cliente avec un autre serveur, il faudra recréer un autre tunnel et reconfigurer le client. Vous pouvez créer deux tunnels différents, mais vous devrez avoir deux applications clientes utilisant des ports différents. Tout cela convient bien sur des environnements simples, mais reste un peu contraignant si l'environnement est complexe.

Pour tunneler un réseau ou de multiples clients, le mieux est d'installer un serveur ssh capable de recevoir plusieurs connexions et servant ainsi de serveur proxy aux clients du réseau pour un service donné et routant les requêtes dans un tunnel sécurisé vers le serveur d'application concerné.

L'objectif est de pouvoir supprimer d'un serveur toute application utilisant des protocoles "non sûrs" (ftp, telnet, rsh...), pour les remplacer par des applications plus sûres, ou alors, s'il n'est pas possible de les désactiver, de les faire passer dans un tunnel crypté.

Des programmes de la même famille comme "scp" ou "sftp" remplacent les commandes ftp ou rcp.

Avec ssh la totalité de la transaction entre un client et le serveur est cryptée. Le client à la possibilité d'utiliser des applications X distantes (X11 forwarding) à partir d'une invite shell dans un environnement sécurisé. On se sert également de SSH pour sécuriser des transactions non sûres comme pop3 ou imap par exemple. De plus en plus, ces applications supportent maintenant SSL aussi bien pour les clients que pour les serveurs.

SSH sur GNU/Linux est généralement composé de 3 packages :

```
OpenSSH général, (openssh),
le serveur OpenSSH (openssh−server)
le client (openssh−clients).
```

Les packages OpenSSH requièrent le paquetage OpenSSL (openssl).

Chaque fois que cela est possible vous devez utiliser une solution qui chiffre vos transactions.

9.2. Mode de fonctionnement de SSH

L'établissement du dialogue entre le client et le serveur suit un protocole particulier :

1. établissement d'une couche transport sécurisée

2. chiffrement des données à l'aide de clefs symétriques pendant la transaction

Le client peut s'authentifier en toute sécurité, et accéder aux applications conformes aux spécifications du protocole.

9.2.1. Mode de fonctionnement de la couche transport SSH

La couche transport assure le chiffrement et le déchiffrement des données. Elle assure également la compression pour améliorer le transfert. Le client et le serveur négocient plusieurs éléments afin que la session puisse s'établir.

1. l'échange des clés
2. l'algorithme de clé publique à utiliser
3. l'algorithme de chiffrement symétrique à utiliser
4. l'algorithme d'authentification de message à utiliser
5. l'algorithme repère (hash) à utiliser

Lors du premier échange, le client ne connaît pas le serveur. Le serveur propose alors une clé hôte qui servira par la suite au client de moyen d'identification du serveur.

```
M0:$ ssh −l mlx M1.foo.org
Warning: Permanently added 'M1,x.y.z.t' (DSA) to the list of known hosts.
mlx@M1.foo.org's password:
Last login: Sat Nov  2 11:37:32 2002 from 212.47.248.114
Linux 2.2.19.
mlx@M1.foo.org:$
```

Voilà un idée de ce que cela donne :

```
freeduc−sup.alt.eu.org,144.85.15.72 ssh−rsa AAAAB3NzaC1y (...)
pegase,195.115.88.38 ssh−rsa AAAAB3NzaC1yc2EAAAABIwAAAIE (...)
freeduc−sup.eu.org,137.194.161.2 ssh−dss AAAAB3NzaC1kc3M (...)
(...)
```

Le risque de ce procesus, vient donc surtout de la première transaction, où le client n'a pas le moyen d'identifier de façon fiable le serveur avec qui il communique. Un pirate peut dans certains cas, tenter de détourner cette opération. Au cours d'un échange, le client et le serveur modifient régulièrement leurs clés. À chaque opération de renouvellement de clé, un pirate qui aurait réussit à décrypter les clés, devrait refaire toute l'opération.

Authentification :

Une fois le tunnel sécurisé mis en place, le serveur envoie au client les différentes méthodes d'authentification qu'il supporte. Dans le cadre d'une authentification par mot de passe, celui−ci peut être envoyé en toute sécurité puisqu'il est chiffré.

Connexion :

Une fois l'authentification réalisée, le tunnel SSH peut multiplexer plusieurs canaux en délégant la tâche à des agents.

```
[mlx@M1 X11]$ pstree −l 24245
sshd−−−bash−+−drakfw
            |−pstree
            |−2*[xclock]
            `−xlogo
```

Ici on voit dans la même session ssh, 2 canaux pour xclock, 1 pour xlogo et 1 pour la commande pstree. Chaque canal est numéroté. Le client peut fermer un canal sans pour autant fermer toute la session.

9.2.2. Fichiers de configuration d'OpenSSH

OpenSSH est constitué de deux ensembles de fichiers de configuration, comme c'est en générale le cas sur les services sous linux (ldap, samba..). Il y a un fichier de configuration pour les programmes clients (ssh, scp et sftp) et l'autre pour le service serveur(sshd). Les informations de configuration SSH qui s'appliquent à l'ensemble du système sont stockées dans le répertoire /etc/ssh : Voici les principaux fichiers de configuration.

```
ssh_config              fichier de configuration client SSH pour l'ensemble
                        du système. Il est écrasé si un même fichier est
                        présent dans le répertoire personnel de
                        l'utilisateur
                        (~/.ssh/config).
sshd_config             fichier de configuration pour sshd.
ssh_host_dsa_key        clé DSA privée utilisée par sshd.
ssh_host_dsa_key.pub    clé DSA publique utilisée par sshd.
ssh_host_key            clé RSA privée utilisée par sshd pour la
                        version 1 du protocole SSH.
ssh_host_key.pub        clé RSA publique utilisée par sshd pour la
                        version 1 du protocole SSH.
ssh_host_rsa_key        clé RSA privée utilisée par sshd pour la
                        version 2 du protocole SSH.
ssh_host_rsa_key.pub    clé RSA publique utilisée par sshd pour la
                        version 2 du protocole SSH.
```

Les informations spécifiques à un l'utilisateur sont stockées dans son répertoire personnel à l'intérieur du répertoire ~/.ssh/:

```
authorized_keys ou parfois   authorized_keys2
                           ce fichier contient une liste de clés
                           publiques "autorisées". Si un utilisateur
                           se connecte et prouve qu'il connaît la clé
                           privée correspondant à l'une de ces clés,
                           il obtient l'authentification. Notez qu'il
                           ne s'agit que d'une méthode d'authentification
                           facultative.

id_dsa                     contient l'identité d'authentification
                           DSA de l'utilisateur.
id_dsa.pub                 la clé DSA publique de l'utilisateur.
id_rsa                     la clé RSA publique utilisée par sshd pour
                           la version 2 du protocole SSH.
identity                   la clé RSA privée utilisée par sshd pour la
                           version 1 du protocole SSH.
known_hosts                ce fichier contient les clés hôte DSA des
                           serveurs SSH auxquels l'utilisateur s'est connecté.
```

Exemple sur une machine :

```
mlx@M1:~$ ls −al .ssh/
total 16
drwx------   2 mlx mlx     4096 Jan 16  2004 ./
drwxr−xr−x   5 mlx mlx     4096 Oct 18 16:12 ../
−rw-------   1 mlx mlx     1192 Mar 11  2004 authorized_keys2
−rw-------   1 mlx mlx      240 Jan 16  2004 known_hosts
```

9.3. Configurer et utiliser SSH

Nous allons faire nos premières expérences avec ssh. Il vous faut un client et un serveur. C'est mieux.

9.3.1. Premiers pas

L'idée est de mettre en place une procédure entre un client et un serveur qui garantit des transactions sécurisées. À la fin, vous pourrez utiliser les ressources du serveur distant dans un tunnel, sans avoir à vous authentifier à chaque fois.

Pour ceux qui ont déjà utilisé les commandes rlogin, rsh, rcp... et les fichiers .rhosts, le principe est identique. La différence fondamentale est que, avec ssh, tout est encrypté.

Pour cela vous devez metter en place les clés qui serviront à ssh pour vous authentifier. Concrètement cela consiste à définir une paire de clé, une publique que vous mettrez sur le serveur distant, une privée que vous conserverez sur votre machine.

La première chose à faire est de vous créer une clé. Voyons comment réaliser cela.

Tout d'abord allez dans votre répertoire personnel.

```
$ cd
$ ssh−keygen −t dsa
$ ssh−keygen −t dsa
Generating public/private dsa key pair.
Enter file in which to save the key (/home/mlx/.ssh/id_dsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/mlx/.ssh/id_dsa.
Your public key has been saved in /home/mlx/.ssh/id_dsa.pub.
The key fingerprint is:
5c:d9:d8:c5:3d:8d:0b:10:33:42:9c:83:45:a1:d0:61 mlx@neptune.foo.org
```

Vérification de .ssh Attention il y a un "." devant ssh

```
[mlx@neptune mlx]$ ls −alR .ssh
.ssh:
total 20
drwx------    2 mlx      mlx          4096 nov 11 18:19 ./
drwx--x--x   37 mlx      mlx          4096 nov 11 18:09 ../
−rw-------    1 mlx      mlx           736 nov 11 18:19 id_dsa
−rw-r--r--    1 mlx      mlx           616 nov 11 18:19 id_dsa.pub
−rw-r--r--    1 mlx      mlx          1956 nov 10 19:38 known_hosts
```

Cette commande a généré une clé DSA par défaut de 1024 bits. La clé privée sera stockée dans ~/.ssh/id_dsa et la clé publique dans ~/.ssh/id_dsa.pub.

Si vous voulez générer une clé RSA2, utilisez l'option "−t rsa" et pour du RSA1 "−t rsa1". Vous devrez entrer une "passphrase". Entre 10 et 30 caractères. Mélangez majuscules, minuscules et chiffres. La clé privée doit ensuite être mise en

lecture seule pour le propriétaire et aucun accès pour les autres.

Pour modifier votre "passphrase" sur une clé privée DSA, utilisez la commande :

```
M0:$ ssh−keygen −p −f ~/.ssh/id_dsa
Enter old passphrase:
Key has comment '/home/mlx/.ssh/id_dsa'
Enter new passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved with the new passphrase.
```

Attention, on oublie pas une "passphrase". Il va falloir maintenant copier la clé publique vers le ou les serveurs distants. Il est préférable que vous ayez un compte, sinon vous devrez demander à l'administrateur distant de réaliser la procédure.

La clé publique, doit être copiée sur le serveur distant dans ~/.ssh/authorized_keys. La clé privée reste sur votre client. Vous pouvez mettre plusieurs clés publiques sur le serveur, si vous le souhaitez ou si vous accédez au serveur avec plusieurs comptes d'accès différents.

Copiez la clé avec scp sur le compte que vous avez sur le serveur :

```
MO:$ cat .ssh/id_dsa.pub | ssh mlx@M1.foo.org \
                    "cat − >>.ssh/authorized_keys[2]"
# Attention, sur certaines machines, le fichier se nomme
# authorized_keys, sur d'autres authorized_keys2
# Dasn l'exemple qui est donné, le répertoire .ssh doit exister.

Warning: Permanently added 'M1.foo.org' (RSA) to the list of known hosts.
mlx@M1.foo.org's password:
```

Le système demande mon mot de passe.

Elle est transférée. On doit pouvoir maintenant réaliser des opérations (commandes) comme scp sur le serveur distant, sans avoir à saisir de mot de passe. Ici on va faire un "ls", sans ouvir de session sur la machine distante.

```
M0:$ ssh mlx@M1.foo.org ls
Enter passphrase for key '/home/mlx/.ssh/id_dsa':
d1
d2
d3
d4
```

Ça marche, le système distant ne me demande plus mon mot de passe, par contre il me demande la "passphrase". Il va falloir aussi essayer de se passer de ça, car s'il est fastidieux de saisir son mot de passe, il est encore plus embêtant de saisir une "passphrase". Nous verrons comment se passer de ça avec un agent.

Remarque : Envoyer une clé par mail n'est pas un système sûr, et même chiffré et signé cela ne garantit pas au destinataire que vous en êtes l'émetteur s'il ne vous a jamais vu. L'administrateur distant peut demander à ce que l'envoyeur justifie qu'il est bien celui qui a envoyé la clé. Il suffit pour cela de téléphoner à l'administrateur et de communiquer "la signature ou empreinte" (finger print) de la clé (ou par sms). On utilise le même procédé pour identifier et vérifier la validité des clés gpg.

Pour obtenir le "finger print" d'une clé utiliser la commande :

```
M0:$ ssh−keygen −l
Enter file in which the key is (/home/mlx/.ssh/id_rsa): .ssh/id_dsa
1024 5c:d9:d8:c5:3d:8d:0b:10:33:42:9c:83:45:a1:d0:61 .ssh/id_dsa.pub
M0:$ ssh−keygen −l
Enter file in which the key is (/home/mlx/.ssh/id_rsa): .ssh/id_dsa.pub
1024 5c:d9:d8:c5:3d:8d:0b:10:33:42:9c:83:45:a1:d0:61 .ssh/id_dsa.pub
```

9.3.2. Utiliser un agent ssh

L'utilisation d'un agent, évite d'avoir à retaper la "passphrase" à chaque fois que l'on sollicite l'utilisation de la clé privée. Un agent stocke en mémoire les clés privées. Voici comment activer un agent :

```
M0:$ ssh−agent
```

La commande met sur la sortie standard des variables environnement à déclarer et à exporter. Faites le.

```
M0:$ SSH_AUTH_SOCK=/tmp/ssh−XXXB76f4/agent.2888; export SSH_AUTH_SOCK;
M0:$ SSH_AGENT_PID=2889; export SSH_AGENT_PID;
M0:$ echo Agent pid 2889;
```

On va maintenant exporter les clés. Cela consiste à les mettre dans le cache de l'agent avec la commande ssh−add. La commande demandera la "passphrase"..

```
M0:$ ssh−add
Enter passphrase for /home/mlx/.ssh/id_dsa:
Identity added: /home/mlx/.ssh/id_dsa (/home/mlx/.ssh/id_dsa)
```

On vérifie maintenant la connexion.

```
M0:$ ssh mlx@M1.foo.org ls
d1
d2
d3
d4
```

Ça marche, nous n'avons plus besoin de taper le mot de passe, ni la "passphrase". Pour supprimer une clé (ici RSA) de l'agent, utilisez l'option "−d"

```
M0:$ ssh−add −d ~/.ssh/id_rsa

Utilisez l'option −D pour supprimer toutes les clés de l'agent.
Utilisez l'option −l pour savoir quelles clés sont chargées par l'agent.
```

9.3.3. Automatisation dans X

Cela peut être automatisé dans un script.

"ssh−agent" est un daemon dont le but est d'intercepter la clé publique lorsque le programme "ssh−add" la lit après avoir demandé la passphrase. "ssh−agent" doit ensuite transmettre la clé aux processus dont il est le "père" du point de vue du système. "ssh−agent" fait bénéficier les processus fils (fork) de ses propriétés.

Pour étendre cela aux session X, si vous lancez l'interface graphique manuellement, cela deviendra par exemple :

```
M0:$ ssh−agent xinit
ou
M0:$ ssh−agent startx
```

Dans tous les cas, il est nécessaire que les variables d'environnement soient définies avant le lancement du serveur X, qui les exportera par défaut à l'ensemble de vos applications graphiques (lancées sous X).

Cela peut aussi être fait dans le fichier startx ou bien ~/.xinitrc ou bien ~/.xsession, en résumé avant le lancement de votre gestionnaire de fenêtres. Par exemple, si c'est le fichier ~/.xinitrc qui lance le gestionnaire de fenêtres, il ressemblera à ceci:

```
#!/bin/sh

ssh−agent −s > /tmp/ssh.keys    # pour y mettre les variables environnement
.  /tmp/ssh.keys                # Exporter les variables
rm  /tmp/ssh.keys               # Faire le ménage après
startx
```

Ainsi, au prochain lancement d'un serveur X, vous n'aurez plus qu'à ouvrir une fenêtre xterm et taper la commande: ssh−add

Une autre solution consiste à mettre dans son .xinit ou .xsession :

```
/usr/local/bin/ssh−add $HOME/.ssh/id_dsa < /dev/null
```

La commande ssh−add demande la passphrase

Pour les sessions xdm par exemple, modifier les fichiers de démarrage :

```
#Au début de /etc/X11/Xsession:
AGENT=$(type −p ssh−agent)
if [ −x "$AGENT" −a −z "$SSH_AUTH_SOCK" ]; then
    if [ −r $HOME/.ssh/identity −o −r $HOME/.ssh2/identification \
             −o −r $HOME/.ssh/id_dsa −o −r $HOME/.ssh/id_rsa ]; then
        SSH_AGENT="$AGENT −−"
    fi
fi
```

9.4. Comprendre la redirection de port (Port Forwarding)

Avant d'aller plus loin il est important de bien comprendre ce qu'est le "port forwarding".

Nous avons vu qu'il y avait en jeu :

1. − l'application cliente
2. − l'application serveur
3. − le client ssh
4. − le serveur ssh

Nous allons voir la redirection locale '−L' et distante '−R'. La différence vient du sens de la connexion. Dans le relayage local, le client tcp et le client ssh sont sur la même machine. Dans le relayage distant ou "remote", le client ssh est sur la même machine que le serveur tcp.

Dans la démarche qui suit, on utilise un client M0 et deux serveurs M1 et M2. On considère que sur chaque serveur fonctionne un serveur ssh. On considère aussi que sur chaque machine on dispose d'un compte d'accès sur chaque machine identique avec les clés publiques installées. Cela évite une authentification à chaque commande ou l'utilisation de l'option '−l' pour préciser le compte à utiliser sur le serveur.

9.4.1. Redirection locale de port (−L Local)

Si on considère ces 3 machines : M0, la machine cliente, M1 et M2, des serveurs. La syntaxe de la commande est la suivante :

```
ssh −L port−local:HOSTNAME:port−distant machine−distante
```

la commande est passée sur M0.

```
M0:$ ssh −L 1234:HOSTNAME:80 M1
```

La commande ouvre une connexion entre le M0:1234 vers HOSTNAME:80 en utilisant le serveur sshd M1:22 (actif sur M1 et sur le port 22). Tout dépend de la valeur que va prendre HOSTNAME. HOSTNAME indique la machine distante sur lequel s'opére la connexion.

Si HOSTNAME est localhost :

```
M0:$ ssh −L 1234:localhost:80 M1
```

Ici localhost indique l'adresse de loopback de la machine distante, c'est à dire ici M1 et sur laquelle tourne le serveur sshd. C'est donc M0:1234 qui est tunnelé vers le port M1:80 en utilisant le service M1:22.

Si HOSTNAME est M1 :

```
M0:$ ssh −L 1234:M1:80 M1 :−/
```

La connexion est identique mais utilisera l'adresse de la M1 (interface réseau) plutôt que l'interface lo.

Si HOSTNAME est M0 :

```
M0:$ ssh −L 1234:M0:80 M1
```

La commande connecte M1 sur M0:80.

Si HOSTNAME est M2 :

```
M0:$ ssh −L 1234:M2:80 M1
```

Il y aura une connexion (tunnel de créé) entre M0 et M1 mais la redirection est effectuée entre M1:1234 et M2:80 en utilisant M1:22. Les transactions sont chiffrées entre M0 et M1, mais pas entre M1 et M2, sauf si un second tunnel ssh est créé entre M1 et M2.

Les redirections de ports ne sont accessibles en théorie que pour les processus locaux (localhost) (pour nous M0). Si la redirection était possible pour des clients (MX ou MY) ou des requêtes distantes qui viendraient se connecter su rM0:1234 dans notre exemple, ces clients pourraient être surpris de se voir reroutés. (Sans parler de risques pour la sécurité car cela signifie que d'autres personnes pourraient utiliser à notre insu le tunnel que nous venons de créer. Prenons un exemple :

```
MO$ ssh −L 1234:M1:80 1M
```

les requêtes locales passées sur M1:1234 sont redirigées vers M1:80, mais des requêtes passées d'autres machines sur M0:1234 ne seront pas, par défaut redirigées.

Il est possible toutefois de passer outre avec l'option '−g' qui autorise des clients distants à se connecter à des ports locaux redirigés.

```
MO$ ssh −g −L 1234:M2:80 M1
```

La commande "lynx http://M0:1234" lancé à partir d'une machine tierce (MX ou MY) redirigera vers M2:80.

Une utilisation de cette fonction sera vue en application un peu plus loin.

9.4.2. Redirection distante de ports (−R Remote)

Ici le client ssh et le serveur TCP sont du même côté. La syntaxe de la commande est la suivante :

```
ssh −R port−distant:HOSTNAME:port−local  machine−distante
```

Le port distant est donc sur la machine distante (remote)

On reprend les mêmes machines que précédemment :

```
M0:$ ssh −R 1234:HOSTNAME:80 M1
```

Ici M0 à le serveur TCP, il sert donc de relai entre une connexion M1:1234 et HOSTNAME:80. La connexion est chiffrée entre M1 et M0. Le chiffrement entre M0 et HOSTNAME dépend de la liaison mise en place.

Si HOSTNAME est localhost, alors on a :

```
M0:$ ssh −R 1234:localhost:80 M1
```

Cela ouvre une connexion depuis M1:1234 vers M0:80 car localhost correspond, ici, à M0. Si un utilisateur passe une requête sur M1:1234, elle sera redirigée vers M0:80.

Si HOSTNAME est M2, on a :

```
M0:$ ssh −R 1234:M2:80 M1
```

Cela ouvre une connexion entre M0 et M1:22, mais les requêtes allant de M1:1234 sont redirigés sur M2:80. Le canal est chiffré entre M1 et M0, mais pas entre M0 et M2.

Enfin dernière remarque, il est possible de passer en paramètre le compte à utiliser sur le serveur qui fait tourner le serveur sshd.

Cette option permet par exemple de donner, à des machines distantes, un accès à un service sur une machine inaccessible autrement.

9.4.3. Schéma de redirection distante de ports

Figure 9−3. Schéma du fonctionnement

9.4.4. Exemple de cas d'utilisation

SSH permet de "tunneler" des protocoles applicatifs via la retransmission de port. Lorsque vous utilisez cette technique, le serveur SSH devient un conduit crypté vers le client SSH.

Cela représente un des plus importants intérêt de SSH. Permettre la création de tunnels "sûrs" pour des protocoles de transports "non sûrs". Ce principe est utilisé pour des applications comme pop, imap, mais également pour des applications X Window.

La retransmission de port mappe (redirige) un port local du client vers un port distant du serveur. SSH permet de mapper (rediriger) tous les ports du serveur vers tous les ports du client.

Pour créer un canal de retransmission de port TCP/IP entre 2 machines qui attend les connexions sur l'hôte local, on utilise la commande suivante :

```
ssh −L port−local:HOSTNAME:port−distant nomutilisateur@nomhôte
```

Une fois que le canal de retransmission de port est en place entre les deux ordinateurs, vous pouvez diriger votre client (POP par exemple) pour qu'il utilise le port local redirigé et non plus vers le port distant avec une transmission en clair.

Nous avons bien vu les options '−L' et '−R'. De nombreuses autres options sont utilisables, par exemple :

```
−L     # Forwarder un port local vers un port distant sur une machine  distante
−R     # Forwarder un port distant vers un port local sur la machine locale
−N     # Ne pas exécuter de commande distante
−f     # Excute le programme en tâche de fond
−l     # Passer en paramètre le login de connexion
−g     # Autoriser des machines distantes à se connecter
       # sur des ports locaux exportés
```

Voyons comment créer un tunnel pour le service d'émulation VT par exemple. On va utiliser un port local compris entre 1024 et 65535 qui sont réservés pour des applications utilisateurs. On va prendre par exemple le port local 1023. Pour créer un tunnel on va utiliser la commande :

```
M0:$  ssh −L1023:localhost:23 mlx@M1.foo.org
```

Ici on utilise (précise) que le commpte utilisé sur la machine distante sera M1.foo.org. Vous devez faire cela si le nom du compte que vous utilisez sur le client diffère de celui que vous utilisez sur le serveur.

On crée ici un tunnel entre le port local 1023 et le port distant 23. Le tunnel étant créé il ne reste plus qu'à utiliser le port local 1023 avec un telnet adresse_de_la_machine 1023

Le fait de quitter ssh ferme la session tunnelée.

Il est également possible d'ouvrir une session temporaire pour un temps donné. Cela évite d'avoir à fermer les connexions. Dans la commande :

```
M0:$  ssh −f  −L1023:localhost:23 mlx@M1.foo.org sleep 10
```

L'option −f, met ssh en tâche de fond. La session sera fermé automatiquement au bout du temps déterminé, seulement si aucun processus n'utilise le canal à ce moment.

Le principe peut être appliqué à d'autres services. Voici comment utiliser un canal sécurisé vers une application (ici un webmail) qui ne l'est pas. (En fait dans la réalité, je vous rassure, l'application présentée sur l'image offre toutes les options de connexions en mode sécurisé :−) L'exemple donné est pris pour illustrer le props)

```
# On crée un tunnel entre le port local et le webmail en utilisant
# le compte mlx@foo.org
M0:$ ssh  −N −f −L 2038:M1:80 mlx@foo.org
```

Figure 9−4. Tunnel HTTP

On peut également sans risque aller sur sa zone public_html en toute sécurité.

```
lynx http://localhost:2038/~mlx
```

La connexion se fait sur la machine locale et sur le port forwardé. Ni le compte utilisateur utilisé, ni le mot de passe ne transitent en clair.

Avec la machine sur lequel le test est réalisé, les commandes :

```
M0:$ ssh  −N −f −L 2038:M1:80 mlx@foo.org
et
M0:$ ssh  −N −f −L 2038:localhost:80 mlx@foo.org
```

ne produisent pas la même chose.

En effet :

```
M0:$ ssh  −N −f −L 2038:M1:80 mlx@foo.org
```

fait référence à une adresse IP qui est redirigée par un Vhost Apache.

```
M0:$ ssh  −N −f −L 2038:localhost:80 mlx@foo.org
```

fait référence à l'adresse de loopback, ici c'est le serveur par défaut (typiquement sur une debian /var/www/) qui est consulté. Si vous avez plusieurs Vhosts, il vous faudra créer un tunnel par Vhost.

De la même façon, on peut l'utiliser pour une session ftp.

```
M0:$ ssh  −N −f −L 2039:M1:21 mlx@foo.org
M0:$ ftp localhost 2039
Connected to localhost.
220 ProFTPD 1.2.5rc1 Server (Debian) [M1]
Name (localhost:mlx): mlx
331 Password required for mlx.
Password:
230 User mlx logged in.
Remote system type is UNIX.
Using binary mode to transfer files.
```

Attention, dans ce dernier exemple, il n'y a que l'authentification qui est chiffrée car le port 20 (ftp−data) n'est pas forwardé.

9.4.5. X and Port Forwarding

Le déport d'application graphique fonctionne sur le même principe. Le client (/etc/ssh/ssh_config), doit avoir l'option "X11Forward" activée pour les systèmes distants. Le serveur (/etc/ssh/sshd_config), doit avoir l'option "X11Forwarding" d'activée. Il est possible de tunneler des applications graphiques dans ssh.

```
# Sur le client /etc/ssh/ssh_config
ForwardX11 yes

# Sur le serveur /etc/ssh/sshd_config
X11Forwarding yes
```

Voyons comment utiliser une application graphique distante :

```
M0:$  ssh −C  mlx@M1.foo.org
```

```
Enter passphrase for key '/home/mlx/.ssh/id_dsa':
M0:$ xclock &
[1] 7771
```

L'horloge distante s'affiche. L'option "−C", active la compression.

Vous pouvez tout mettre sur la même ligne de commande, avec l'option "−f", qui "fork" l'application demandée.

```
M0:$  ssh −C −f  mlx@M1.foo.org xclock
```

Notez le prompt, ici que l'ouverture de session a été effectuée en tâche de fond (−f). Pour fermer le canal ssh, il suffit de fermer l'application.

9.4.6. Automatisation de tâches SSH

Dans la mesure où il est possible de passer des commandes à distances sur un serveur, sans avoir à saisir de mot de passe ou de "passphrase", on peut envisager l'automatisation de tâches entre machines et dans des tunnels, comme par exemple les sauvegardes ou la synchronisation de fichiers. Il suffira pour cela de créer un compte associé à la tâche, lui créer une clé privé et exporter sa clé publique sur les différentes machines.

Attention toutefois, les manipulations sur les serveurs requièrent un accès root. Cela signifie que votre compte opérateur, devra avoir un accès root sur le serveur ce qui n'est jamais très bon.

Vous aurez intérêt à réfléchir à la façon de réaliser le traitement sans que ce compte opérateur ait besoin du compte "super utilisateur".

Si vous avez un serveur ayant un dm (Desktop Manager) comme gdm par exemple, disponible sur le réseau vous pouvez lancer les émulations X Window des clients en appelant le serveur. Par exemple sur le client faire :

```
X −query x.y.z.t :u
ou encore
X −broadcast :u
```

avec u pouvant varier de 0 à 5, suivant que vous voulez avoir la session graphique sur F7...F12. Pour le système vc7 que vous utilisez par défaut avec CTRL ALT F7 correspond au premier "display" :0.

9.5. Scénario d'utilisation d'un proxy ssh

Maintenant que nous y voyons plus clair, voici deux scénarios d'utilisation d'un proxy ssh.

9.5.1. Proxy HTTP

Vous êtes sur un réseau avec un accès extérieur limité. Pas d'accès http :−(

Vous devez disposer d'une machine à l'extérieur sur laquelle tourne un serveur ssh et un proxy Squid par exemple.

Par défaut Squid utilise le port 3128. On met tout d'abord les autorisations à Squid afin qu'il accepte nos requêtes, sinon elles seront rejetés. (Voir les ACL du fichier /etc/squid.conf). Si vous êtes pressés un "http_access allow all" est brutal, mais ça marche ;−) Nous allons créer un tunnel vers ce service de la machine locale vers la machine distante sur le port 3128.

```
ssh  −N −f −L 3128:M1:3128 mlx@M1
```

Il ne reste plus qu'à configurer votre navigateur pour qu'il utilise le proxy local "localhost:3128" et le tour est joué. Vous pouvez naviguer en toute sécurité, si vos requêtes sont espionnées au niveau du firewall de votre boîte, elle ne pourront plus être lues.

Cette configuration présente une limitation. Vous ne pouvez utiliser le proxy qu'à partir de la machine sur laquelle vous avez créé le tunnel (M0).

Si vous êtes mobile dans votre entreprise et que vous souhaitez accéder à l'extérieur à partir d'autres machines, ou encore que vous voulez laisser cet accès à des amis qui utilisent le même réseau que vous, il faut procéder autrement.

```
ssh  −g −N −f −L 3128:M1:3128 mlx@M1
```

L'option "−g" va transformer votre machine en passerelle pour le tunnel. Les autres n'auront plus qu'à mettre comme proxy, le nom ou l'adresse de votre machine et le port 3128.

Si vous souhaitez par contre ouvrir le service de votre proxy à d'autres amis mais pouvant être dans d'autres boîtes ou sur d'autres réseaux, cela se complique un peu, car chacun d'eux doit pouvoir créer un tunnel vers votre proxy. Il faut donc, sur votre proxy créer plusieurs ports d'écoute pour les tunnels et rediriger tout ça vers votre proxy.

```
ssh −N −f −g −L 3130:localhost:3128 mlx@localhost
ssh −N −f −g −L 3131:localhost:3128 mlx@localhost
etc... etc...
```

Ici on a créé sur le proxy 2 ports, 3130 et 3131 qui redirigent vers le port 3128. Il suffit ensuite à partir des machines distantes (réseaux distants) de créer les tunnels vers ces ports.

```
MY$ ssh  -g -N -f -L 3128:localhost:3130 mlx@M1
MZ$ ssh  -g -N -f -L 3128:localhost:3130 mlx@M1
```

et d'utiliser les redirections comme cela a été expliqué plus haut.

Si vous ne souhaitez rediriger les requêtes que pour une machine (site web) en particulier vous pouvez créer un tunnel de la façon suivante :

```
M0:$ ssh  -N -f -L 3128:SITEWEB:3128 mlx@M1
```

où SITEWEB est l'url du site web que vous souhaitez atteindre (typiquement un webmail). Ici, la configuration du navigateur ne change pas. Le proxy est le même. Vous évitez juste l'utilisation d'un squid si vous n'en avez pas.

Remarque, dans le navigateur vous pouvez taper tout ce que vous voulez (yahoo, wanadoo, google...) vous arriverez toujours sur SITEWEB.

9.5.2. Autres scénarios

Si vous avez compris le principe pour le processus décrit ci-dessus, vous pouvez l'appliquer pour tous les autres services (messagerie pop ou imap, news, cvs, vnc ...).

Si un admin faisait trop de rétorsion, ne dites plus rien, vous savez maintenant comment vous y prendre.

9.6. Utilisation de rsync

rsync est un outil largement utilisé pour la synchronisation de répertoires sur la même machine ou sur des machines distantes. (création de mirroirs distants).

rsync est intéressant car il ne metrta à jour sur le "repository" qui reçoit uniquement les fichiers qui ont été créés ou modifiés. Cela procure un gain non négligeable par rapport à une simple "recopie" de toute l'arborescence dans le cas ou peu de fichiers sont modifiés.

rsync et rsyncd, associés à des scripts et à la crontab, est une option remarquable pour la réplication de disques entre 2 ou plusieurs machines.

Si vous souhaitez "mirorrer" un disque local vers un répertoire que vous avez sur une autre machine sur internet, il vous faudra également passer par une procédure d'authentification. Si vous avez exporter votre clé publique sur la machine distante, vous allez pouvoir synchroniser les disques dans un tunnel sécurisé avec ssh et sans avoir à entrer votre mot de passe. Par exemple, la commande :

```
cd & & rsync -e ssh -valptz  * mlx@M1.foo.org
```

synchronisera votre $HOME local, sur le $HOME de la machine distante.

Il existe bien sûr des applications graphiques basées sur le concept rsync.

Une autre option pour la synchronisation de disques distants est "unison". unison est un produit particulièrement performant :

```
http://www.cis.upenn.edu/~bcpierce/unison/index.html
```

unison permet l'utilisation en mode commande (scripts) mais propose aussi une interface graphique.

9.7. Utilisation de SCP et de SFTP

L'utilisation de ces commandes est relativement simple. SCP permet de faire de la copie de fichier. SFTP est utilisable en mode intéractif ou en mode batch et ressemble plus au FTP.

9.7.1. Utilisation de scp

La commande

```
M0:$ ssh mlx@M1.foo.org "ls -al psionic"

-rw-r--r--    1 root     root        64562 Nov 23 23:05 hostsentry-0.02-4.noarch.rpm
-rw-r--r--    1 root     root        26955 Nov 23 23:05 logsentry-1.1.1-1.i386.rpm
-rw-r--r--    1 root     root        48804 Nov 23 23:06 portsentry-1.1-fr7.i386.rpm
-rw-r--r--    1 root     root        48804 Nov 23 23:13 portsentry-1.1-fr7.i386.rpm.1
```

La commande :

```
ssh mlx@M1.foo.org "ls -al"
```

donne la liste des fichiers distants qui sont dans le répertoire "psionic". Pour copier ces fichiers localement dans un répertoire psionic on va utiliser :

```
cd && mkdir psionic
scp mlx@M1.foo.org:/home/mlx/psionic/* ~/psionic
```

Ou pour envoyer les fichiers du répertoire local psionic, vers le répertoire tmp qui est dans /home/mlx de la machine M1.foo.org :

```
scp ~/psionic/* mlx@foo.org:/home/mlx/tmp
```

9.7.2. Utilisation de sftp

sftp peut être utilisé pour du transfert de fichier en mode sécurisé.

```
sftp  mlx@M1.foo.org
sftp>
```

On obtient un prompt, ici le système ne m'a pas demandé de m'authentifier. Pour avoir une liste des commande, utiliser "help".

```
sftp> help
Available commands:
cd path                       Change remote directory to 'path'
lcd path                      Change local directory to 'path'
chgrp grp path                Change group of file 'path' to 'grp'
chmod mode path               Change permissions of file 'path' to 'mode'
chown own path                Change owner of file 'path' to 'own'
help                          Display this help text
get remote-path [local-path]Download file
lls [ls-options [path]]     Display local directory listing
ln oldpath newpath            Symlink remote file
lmkdir path                   Create local directory
lpwd                          Print local working directory
ls [path]                     Display remote directory listing
lumask umask                  Set local umask to 'umask'
mkdir path                    Create remote directory
put local-path [remote-path]Upload file
pwd                           Display remote working directory
exit                          Quit sftp
quit                          Quit sftp
rename oldpath newpath        Rename remote file
rmdir path                    Remove remote directory
rm path                       Delete remote file
symlink oldpath newpath       Symlink remote file
version                       Show SFTP version
!command                      Execute 'command' in local shell
!                             Escape to local shell
?                             Synonym for help
```

9.8. Références

Voir les pages de manuels de ssh et aussi :

1. openssh
2. miscmag

Un article en Français de JPG avec la documentation de sftp en Français:-)

Chapitre 10. Mettre en place un VPN avec PPP et SSH

Approche de SSH, de PPP, des VPN et des services mandataires

10.1. Présentation

Nous allons voir comment mettre en place un réseau virtuel privé qui s'appuie sur le protocole PPP dans un tunel SSH. Cette solution va permettre de mettre en place sur les clients tout type de service mandataire (proxy) pour accéder en toute sécurité à des services serveurs, dans une liaison point à point, et ainsi utiliser des protocoles en toute sécurité, que ceux-ci soient chiffrés ou non.

Si vous ne connaissez pas SSH, je vous recommande de commencer par le document qui aborde le fonctionnement de ce protocole et la mise en place de tunnels sécurisés avec ce produit. Certains aspects qui sont abordés ne seront pas repris ici.

Il sera fait référence à la maquette suivante :

Figure 10−1. Schéma maquette

La machine cliente est sur un réseau privé ou public. Peu importe. Il dispose des applications clientes SSH, PPP, d'un UA (client de messsagerie) et éventuellement un serveur SMTP pour les besoins de la démonstration.

Le serveur est sur une adresse publique. Mettre le serveur sur une adresses privée n'est pas compliquée, mais nécessite de mettre en place des tables de translation d'adresses sur les routeurs. Nous ferons donc sans cela pour ne pas multiplier les problèmes, mais dans la réalité les VPN servent surtout à relier deux ou plusieurs réseaux privés distants, relié par un réseau public ou "non sûr"..

Le serveur dispose de tous les services (dns, smtp, pop3, imap, proxy http(s) et ftp ...) qui serviront aux tests, d'un serveur sshd et d'un serveur pppd pour la création du VPN et du tunnel.

Les routeurs relient deux segments distants via internet ou tout autre type de liaison. Ils peuvent être des pare−feu.

Voici les adresses ethernet affectées aux machines, et les adresses PPP qui seront utilisées pour le VPN :

```
         Client           Serveur
eth0     192.168.90.2     195.115.88.38
ppp0     192.168.0.2      192.168.0.1
```

10.2. Le protocole PPP

Sans trop entrer dans les détails, nous allons faire un petit tour du protocole PPP puisqu'il en est question dans ce document.

Le protocole PPP (Point to Point Protocole) est un protocole de niveau 2. Il supporte des liaisons point−à point synchrones ou asynchrones.

PPP est composé de trois grandes entités :

1. Un protocole qui servira à l'encapsulation des paquets avant dépôt sur le média physique : HDLC. HDLC est un protocole de liaison de données.
2. LCP (Link Control Protocol) qui sert à établir (ou rompre) la liaison, qui permet de la tester et de la configurer.
3. NCP (Network Control Protocol) qui sert d'interface pour les protocoles de niveau 3. Il n'y a pas un (1) protocole NCP, mais "n". En effet, chaque protocole de niveau 3 dispose de sa propre interface particulière. Sur ip, l'implémentation NCP se nomme IPCP (IP Control Protocol).

Un paquet PPP peut véhiculer plusieurs protocoles de niveau 2 ayant chacun un rôle, ou des paquets contenant des données de niveau 3. Voici quelques exemples avec les numéros de protocoles.

1. h021, indique un transport de données IP
2. 8021, IPCP. Ce protocole permet de déterminer les adresses de la liaison point à point entre les deux noeuds distants. Affectation statique ou dynamiquedes adresses, compression des entêtes IP...
3. C021, paquet contenant des données LCP
4. C023, paquets de type PAP, Password Authentification Protocole
5. C223, paquet sde type CHAP, Challenge Handshake Authentification.

Les numéros de protocoles sont stocké dans un champ "contrôle" du paquet PPP. PAP et CHAP servent à l'authentification des utilisateurs.

10.3. Configuration et installation du VPN

Préparation du client et du serveur.

10.3.1. Première étape : configuration de SSH

La première chose à faire est de mettre en place un moyen de lancer le daemon pppd chaque fois que vous voudrez mettre en place un VPN entre votre client et le serveur. Il y deux solutions. Soit vous créez un compte spécifique sur le serveur (ce que nous allons faire), et qui aura en charge de lancer le daemon, soit vous utilisez votre propre compte. Dans un cas comme dans l'autre, la procédure est très similaire.

Sur le serveur, créez un compte VPN :

```
adduser −−disabled−password vpn
```

On donne le droit à cet utilisateur de lancer pppd. Il faut rajouter une ligne dans le fichier.

```
serveur# visudo /etc/sudoers
# Ajoutez la ligne suivante
```

```
vpn     ALL=NOPASSWD:/usr/sbin/pppd
```

Comme la liaison sera chiffrée dans un tunnel SSH, il est nécessaire de mettre également un moyen qui permette cela le plus simplement possible. Cela est réalisé par un système de clé publique et privée. Si vous n'en avez pas vous pouvez vous en créer une. Ne mettez pas de mot de passphrase (vous faites entrée, ce n'est pas vraiment utile. Vous êtes sur le client :

```
[client]$ ssh-keygen -t dsa
Generating public/private dsa key pair.
Enter file in which to save the key (/client/.ssh/id_dsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /client/.ssh/id_dsa.
Your public key has been saved in /client/.ssh/id_dsa.pub.
The key fingerprint is:
c5:a5:90:b2:19:9d:bf:3a:0b:c9:64:f7:98:1e:e0:dc client@mr
```

Vous avez la clé publique et la clé privée sur le client, dans votre répertoire personnel et dans le sous-répertoire ".ssh". Il vous faut copier la clé publique sur le serveur.

```
scp .ssh/id_dsa.pub VotreCompteSurLeServeur@serveur:
```

Maintenant il ne reste plus qu'à déclarer cette clé publique comme valide et utilisable par le compte "VPN". Vous êtes sur le serveur.

```
serveur# mkdir -p /home/vpn/.ssh
serveur# cat /home/VotreCompteSurLeServeur/id_dsa.pub >> \
         /home/vpn/.ssh/authorized_keys2
serveur# chmod 700 /home/vpn/.ssh &&  chmod 600 /home/vpn/.ssh/authorized_keys2
```

Normalement vous devriez pouvoir vous connecter à partir du client sur le serveur en utilisant le compte VPN sans entrer de mot de passe.

```
[client]$ ssh -l vpn serveur
```

Si cela ne fonctionne pas, et que le serveur sshd est actif, vérifiez que vous n'avez pas commis d'erreur de manipulation. Reprenez bien la procédure.

Si le serveur vous demande un mot de passe et que l'accès fonctionne en mettant un mot de passe, c'est que vous avez saisi une "passphrase". Utilisez ssh-agent et ssh-add pour ne plus avoir à sasir de mot de passe.

Si tout fonctionne, c'est terminé.

Si d'autres personnes veulent mettre en place des VPN sur le serveur, il suffit de rajouter leurs clés publique dans le fichier authorized_keys2.

Si vous ne voulez pas utiliser de compte "VPN" ou autre mais le vôtre, il suffit de modifier le fichier sudoers en mettant votre compte à la place de "vpn".

10.3.2. Test de la connexion

Il faut maintenant pouvoir activer et/ou désactiver le VPN à la demande. Nous allons pour cela, utiliser un script que vous pourrez adapter.

```
#!/bin/sh
# /usr/local/bin/vpn-pppssh
#
# This script initiates a ppp-ssh vpn connection.
# see the VPN PPP-SSH HOWTO on http://www.linuxdoc.org for more information.
#
# revision history:
# 1.6 11-Nov-1996 miquels@cistron.nl
# 1.7 20-Dec-1999 bart@jukie.net
# 2.0 16-May-2001 bronson@trestle.com


#
# You will need to change these variables...
#


# The host name or IP address of the SSH server that we are
# sending the connection request to:
SERVER_HOSTNAME=serveur.votredomaine.org

# The username on the VPN server that will run the tunnel.
# For security reasons, this should NOT be root.  (Any user
# that can use PPP can intitiate the connection on the client)
SERVER_USERNAME=vpn

# The VPN network interface on the server should use this address:
SERVER_IFIPADDR=192.168.0.1
```

```
# ...and on the client, this address:
CLIENT_IFIPADDR=192.168.0.2


# This tells SSH to use unprivileged high ports, even though it's
# running as root.  This way, you don't have to punch custom holes
# through your firewall.
LOCAL_SSH_OPTS="-P"


#
# The rest of this file should not need to be changed.
#



PATH=/usr/local/sbin:/sbin:/bin:/usr/sbin:/usr/bin:/usr/bin/X11/:

#
# required commands...
#

PPPD=/usr/sbin/pppd
SSH=/usr/bin/ssh

if ! test -f $PPPD  ; then echo "can't find $PPPD";  exit 3; fi
if ! test -f $SSH   ; then echo "can't find $SSH";   exit 4; fi


case "$1" in
  start)
    echo -n "Starting vpn to $SERVER_HOSTNAME: "
# Modifier les 3 lignes ci-dessous afin d'ôter les \ et les CR/LF
    ${PPPD} updetach noauth passive pty "${SSH} ${LOCAL_SSH_OPTS}\
    ${SERVER_HOSTNAME} -l${SERVER_USERNAME} sudo ${PPPD} nodetach\
    notty noauth" ipparam vpn ${CLIENT_IFIPADDR}:${SERVER_IFIPADDR}
    echo "connected."
    ;;

  stop)
        # echo -n "Stopping vpn to $SERVER_HOSTNAME: "
# Modifier les 2 lignes ci-dessous afin d'ôter les \ et les CR/LF
        PID=`ps ax | grep "${SSH} ${LOCAL_SSH_OPTS}  ${SERVER_HOSTNAME}\
            -l${SERVER_USERNAME}"| grep -v ' passive ' | grep -v 'grep '\
            | awk '{print $1}'`
        if [ "${PID}" != "" ]; then
          kill $PID
          echo "Kill $PID, disconnected."
        else
          echo "Failed to find PID for the connection"
        fi
    ;;

  config)
    echo "SERVER_HOSTNAME=$SERVER_HOSTNAME"
    echo "SERVER_USERNAME=$SERVER_USERNAME"
    echo "SERVER_IFIPADDR=$SERVER_IFIPADDR"
    echo "CLIENT_IFIPADDR=$CLIENT_IFIPADDR"
  ;;

  *)
    echo "Usage: vpn {start|stop|config}"
    exit 1
    ;;
esac

exit 0
```

Pour l'utiliser, il suffit de faire un "./vpn start". Si vous obtenez quelque chose proche de cela, c'est que votre vpn est bien créé.

```
[root]# ./vpn-pppssh start
Starting vpn to serveur: Using interface ppp0
Connect: ppp0 <--> /dev/pts/1
Looking for secret in /etc/ppp/pap-secrets for client mr server (null)
Looking for secret in /etc/ppp/chap-secrets for client mr server (null)
Deflate (15) compression enabled
local  IP address 192.168.0.2
remote IP address 192.168.0.1
connected.
```

Sur le client et sur le serveur, les interfaces ppp0 doivent être actives :

```
# Sur le client, ifconfig
ppp0      Lien encap:Protocole Point-à-Point
          inet adr:192.168.0.2  P-t-P:192.168.0.1  Masque:255.255.255.255
          UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1500  Metric:1
```

```
          RX packets:126 errors:0 dropped:0 overruns:0 frame:0
          TX packets:102 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 lg file transmission:3
          RX bytes:10139 (9.9 Kb)  TX bytes:9029 (8.8 Kb)

# Sur le serveur, ifconfig
ppp0      Lien encap:Protocole Point−à−Point
          inet adr:192.168.0.1  P−t−P:192.168.0.2  Masque:255.255.255.255
          UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1500  Metric:1
          RX packets:102 errors:0 dropped:0 overruns:0 frame:0
          TX packets:126 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 lg file transmission:3
          RX bytes:9029 (8.8 KiB)  TX bytes:10139 (9.9 KiB)
```

Le protocole a installé les routes de ces interfaces et qui viennent compléter celles déjà existantes :

```
# Sur le client :
[Client]# route −n
Table de routage IP du noyau
Destination     Passerelle      Genmask         Indic Metric Ref    Use Iface
192.168.0.1     0.0.0.0         255.255.255.255 UH    0      0      0 ppp0


# Sur le serveur
Table de routage IP du noyau
Destination     Passerelle      Genmask         Indic Metric Ref    Use Iface
192.168.0.2     0.0.0.0         255.255.255.255 UH    0      0       0 ppp0
```

Sur le client, les commandes :

```
ping 192.168.0.1
ping 192.168.0.2
```

fonctionnent. Le routage est donc bien actif entre ces interfaces, nous allons pouvoir faire fonctionner des applications clientes et serveur sur ce canal.

10.4. Explication sur le fonctionnement de la maquette

Figure 10−2. Schéma du dialogue

Sur le client, les applications sont configurées pour utiliser l'adresse ip affectée à ppp0. Le dialogue peut être effectué sans chiffrement. Le paquet est délégué à SSH qui va chiffrer puis déposer le paquet sur eth0.

Sur le serveur les paquets arrivent chiffrés sur eth0. Ils sont déchiffrés par SSH et routés sur l'interface ppp0 du serveur qui les délivre au service serveur.

Cela présente bien sûr un inconvénient. Celui de charger les processeurs et la bande passante bien que PPP supporte la compression des données. Il n'est pas toujours facile de trouver un bon compromis entre des options de sécurité par chiffrement tout en tenant compte des coûts que cela induit.

Figure 10−3. Encapsulation des trames

Le schéma montre que chaque paquet subit deux traitements de plus que s'il était traité normalement pour l'émission, ce qui en génère également deux de plus lors de la réception.

10.5. L'analyse de trame

L'objet de cette manipulation est de vérifier simplement le bon fonctionnement du vpn à l'aide d'une requête DNS (dig www.yahoo.fr) entre le client et le serveur. La capture intercepte ce qui se passe sur eth0 et ppp0.

Le fichier resolv.conf est configuré pour utiliser le serveur de nom :

```
[client]# more /etc/resolv.conf
nameserver 195.115.88.38
```

Les requêtes passent dans ce cas par eth0.

Sur ppp0 rien ne passe, sur eth0 par contre, il y a du traffic.

```
[client]# ettercap -T -i   eth0
Listening on eth0... (Ethernet)
  eth0 ->       00:90:F5:28:D5:06      192.168.90.2     255.255.255.0
Mon Nov 22 10:11:59 2004
 UDP  192.168.90.2:32885 --> 195.115.88.38:53 |
 .}...........www.yahoo.com.....
```

Il s'agit bien d'une requête UDP qui est passée sur ETH0, en utilisant comme source 192.168.90.2:32885 et à destination de 195.115.88.38:53. Le traffic n'est pas chiffré.

On va modifier le resolv.conf afin que les requêtes passent par le VPN.

```
nameserver 192.168.0.1
```

Et on lance la capture sur ppp0 :

```
[client]# ettercap -T -i   ppp0
  ppp0 ->       00:00:00:00:00:00       192.168.0.2   255.255.255.255

Mon Nov 22 10:24:04 2004
 UDP  192.168.0.2:32885 --> 192.168.0.1:53 |
 iF...........www.yahoo.com.....
Mon Nov 22 10:24:04 2004
 UDP  192.168.0.1:53 --> 192.168.0.2:32885 |
```

Maintenant on a bien un traffic sur ppp0. Noter qu'elle n'a pas d'adresse MAC car c'est une interface logique. Il s'agit bien d'une requête UDP de l'hôte source 192.168.0.2:3288 vers l'hôte destination 192.168.0.1:53. Ici le traffic n'est pas chiffré.

On la lance également sur eth0 :

```
[client]# ettercap -T -i   eth0
Listening on eth0... (Ethernet)
  eth0 ->       00:90:F5:28:D5:06      192.168.90.2     255.255.255.0

Mon Nov 22 10:20:39 2004
TCP  195.115.88.38:22 --> 192.168.90.2:33102 | AP
..!..O.R$.[NP...<G.O._uRH?O."..M36<.Gd.?..      ..Vx.
Mon Nov 22 10:20:39 2004
TCP  192.168.90.2:33102 --> 195.115.88.38:22 | AP
8..1.xV....]....D..q.7:3y.%.&.J..2..8.Qp%.*.."..
```

Là le dialogue n'est plus dirigé vers de l'UDP/53 mais vers du TCP/22. Il s'agit bien de SSH et plus rien n'est lisible.

La maquette fonctionne parfaitement, le service mis en place est le premier service proxy actif. Il s'agit d'un service proxy DNS.

10.6. Les services pop, imap et smtp

Pour les services pop et imap il n'y a aucun difficulté. Il suffit de configurer le client pour qu'il relève les courriers sur 192.168.0.1.

En ce qui concerne SMTP il y a deux scénarios posibles :

1. utiliser directement le serveur smtp qui est sur le serveur,
2. utiliser un serveur smtp qui est sur sa propre machine comme cela se fait fréquemment.

Dans le premier cas, la procédure est assez simple. Dans la configuration de votre client de messagerie (paramètre envoie de message), vous mettez 192.168.0.1, c'est à dire l'adresse du serveur.

Si vous utilisez votre propre serveur SMTP local sur le client, il faut mettre l'adresse du client : 192.168.0.2.

Mais cela ne suffit pas. Par défaut, si vous ne précisez rien, un serveur SMTP envoie le message directement au destinataire. C'est la fonction de routage du service SMTP qui assure cela. Il se base sur l'enregistrement MX de la zone destinataire. Pour faire cela, il faut configurer le serveur SMTP pour qu'il utilise l'interface eth0 et non pas ppp0. La configuration du client de messagerie contient, dans ce cas, l'adresse ip de eth0, mais en faisant cela, on utilise plus du tout le VPN.

Pour utiliser le VPN mis en place, il faut indiquer au serveur SMTP local de ne pas envoyer directement les messages sur internet vers le destinataire, mais de les faire relayer par le serveur smtp du serveur.

Comment configurer cela ?

On va prendre comme exemple Postfix.

Sur le client, il faut ajouter un paramètre dasn /etc/postfix/main.cf :

```
relayhost = 192.168.0.1 # On de mande au s"erveur de relayer
```

pour indiquer au serveur smtp local, que les messages seront relayés par le serveur d'adresse 192.168.0.1.

Maintenant le serveur. Il faut lui indiquer qu'il doit accepter et traiter les paquets provenant de l'adresse 192.168.0.2, voire carrément d'un réseau 192.168.0.0/24 car postfix est généralement configuré pour ne pas relayer les messages qui ne proviennent pas de son réseau.

Sur le client il faut rajouter un paramètre dans /etc/postfix/main.cf, et ajouter à la variable "mynetworks", l'adresse du VPN :

```
# Vous pouvez en avoir d'autres.
mynetworks = 127.0.0.1/32, 192.168.0.0/24
```

à partir de ce moment, il est possible d'utiliser son service SMTP local pour ne pas changer ses habitudes, cela, en toute sécurité quel que soit l'endroit où on se trouvons. (Du moins entre les deux points du VPN) et avec l'énorme avantage d'utiliser un serveur SMTP déclaré ou officiel qui répond au contraintes de plus en plus draconniennes que mettent en place les FAI et les administrateurs systèmes.

10.7. Les services HTTP(s) et FTP

La mise en place d'un service proxy squid, ne pose pas non plus de difficulté une fois le VPN installé. Il doit falloir moins de 3 minutes pour installer Squid sur une debian. Ensuite il faut juste lui indiquer les machines dont il doit traiter les requêtes.

Comme le service sera installé sur un serveur public, on fera attention aux autorisations données afin de limiter les risques. Cela se passe dans le fichier de configuration "/etc/squid.conf" et dans le paragraphe qui décrit les ACL.

On déclare une nouvelle ACL, et on autorise les machines qui répondent à ce critère :

```
acl vpn src 192.168.0.0/255.255.255.0
http_access allow vpn
```

Pour le serveur c'est terminé.

Il ne reste plus qu'à configurer les clients d'accès (navigateur) pour qu'ils utilisent le proxy : 192.168.0.1.

Là encore toutes les transactions entre le client et le serveur seront chiffrées, et on bénéficie en plus d'un serveur de cache.

10.8. Conclusion

Cette solution est assez simple à mettre en place. Elle est particulièrement intéressante pour les clients nomades. Ils peuvent, quelque soit l'endroit où ils se trouvent, mettre en place un VPN entre le client et un serveur, sans avoir à se soucier de l'état du réseau sur lequel ils se trouvent, et peuvent utiliser quasiment tous les services réseau sans se préoccuper des règles installées sur les parefeu, puisque tous les services sont routés dans le même tunnel.

10.9. Références et annexes

1. nst
2. tldp.org
3. linux_network
4. linuxsecurity

Chapitre 11. Les fichiers `hosts`

Résolution de noms et adressage statique

Résolution de noms et adressage statique

Mettre en place et comprendre la résolution de noms d'hôtes.

11.1. Présentation

L'atelier présente la résolution de noms d'hôtes sur un petit réseau à l'aide d'un fichier `hosts`.

Vous utiliserez la commande **ping** pour diagnostiquer le fonctionnement du réseau.

Il est en 3 parties :

- une présentation de la résolution de nom sur un réseau local,
- un TP,
- un questionnaire.

11.1.1. Avant de démarrer

Vous devez connaître la classe d'adresse de votre réseau. Vous devez connaître également les adresses des hôtes que vous voulez adresser ainsi que leurs noms d'hôtes.

11.1.2. Fiche de cours

Dans un réseau, on assigne généralement un nom à chaque hôte. Le terme d'hôte est pris dans son sens large, c'est à dire un << noeud de réseau >>. Une imprimante, un routeur, un serveur, un poste de travail sont des noeuds qui peuvent avoir un nom d'hôte, s'ils ont une adresse IP.

On parle de << nom d'hôte >> sur les réseaux qui utilisent le protocole TCP/IP. Ne pas confondre le << nom d'hôte >> avec le << nom Netbios >> qui est utilisé sur les réseaux Microsoft" ou IBM.

Le nom d'hôte est assigné à un noeud qui est configuré avec une adresse IP. Le nom permet d'adresser le noeud, autrement qu'avec l'adresse IP. Par exemple, si le réseau est équipé d'un serveur d'adresse 192.68.0.100 et dont le nom d'hôte est `srv1`, il sera alors possible de saisir les commandes suivantes :

telnet *`192.68.0.100`* ou bien

telnet *`srv1`*

Le nom sert de mnémonique, qui évite de retenir toutes les adresses IP du réseau. Le protocole TCP/IP se charge de la résolution des noms d'hôtes, ensuite le protocole ARP, se charge de la résolution des adresses IP en adresses MAC (Ethernet le plus souvent).

Pour que la résolution de nom fonctionne, il faut déclarer dans un fichier tous les noms d'hôtes, et pour chaque nom, son adresse IP. Cette déclaration est réalisée dans le fichier `/etc/hosts`.

> Le processus de résolution équivalent peut être mis en oeuvre sur des réseaux qui utilisent Windows 9x, Windows NT Server, Windows NT Workstation. Vous devrez alors créer les fichiers respectivement dans les répertoires `Windows` et `winnt\system32\drivers\etc`. Vous trouverez dans ces répertoires, si TCP/IP est installé un fichier `host.sam` qui peut vous servir d'exemple

11.2. Travaux Pratiques

Vous utiliserez un éditeur joe ou emacs afin de modifier le fichier `/etc/hosts`. Utilisez l'algorithme suivant pour créer / modifier votre fichier :

```
Pour chaque hôte du réseau faire
   mettre un enregistrement
Fin pour
```

Les enregistrements ont la structure suivante : AdresseIP Nom1 [...NomN]

Exemple : 195.115.88.35 foo foo.foo.org becassine

Consultez également la commande **`man *hosts*`**

- Établissez la nomenclature des machines du réseau. Configurez le fichier `host` avec la nomenclature. Testez la résolution de nom avec la commande **ping**, puis en utilisant les services telnet et ftp.
- Modifiez la correspondance Nom / Adresse d'une des machines que vous avez dans votre fichier `host` et accédez−y avec telnet. Que se passe−t−il ?
- Débranchez la jarretière de votre carte réseau, et réutilisez les commandes **`ping *localhost*`**, **`ping *127.0.0.1*`**, **`ping *UneMachineDistante*`**. Que se passe−t−il et que peut−on en déduire ?

Attention, plus tard nous verrons la résolution de nom par un autre service DNS. Les deux solutions (DNS et Hosts) ne sont pas exclusives, par contre on peut jouer sur l'ordre qui doit être appliqué. Cela est traité par le fichier de configuration `/etc/host.conf`.

```
$ more /etc/host.conf
order hosts,bind
```

Il faut bien se souvenir de ça, car dans l'exemple donné ci−dessus, le fichier `hosts` est prioritaire sur DNS.

11.3. Questions

- Quelle est la commande qui permet d'obtenir le nom d'hôte de la machine locale ?
- Quelles sont les informations que donne la commande **ifconfig** ?
- Donnez la commande qui permet de n'envoyer qu'un seul ping à une machine distante (voir **man *ping***)
- Quelle est la taille d'un paquet envoyé par la commande **ping** ?
- Quelle est la commande qui permet d'envoyer des paquets de 1500 octets ?
- Quelle est la commande **ping** qui permet d'envoyer des paquets en flot ininterrompu ?
- Quel protocole utilise la commande **ping** ?

Chapitre 12. Installation d'un serveur HTTP

Les services web − Fiche de cours

Les services web − Fiche de cours

Configuration d'un serveur Apache et mise en place de services web

Vous pourrez récupérer les documents nécessaires sous forme d'archives sur le serveur de linux−france. Pour cela voir la page d'introduction du document.

12.1. Résumé

Installation et configuration d'un serveur HTTP avec Apache.

Mots clés << Serveur Web >>, << Serveur HTTP >>, << Apache >>

Description et objectifs de la séquence

Le document doit vous permettre de mettre en place un serveur Apache supportant :

- des accès anonymes,
- des accès authentifiés par Apache,
- un accès à des pages personnelles,
- mettre en place des scripts CGI,
- mettre en place des serveurs web virtuels.

12.2. Présentation du serveur Apache

Ce chapitre donne un aperçu des fonctions et de l'environnement du serveur Apache. Vous pourrez retrouver tous les aspects développés dans la documentation du produit.

Il existe des outils graphiques de configuration et d'administration d'Apache. Vous allez réaliser les TP(s) de cet atelier en mode commande.

12.2.1. Présentation de l'environnement

- le binaire **apachectl** est dans `/usr/sbin`,
- les fichiers de configuration sont dans `/etc/apache/`
- la documentation est dans `/usr/share/doc`
- Le script de lancement du service serveur dans `/etc/init.d`
- Les journaux sont dans `/var/log/apache/`

Faites une copie de sauvegarde des fichiers de configuration avant toute manipulation.

12.2.2. Installation d'un service minimum

Ce paragraphe décrit les principaux paramètres pour mettre en place un service HTTP minimum, avant de lancer le service serveur. Vous utiliserez le fichier de configuration d'Apache `httpd.conf`.

- port 80, indique quel est le port utilisé par le service (par défaut 80). Il est possible d'utiliser un autre port, par contre vous devrez spécifier au navigateur quel est le port utilisé par le serveur. Si vous configurez par exemple le port 8080 sur une machine `www.MonDomaine.edu`, vous devrez spécifier dans le navigateur `www.MonDomaine.edu:8080`, pour que le serveur reçoive et traite votre requête.
- user www−data et group www−data, spécifient le compte anonyme utilisé par le serveur une fois qu'il est lancé. En effet, pour accéder aux ports inférieurs à 1024, le serveur utilise un compte administrateur, ce qui présente des dangers. Une fois le processus actif, il utilisera l'UID d'un autre compte (ici nobody). Ce compte doit pouvoir lire les fichiers de configuration et ceux de la racine du serveur HTTP. D'autres distributions utilisent le compte << nobody >> ou << apache >>

- ServerAdmin root@localhost, précise quel est le compte qui reçoit les messages. Par défaut le compte administrateur sur la machine locale (à modifier pour une adresse comme `root@MonDomaine.edu`).
- ServerRoot `/etc/apache`, indique l'adresse du répertoire racine du serveur, où sont stockés les fichiers de configuration du serveur HTTP. Cette adresse peut être modifiée.
- ErrorLog, fichier `error_log`, journalisation des erreurs. L'adresse est calculée à partir de ServerRoot. Si ServerRoot est `/etc/httpd` et ErrorLog logs/error_log, le chemin complet est `/var/log/apache/logs/error_log`.
- ServerName www.MonDomaine.edu, indique le nom ou l'alias avec lequel la machine est désignée. Par exemple, l'hôte `ns1.MonDomaine.edu`, peut avoir le nom d'alias `www.MonDomaine.edu`. Voir la résolution de nom avec un DNS.
- DocumentRoot /var/www, indique l'emplacement par défaut des pages HTML quand une requête accède au serveur. (exemple : la requête http://www.MonDomaine.edu/index.html pointe en fait sur le fichier local /home/httpd/html/index.html).
- ScriptAlias /cgi−bin/ /usr/lib/cgi−bin, de la forme "ScriptAlias FakeName RealName", indique où sont physiquement situés les scripts sur le disque, ainsi que l'alias utilisé par les développeurs pour le développement des scripts et des pages. Un développeur utilisera un lien (exemple : /cgi−bin/NomDuScript où /cgi−bin/ est un alias sur /home/httpd/cgi−bin/), et c'est le script /home/httpd/cgi−bin/NomDuScript qui sera effectivement exécuté. La mise en place d'alias permet de restructurer ou déplacer un serveur sans avoir à modifier toutes les pages développées.
- UserDir public_html, ce paramètre décrit le processus utilisé pour accéder aux pages personnelles d'une personne, si ces pages sont stockées dans son répertoire personnel. Supposons que vous êtes l'utilisateur "bestof" du réseau et que vous ayez des pages personnelles. Il sera possible d'accéder à vos pages, avec l'adresse suivante: www.MonDomaine.edu/~bestof/index.html. Le (tilde "~") permet d'accéder à votre répertoire personnel. La requête sera réellement exécutée sur "/home/bestof/public_html/index.html.

  Attention, vérifier que le répertoire personnel ne soit pas en mode 700, car personne ne pourrait accéder aux pages personnelles.
- Alias /CheminVu/ /CheminRéel/, par exemple : "/icons/ /usr/share/apache/icons/", ce paramètre permet de renommer, à la manière d'un lien logique, un emplacement physique avec un nom logique.

  Exemple: vous voulez que www.MonDomaine.edu/test/index.html, ne corresponde pas physiquement à un répertoire sur la racine du serveur HTTP mais à un emplacement qui serait /usr/local/essai. Vous pouvez mettre dans le fichier de configuration d'Apache un alias de la forme: alias /test/ /usr/local/essai/
- DirectoryIndex donne le ou les noms des fichiers que le serveur doit rechercher si le navigateur passe une requête sur un répertoire. Par exemple sur une requête http://www.MonDomaine.edu, le serveur va rechercher dans l'ordre s'il trouve un fichier index.html, index.stml, index.cgi... en fonction des paramètres de cette variable.
- Les fichiers .htaccess : Apache permet de sécuriser les accès répertoire par répertoire. Il est possible de définir, le nom du fichier qui contiendra les possibilités d'accès par un utilisateur à un répertoire donné. Par défaut la valeur est .htaccess. Ce paramètre est modifiable.
- Limitations de la sécurité par répertoire: ce procédé alourdit la charge du serveur. En effet, si une requête est passée sur www.MonDomaine.edu/rep1/rep2/index.html, le serveur va vérifier dans chaque répertoire rep1, rep2... l'existence d'un fichier .htaccess. Ce sont les règles du dernier fichier qui seront appliquées. Ce processus est mis en oeuvre pour chaque accès. Cette directive est donc à utiliser avec beaucoup de parcimonie car elle crée une surcharge pour le serveur.

  La directive AllowOverride None, permet de désactiver l'utilisation des fichiers .htaccess dans les niveaux inférieurs. La directive AllowOverride peut être utilisée avec d'autres options par exemple: AuthConfig.

  Les fichiers .htaccess peuvent, s'ils sont présents spécifier leurs propres directives d'authentification,

  La directive ExecCGI, permet l'exécution de scripts cgi dans ce répertoire.
- Sécuriser un répertoire en autorisant/refusant l'accès

  Pour chaque répertoire "UnRépertoire", sur lequel on désire avoir une action, on utilisera la procédure suivante:

```
<Directory UnRépertoire>
...Ici mettre les actions...
</Directory>
```

  Tout ce qui est entre les balises s'applique au répertoire "UnRépertoire".

  Exemple: On désire autoriser l'accès du répertoire "/intranet" aux machines du réseau d'adresse 192.168.1.0/24 et de nom de domaine MonDomaine.edu, et l'interdire à tous les autres.

```
<Directory /intranet>
#Ordre de lecture des règles
order allow,deny
deny from all
allow from 192.168.1 #ou encore allow from .MonDomaine.edu
</Directory>
```

  Il importe de préciser dans quel ordre les règles de restriction vont être appliquées. Cet ordre est indiqué par le mot réservé << order >>, par exemple << order deny,allow >> (On refuse puis on alloue l'accès à quelques adresses, c'est à dire que toutes les régles deny vont être lues d'abord, puis ce sera le tour de toutes les règles allow) ou << order allow,deny >> (on accepte généralement les accès mais il sont refusés pour quelques adresses : ici, on prend en compte en premier lieu toutes les règles allow dans l'ordre trouvé, puis ensuite toutes les règles deny).

  Exemple: On désire que l'accès soit majoritairement accepté, sauf pour un ou quelques sites.

```
<directory /home/httpd/html>
AllowOverride none
Order deny,allow
deny from pirate.com badboy.com cochon.com
allow from all
</directory>
```

- Authentifier l'accès à un répertoire : ce procédé va permettre de sécuriser l'accès à un répertoire ou à des fichiers. L'accès sera autorisé à une ou plusieurs personnes ou encore à un ou plusieurs groupes de personnes.

  AuthName, définit ce qui sera affiché au client pour lui demander son nom et son mot de passe,

  AuthType, définit le mode d'authentification et d'encryptage << basic >> avec HTTP/0 ou << MD5 >> par exemple avec HTTP/1.

  AuthUserFile, définit le fichier qui contient la liste des utilisateurs et des mots de passe. Ce fichier contient deux champs (Nom d'utilisateur, Mot de passe crypté). Vous pouvez créer ce fichier à partir du fichier /etc/passwd (attention ! faille de sécurité. Il n'est pas forcément avisé d'avoir le même mot de passe pour accéder à Linux et pour accéder à un dossier Web) ou avec la commande "htpasswd" d'Apache.

  Exemple du mode d'utilisation de la commande "htpasswd" :

```
root@mr:/home# htpasswd −−help
        htpasswd [−cmdps] passwordfile username
 −c  Create a new file.

#> htpasswd −c /etc/apache/users mlx
Ici on crée le fichier /etc/apache/users et on ajoute un compte.
N'utiliser l'option "−c" que la première fois.
```

  AuthGroupFile définit le fichier qui contient la liste des groupes et la liste des membres de chaque groupe,

  Require permet de définir quelles personnes, groupes ou listes de groupes ont une permission d'accès.

  Exemple de fichier AuthUserFile :

```
doudou:zrag FmlkSsSjhaz
didon:PsddKfdqhgf.fLTER
```

  Exemple de fichier AuthGroupFile :

```
users: tintin milou haddock dupond dupont tournesol
tournesol dupont
```

  Exemple d'autorisation :

```
require user tintin dupond /* tintin et dupond ont un accès */
require group users /* le groupe users à un accès */
require valid−user /* toute personne existant dans AuthUserFile */
```

  Exemple d'accès sécurisé sur un répertoire :

```
<Directory /home/httpd/html/intranet/>
AuthName PatteBlanche
AuthType basic
AuthUserFile /etc/httpd/conf/users
AutGroupFile /etc/httpd/conf/group
     <Limit GET POST>#Ici il faudra un mot de passe
        require valid−user
     </Limit>
</Directory>
```

  Voici la fenêtre sécurisée que propose Netscape sur l'URL http://localhost/essai:

  Figure 12−1. Accés sécurisé sur un répertoire par Apache

La déclaration d'un accès authentifié sur un répertoire est faite dans le fichier de configuration d'Apache, ou alors en créant un fichier ".htaccess" dans le répertoire que l'on souhaite sécuriser. Le fichier ".htaccess" contient les mêmes directives que celles qui auraient été déclarées dans le fichier httpd.conf.

Attention :

```
Si vous mettez les clauses d'accès restreint pour un
répertoire dans le fichier de configuration d'Apache, les clauses
seront incluses entre 2 balises :
<Directory ...>
</Directory ...>

Si vous mettez les clauses de restriction dans un fichier ".htaccess",
```

```
vous n'avez pas besoin de mettre ces balises.
```

12.2.3. Activation du serveur

Utilisez les commandes suivantes pour activer, désactiver le serveur:

/etc/init.d/apache start

/etc/init.d/apache stop

/etc/rc.d/init.d/apache status

Pour relire le fichier de configuration alors qu'apache est déjà lancé, utilisez :

/etc/init.d/apache reload

Pensez dans tous les cas à consulter les journaux afin de détecter les dysfonctionnements.

12.2.4. Test de la configuration

Lancez le navigateur et tapez l'url http://localhost. Vous devriez pouvoir utiliser indifféremment l'adresse IP ou le nom d'hôte de votre machine. Vous devez également pouvoir accéder à partir des autres machines de la salle.

12.3. Questions

- Quel protocole et quel port utilise le serveur Apache ?
- Comment se nomme le principal fichier de configuration d'Apache, et où se trouve−t−il ?
- Dans quel répertoire sont situées les pages du serveur ?
- Vous modifiez le port d'utilisation du serveur et vous faites un essai à partir d'un client. L'accès ne fonctionne pas. Donnez au moins deux raisons possibles et les moyens de remédier à ce problème.
- Quel est le paramètre qui permet l'utilisation de répertoires personnels pour les utilisateurs afin de déployer leurs sites WEB personnels ?
- Vous activez le paramètre du répertoire personnel dans Apache et relancez le serveur. Vous essayez l'accès sur votre compte or il est refusé. Que se passe−t−il et comment corriger le problème ?
- Dans quels répertoires se trouvent les fichiers log d'Apache et comment se nomment ces fichiers ?

Chapitre 13. TP 1 : installation d'un serveur HTTP

13.1. Résumé

Installation d'un serveur WEB − TP(s)

La séquence est bâtie pour des travaux réalisés avec plusieurs machines. Certaines parties pourront être réalisées sur votre propre machine, celle−ci servant de client et de serveur.

Vous devez avoir un navigateur d'installé, par exemple mozilla, konqueror, galeon...

La résolution de noms doit fonctionner.

Attention : Les paramètres peuvent différer d'une version à l'autre de Linux ou d'une distribution à l'autre. J'utilise dans ce document des variables, vous devrez y substituer les valeurs réelles de votre environnement.

- $APACHE_HOME, répertoire dans lequel sont stockées les pages du serveur.
- $APACHE_CONF, répertoire dans lequel sont stockés les fichiers de configuration.
- $APACHE_USER, compte utilisateur sous lequel fonctionne Apache.
- $APACHE_GROUP, groupe auquel est rattaché le compte $APACHE_USER.

13.2. Installation d'un serveur Web

13.2.1. Introduction

Vous allez réaliser les opérations suivantes:

- configurer le serveur HTTP pour qu'il soit activé en mode standalone
- activer le serveur HTTP,
- tester le fonctionnement du serveur

A la fin vous devriez pouvoir accéder sur toutes les machines (serveurs HTTP) du réseau à partir du navigateur client.

Attention Vous utiliserez les éléments donnés dans la fiche de cours.

13.2.2. Configuration du serveur

Vous allez réaliser une configuration de base du serveur. Vous allez donc modifier le fichier httpd.conf. Avant toute modification, faites une copie de sauvegarde des fichiers.

Ouvrez le fichier à l'aide d'un éditeur, relevez et vérifiez les paramètres suivants. Pour chacun de ces paramètres vous noterez leurs rôles à partir des commentaires donnés par les fichiers httpd.conf (pensez à enregistrer vos modifications) :

- ServerName, nom pleinement qualifié de votre serveur (FQDN)
- ServerType standalone
- Port 80
- User $APACHE_USER
- Group $APACHE_GROUP
- ServerAdmin root@localhost
- ServerRoot /etc/apache
- DocumentRoot $APACHE_HOME/html
- UserDir public_html
- DirectoryIndex index.html index.shtml index.cgi
- AccessFileName .htaccess
- Alias /icons/ $APACHE_HOME/icons/
- ScriptAlias /cgi−bin/ $APACHE_HOME/cgi−bin/

13.2.3. Activation du serveur

Regardez dans la fiche de cours les commandes de lancement du service serveur et la procédure de test. Regardez dans les fichiers de log et dans la table de processus si le service est bien démarré. Vérifier avec la commande netstat que le port 80 est bien ouvert.

Notez toutes les commandes que vous utilisez.

13.2.4. Test de la configuration

A ce stade le serveur est configuré et fonctionne. Il ne reste plus qu'à réaliser les tests. Vous devez pour cela activer le navigateur.

Faites les tests à partir de la machine locale et d'une machine distante. Utilisez les adresses localhost, 127.0.0.1, les adresses IP et les noms d'hôtes.

Si tout fonctionne vous êtes en mesure de déployer votre site. Il suffira pour cela de l'installer dans l'arborescence $APACHE_HOME.

Dépannage: si cela ne fonctionne pas, procédez par élimination.

- 1 − Essayez avec les adresses IP des machines. Si ça fonctionne c'est que la résolution de noms n'est pas en place.
- 2 − Vérifiez que votre serveur est bien actif.
- 3 − Vérifiez que la configuration du serveur est correcte. Si vous apportez des modifications vous devez réinitialiser le serveur HTTP.

13.2.5. Auto−évaluation sur le premier TP

- Quels sont le/les fichiers de base pour la configuration du serveur apache et dans quels répertoires sont−ils situés ?
- Comment se nomme le compte d'utilisateur qui utilise le serveur http ?
- Quels sont les permissions d'accès par défaut sur le site principal du serveur ?
- Dans quel répertoire sont installés par défaut les pages HTML du site ?
- Quels sont les deux modes de lancement du serveur ?
- Dans quel fichier détermine−t−on ce mode de fonctionnement ?
- Dans quel répertoire par défaut sont stockés les scripts CGI et quel en est l'alias ?
- Quel est le principal rôle des alias ?
- Quelle(s) procédure(s) peut−on utiliser pour déterminer l'état du serveur et son bon fonctionnement ?
- Vous installez un serveur Apache sur une machine d'adresse 192.168.90.1 et de nom foo.foo.org. Lors des tests sur la machine locale, les commandes http://localhost, http://127.0.0.1, http://192.168.90.1 fonctionnent et http://foo.foo.org ne fonctionne pas. Lors des tests à partir d'une machine distante les commandes http://192.168.90.1 et http://foo.foo.org fonctionnent.

  Que peut−on en déduire et comment résoudre le problème ?

Chapitre 14. TP 2 : création de pages Web

14.1. Résumé

Vous allez réaliser les opérations suivantes

- Vérifier que la configuration de votre machine est correcte,
- Développer quelques pages HTML puis les déployer,
- Tester les nouvelles pages à partir d'un client Linux et Windows.

Vous utiliserez les archives qui vous sont fournies. Ces archives sont composées des quelques pages html qui vous serviront de site web et de scripts cgi.

14.2. Vérification de la configuration

Installez le service serveur et vérifiez qu'il est bien configuré et actif.

Pour tester la configuration de votre serveur, vous pouvez également utiliser la procédure suivante à partir de l'hôte local ou d'un hôte distant.

Lancez la commande suivante "$ telnet @IP du PC 80" (exemple : telnet 192.168.1.1 80 si cette adresse est celle de votre machine)

Cette commande crée une connexion au serveur httpd (port 80). Ce dernier invoque un agent.

Identifiez la connexion réseau dans une autre fenêtre xterm et avec la commande :

```
netstat −atup | grep ESTABLISHED
root@mr:/home# netstat −atup | grep ESTABLISHED
#Vous devriez obtenir quelque chose comme :
tcp        0      0 mr:33073   mr:www           ESTABLISHED 1513/telnet
tcp        0      0 mr:www     mr:33073         ESTABLISHED 1508/apache
```

Ensuite, transmettez à l'agent la ligne (commande) suivante : "GET /index.html"

Vérifiez que l'agent transmet de manière transparente le document HTML, et qu'il coupe automatiquement la connexion.

14.3. Installation d'un site Web

Vous allez utiliser les documents HTML fournis en annexe. Vous allez procéder de la façon suivante:

- Créez un répertoire $APACHE_HOME/journal pour y mettre toutes les pages html
- Copiez les images dans $APACHE_HOME/icons (normalement /usr/share/apache/icons)
- Copiez le script cgi compilé "prog" dans $APACHE_HOME/cgi−bin (normalement /usr/lib/cgi−bin)

Testez le site à partir d'un navigateur avec la commande http://@URLDuServeur/journal/

Le site est maintenant déployé, testez l'enchaînement des pages, l'affichage des images.

Le formulaire ne fonctionne pas encore. Vous avez copié le script compilé "prog" dans "/usr/lib/cgi−bin". Vérifiez quel est le nom du script que le formulaire "form.html" essaie de lancer sur le serveur.

Modifiez le nom du script ou le formulaire en conséquence.

Testez le fonctionnement du formulaire dans un navigateur.

Si vous rencontrez des difficultés sur l'exécution du script, vérifiez dans le fichier de configuration d'apache que vous avez bien :

```
    ScriptAlias /cgi−bin/ "/usr/lib/cgi−bin/"
et
    <Directory "/usr/lib/cgi−bin">
        AllowOverride None
        Options None
        Order allow,deny
        Allow from all
    </Directory>
```

14.4. Développement d'un site

Réalisez sous LINUX votre curriculum vitae en langage HTML. Celui−ci devra être composé de plusieurs documents reliés par des liens (ancres). Il sera installé dans $APACHE_HOME/cv et les images dans le répertoire référencé par l'alias /icons/.

- − 1 page d'accueil de présentation avec les liens sur les autres pages,
- − 1 page pour la formation initiale,

- − 1 page pour les expériences professionnelles,
- − 1 page pour les loisirs, passions...

Chaque page doit vous permettre de revenir à la page d'accueil.

Mettez les pages dans le répertoire qui était prévu $APACHE_HOME/cv

14.5. Test de vos pages

Vous allez vous connecter à votre site à partir d'un client distant. Utilisez de préférence les adresses URL. Corrigez les erreurs si l'accès n'est pas réalisé.

14.6. Utilisation des alias

Afin de comprendre le fonctionnement des alias vous allez maintenant réaliser quelques manipulations. Vous allez déplacer le répertoire qui contient les images de "$APACHE_HOME/icons" vers "/tmp/httpd/icons.

- Réalisez l'opération de déplacement du répertoire vers /tmp/httpd/icons,
- Apportez les modifications nécessaires aux fichiers de configuration d'Apache
- Vérifiez le résultat.

Vous constatez ainsi qu'il est possible de déplacer un répertoire sur un serveur, sans qu'il soit nécessaire pour autant de modifier toutes les pages utilisant ce répertoire.

14.7. Auto évaluation sur le deuxième TP

- Quel est le nom de la page par défaut qui est ouverte par le navigateur dans un répertoire du serveur HTTP.
- Quel intérêt procure l'utilisation des alias ?
- Dans quels répertoires sont, par défaut installés les pages HTML, scripts CGI, images et comment se nomment les alias ?
- On crée un répertoire $APACHE_HOME/html/journal pour y stocker des pages HTML. Il n'est pas possible d'y accéder alors que pour les autres sites tout fonctionne. Voici le message renvoyé par le navigateur. Aucune mesure de sécurité n'a été mise en oeuvre.

  ```
  Forbidden
  you don't have permission to access / on this server
  ```

  Quelle est la cause du problème et comment y remédier ?

Chapitre 15. TP 3 : configuration des répertoires personnels

Vous allez mettre en place un accès pour les utilisateurs du système. Ceux−ci auront la possibilité de mettre leurs pages personnelles dans leur répertoire privé.

Vous allez réaliser les opérations suivantes:

- Configurer le compte personnel,
- Développer un site personnel,
- Tester l'accès au site personnel.

Relevez dans le fichier de configuration d'Apache le nom du répertoire dans lequel doivent être stockées les pages personnelles.

15.1. Configurer le compte personnel

- Créez un compte d'utilisateur. Je vais utiliser, pour la description des opérations le compte "mlx",
- Allez dans le répertoire personnel /home/mlx,
- Créez le répertoire du site Web personnel ,
- Dans ce répertoire vous allez créer un répertoire pour les pages, un pour les images, un pour les scripts CGI avec la commande mkdir.

15.2. Développer un site personnel

Vous allez utiliser les pages HTML fournies en annexes. Utilisez les documents du TP précédent si vous en avez besoin.

Installez les fichiers fournis en annexe dans les répertoires adéquats.

Modifiez les pages HTML à l'aide d'un éditeur pour qu'elles utilisent les images de votre répertoire personnel "~/public_html/images" et le script CGI de votre répertoire personnel "~/public−html/cgi−bin" et pas ceux qui sont dans "$APACHE_HOME/cgi−bin".

Pour autoriser l'utilisation de scripts CGI dans un répertoire, vous devez le déclarer pour le serveur Apache. Voici trois exemples :

```
<Directory /home/*/public_html/cgi-bin>
    Options ExecCGI
    SetHandler cgi-script
</Directory>
<Directory /home/*/public_html>
                Options +ExecCGI
</Directory>
<Directory /var/www/journal>
                Options +ExecCGI
</Directory>
```

Recherchez également la ligne dans le fichier httpd.conf :

```
AddHandler cgi-script .cgi .sh .pl
```

Décommentez là ou ajoutez la si elle n'y est pas.

Copiez le script dans le répertoire que vous réservez pour les scripts. Vérifiez si c'est un programme "C" compilé qu'il porte bien l'extension ".cgi", au besoin renommez-le.

15.3. Tester l'accès au site personnel

Vous pouvez maintenant tester votre site personnel. A l'aide d'un navigateur utilisez l'URL "http//localhost/~mlx", (remarquez l'utilisation du "~" pour définir le répertoire personnel.)

Corrigez toutes les erreurs que vous pouvez rencontrer (problèmes d'alias, page principale, page de liens, problèmes de scripts, permissions d'accès au répertoire...)

Faites le test avec les sites personnels situés sur les autres machines.

15.4. Auto-évaluation sur le troisième TP

- Quel avantage présente l'utilisation des répertoires personnels pour le développement de sites Web ?
- Vous installez votre site personnel et vos pages. Vous tentez de réaliser un test or vous n'arrivez pas à accéder à vos pages. Quels peuvent être les problèmes et comment y remédier ?
- Vous rencontrez un problème de configuration. Vous apportez les corrections dans les fichiers de configuration, or la modification n'est toujours pas prise en compte sur le client. Que se passe-t-il et comment corriger le problème ?
- Comment avez vous fait pour que les scripts personnels soient chargés et exécutés de /home/mlx/public_html/cgi-bin
- Pour l'utilisateur mlx, sur la machine saturne et le domaine toutbet.edu, donnez:

    1. l'adresse URL de son site personnel,
    2. l'emplacement physique de son répertoire personnel sur la machine,
    3. le nom (et chemin complet) du fichier qui est activé quand on accède à son site.

Chapitre 16. TP 4 : mise en place d'un accès sécurisé

Vous allez réaliser les opérations suivantes:

- 1 - Déployer un site d'accès en ligne
- 2 - Sécuriser l'accès à ce site par un mot de passe
- 3 - Tester la configuration.

16.1. Déployer un site d'accès en ligne

Vous allez utiliser les pages fournies en annexe.

Créez un répertoire sur votre machine. "mkdir $APACHE_HOME/protege"

Copiez les pages dans ce répertoire.

16.2. Sécuriser l'accès à ce site par un mot de passe

Dans un premier temps vous allez interdire l'accès à tout le monde. Pour cela vous allez modifier le fichier de configuration d'Apache et y mettre les lignes suivantes :

```
<Directory $APACHE_HOME/protege>
order deny,allow
deny from all
</Directory>
```

Arrêtez puis relancez le serveur et faites un test à partir d'un navigateur. Notez le message qui apparaît. Plus personne n'a accès au site.

Pour mettre un accès sécurisé par mot de passe il manque 2 éléments:

- Modifiez la configuration d'accès au répertoire,
- Créez le mot de passe crypté.

Modifiez la configuration d'accès au répertoire

```
<Directory $APACHE_HOME/protege>
AuthName Protected
AuthType basic
AuthUserFile $APACHE_CONF/users # fichier de mots de passe
<Limit GET POST>
require valid-user   # ici on demande une authentification
</Limit>
</Directory>
```

Créez le mot de passe crypté.

Le mot de passe est un fichier texte à deux champs séparés par un "deux points" (:). Le premier champ contient le compte d'utilisateur, le deuxième contient le mot de passe crypté.

Pour créer ce fichier, les comptes et les mots de passe, utilisez la commande "htpasswd"

16.3. Tester la configuration.

Ouvrez une session à l'aide d'un navigateur et ouvrez l'URL "http://localhost/protege"

Une fenêtre doit s'ouvrir, entrez le nom d'utilisateur et le mot de passe.

Réalisez les opérations à partir de machines distantes.

Dépannage:

- Vérifiez le nom du répertoire que vous avez créé et la déclaration dans le fichier access.conf,
- Vérifiez le nom et la structure du fichier dans lequel vous avez mis les mots de passe.
- Si vous faites plusieurs tests, quittez puis relancez le navigateur après chaque session ouverte ou refusée,
- Vérifiez que le répertoire soit bien en mode 755 (chmod)
- Si cela ne fonctionne toujours pas reprenez le processus au début:

  − affectez toutes les permissions à tout le monde,

  − supprimez toutes les permissions à tout le monde,

  − affectez les restrictions.
- Utilisez un compte sans mot de passe. Le fichier $APACHE_CONF/users va contenir uniquement la chaîne: mlx, mais il n'y a pas le mot de passe.

  Si cela fonctionne alors le problème vient du cryptage du mot de passe.

Une fois que vous avez été authentifié, quittez et relancez le navigateur si vous voulez refaire le test d'accès à la ressource protégée.

16.4. Les fichiers .htaccess

En vous basant sur ce que vous venez de faire, sécurisez l'accès à un autre répertoire en utilisant un fichier .htaccess.

16.5. Auto−évaluation sur le quatrième TP

- Dans quel cas un accès sécurisé peut−il être utilisé ?
- Vous affectez un mot de passe à un utilisateur distant. Vous faites un test sur votre machine tout semble fonctionner. Il vous appelle pour vous dire que ses requêtes sur le site protégé sont toujours rejetées. Que se passe−t−il ?
- Si on utilise les possibilités du système, quelle autre solution peut−on utiliser pour interdire l'accès à un répertoire.

Chapitre 17. TP 5 : Utilisation de scripts CGI

Il existe deux méthodes qui permettent de faire communiquer un formulaire html avec un script CGI. La méthode POST et la méthode GET. Nous utiliserons ici la méthode POST.

Ce TP doit vous permettre de développer un formulaire et un script CGI en C. Vous devez savoir compiler un programme.

Vous allez réaliser les opérations suivantes:

- Étudier les sources fournies en annexe.
- Développer un formulaire et adaptez les scripts,
- Tester le fonctionnement des scripts.

17.1. Étudier les sources fournies en annexe

Transférez les programmes C, les .h et le makefile dans votre répertoire personnel. Étudiez attentivement les sources. Testez le fonctionnement du script.

17.2. Développer un formulaire et adapter les scripts

A partir de l'exemple fourni, développez les pages HTML d'un site commercial qui doit permettre la prise de commande à distance de pizzas. Il doit y avoir au moins 3 pages:

- une page principale,
- une page de description des produits,
- une page (formulaire) pour passer commande contenant tous les types de champs qui existent dans le formulaire form.html. (liste déroulante, bouton radio, zone de texte)

Exemple :

- Zone de texte (Nom du client)
- Liste déroulante (Calzone, Margarita, Quatre−saisons)
- Bouton radio (Grande, Moyenne, Petite)
- Case à cocher (Avec des anchois)
- Vous afficherez au client le résultat de sa commande.

Adaptez le script CGI qui vous est fourni, à votre formulaire.

17.3. Tester le fonctionnement de votre script.

Pour tester le script:

- ouvrez la page principale de votre site à l'aide d'un navigateur,
- passez des commandes,
- vérifiez les résultats retournés par le script. (réponse).

17.4. Auto−évaluation sur le cinquième TP

- Que signifie CGI ?
- Quel intérêt présente l'utilisation de scripts CGI ?
- Quelle est la différence entre la méthode GET et POST ?
- Comment se nomment les variables d'environnement qui contiennent la chaîne (paramètres) du formulaire ?
- Comment est structurée cette chaîne ?
- Quel est le caractère de concaténation des chaînes ?
- Pourquoi la réponse contient dans l'entête la chaîne Content−type: text/html ?
- A quoi correspondent ces 2 paramètres text et html ?

Chapitre 18. TP 6 : Serveurs webs virtuels et redirection

Il est préférable d'avoir réalisé les TP sur les serveurs de noms avant de réaliser celui−ci.

Pour réaliser ce TP, vous devrez avoir un serveur de noms qui fonctionne.

La mise en place de serveurs webs virtuels, permet de faire cohabiter plusieurs serveurs sur un même hôte. Nous verrons qu'il y a plusieurs techniques pour faire cela :

- Les serveurs virtuels basés sur le nom, dans ce cas, vous devrez désigner pour une adresse IP sur la machine (et si possible le port), quel est le nom utilisé (directive ServerName), et quelle est la racine du site (directive DocumentRoot).

```
# Ici toutes les adresses ip sont utilisées, on peut mettre *
    NameVirtualHost *

    <VirtualHost *>
    ServerName www.domain.tld
    DocumentRoot /www/domain
    </VirtualHost>

    <VirtualHost *>
```

```
    ServerName www.otherdomain.tld
    DocumentRoot /www/otherdomain
    </VirtualHost>

# Ici on utilise une adresse en particulier
# Toutes les requêtes sur http://www.domain.tld/domain
# pointeront sur /web/domain
    NameVirtualHost 111.22.33.44

    <VirtualHost 111.22.33.44>
    ServerName www.domain.tld
    ServerPath /domain
    DocumentRoot /web/domain
    </VirtualHost>
```

- Les serveurs virtuels basés sur des adresses IP. Dans ce cas, chaque serveur aura sa propre adresse IP. Vous devrez également avoir un serveur de noms sur lequel toutes les zones et serveurs sont déclarés, car c'est ce dernier qui assurera la correspondance "adresse ip <−> nom du serveur" :

```
# Chaque serveur peut avoir son propre administrateur, ses
# propres fichiers de logs.
# Tous fonctionnent avec la même instance d'Apache.
# Il est possible de lancer plusieurs instances d'apache
# mais sur des ports différents

    <VirtualHost www.smallco.com>
    ServerAdmin webmaster@mail.smallco.com
    DocumentRoot /groups/smallco/www
    ServerName www.smallco.com
    ErrorLog /groups/smallco/logs/error_log
    TransferLog /groups/smallco/logs/access_log
    </VirtualHost>

    <VirtualHost www.baygroup.org>
    ServerAdmin webmaster@mail.baygroup.org
    DocumentRoot /groups/baygroup/www
    ServerName www.baygroup.org
    ErrorLog /groups/baygroup/logs/error_log
    TransferLog /groups/baygroup/logs/access_log
    </VirtualHost>
```

La redirection est un service un peu particulier, qui diffère de celui des services web virtuels. Il permet de rediriger une partie du site web à un autre endroit du disque physique. La encore on utilisera une des fonctions d'Apache qui permet de définir des alias.

Prenons par exemple un site installé physiquement sur "/var/www" et accessible par l'url "http://FQDN". L'url "http://FQDN/unREP" devrait correspondre normalement à l'emplacement physique "/var/www/unREP". La redirection va permettre de rediriger physiquement vers un autre répertoire.

Cette technique est largement utilisée par des applications ou pour la création d'espaces documentaires.

L'URL "http://FQDN/compta", pourra pointer physiquement dans "/usr/lib/compta" au lieu de "/var/www/compta".

Dans cet atelier, vous allez réaliser les opérations suivantes:

1. Installer un service de serveurs web virtuels par adresse et par nom,
2. Mettre en place un service de redirection par alias.

18.1. Avant de commencer sur les serveurs web virtuels

Configurez votre serveur de noms si ce n'est pas fait. Vous pouvez vous aider de l'annexe sur le "web−hosting" ci−dessous.

Pour créer des alias dynamiquement à votre interface réseau, utilisez la commande suivante:

```
ifconfig eth0:0 192.168.0.1
ifconfig eth0:1 192.168.1.1
ifconfig
eth0      Lien encap:Ethernet  HWaddr 00:D0:59:82:2B:86
          inet adr:192.168.90.2  Bcast:192.168.90.255  Masque:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:95041 errors:0 dropped:0 overruns:0 frame:0
          TX packets:106227 errors:0 dropped:0 overruns:26 carrier:0
          collisions:0 lg file transmission:100
          RX bytes:57490902 (54.8 MiB)  TX bytes:11496187 (10.9 MiB)
          Interruption:11 Adresse de base:0x3000

eth0:0    Lien encap:Ethernet  HWaddr 00:D0:59:82:2B:86
          inet adr:192.168.0.1  Bcast:192.168.0.255  Masque:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          Interruption:11 Adresse de base:0x3000
```

```
eth0:1    Lien encap:Ethernet  HWaddr 00:D0:59:82:2B:86
          inet adr:192.168.1.1  Bcast:192.168.1.255  Masque:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          Interruption:11 Adresse de base:0x3000

lo        Lien encap:Boucle locale
          inet adr:127.0.0.1  Masque:255.0.0.0
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:70 errors:0 dropped:0 overruns:0 frame:0
          TX packets:70 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 lg file transmission:0
          RX bytes:4564 (4.4 KiB)  TX bytes:4564 (4.4 KiB)
```

Créez votre serveur de noms pour deux domaines par exemple (domain1.org et domain2.org), adaptez la configuration du fichier "/etc/resolv.conf", vérifiez le bon fonctionnement de la résolution de noms sur les CNAME (www.domain1.org et www.domain2.org).

18.2. Serveur web virtuel basé sur les adresses ip

D'abord les déclarations dans le fichier /etc/apache/Vhosts:

```
# Web hosting ip based
NameVirtualHost 192.168.0.1
<VirtualHost 192.168.0.1>
DocumentRoot /home/web/domain1
ServerName www.domain1.org
</VirtualHost>

NameVirtualHost 192.168.1.1
<VirtualHost 192.168.1.1>
DocumentRoot /home/web/domain2
ServerName www.domain2.org
</VirtualHost>
```

Inclusion à la fin du fichier de configuration d'apache :

```
Include Vhosts
```

Préparation des sites webs (sans trop se casser la tête) :

```
# mkdir −p /home/web/domain1 /home/web/domain2
# echo "<H1>Salut domain1 </H1>" > /home/web/domain1/index.html
# echo "<H1>Salut domain2 </H1>" > /home/web/domain2/index.html
```

On relance Apache :

```
/etc/init.d/apache restart
```

C'est terminé, les requêtes sur "http://ww.domain1.org", "http://www.domain2.org", doivent fonctionner et vous renvoyer la bonne page.

18.3. Serveur Web virtuel basé sur le nom

On va utiliser l'adresse ip de ns1.domain1.org pour les url w3.domain1.org et wiki.domain1.org. w3 et wiki sont deux zones de déploiement de site web différentes, sur un même serveur HTTP.

Modification du fichier Vhosts :

```
# Web hosting name based (basé sur le nom)
#Site secondaire de domain1
 <VirtualHost 192.168.0.1>
ServerName w3.domain1.org
DocumentRoot /home/web/w3
</VirtualHost>

 <VirtualHost 192.168.0.1>
ServerName wiki.domain1.org
DocumentRoot /home/web/wiki
</VirtualHost>
```

Relancer Apache.

Création des sites web :

```
# mkdir −p /home/web/wiki /home/web/w3
# echo "<H1>Salut w3 </H1>" > /home/web/w3/index.html
# echo "<H1>Salut wiki </H1>" > /home/web/wiki/index.html
```

C'est terminé, les requêtes sur :

```
http://w3.domain1.org
http://wiki.domain1.org
```

doivent retourner les bonnes pages.

18.4. Application sur la redirection

Nous allons créer un espace documentaire pour le domain "domain1.org". Il sera accessible par l'url "http://www.domain1.org/mesdocs", mais il sera localisé dans "/etc/domain1doc".

Créez un répertoire "domain1doc" dans /etc pour ce nouveau projet et mettez−y le fichier "apache.conf" ci−dessous:

```
# Création du répertoire
mkdir /etc/domain1doc

# Création du fichier /etc/domain1doc/apache.conf
Alias /mesdocs /etc/domain1doc/
<Directory /etc/domain1doc>
Options FollowSymLinks
AllowOverride None
order allow,deny
allow from all
</Directory>
```

Faites une inclusion de ce fichier dans le fichier de configuration d'Apache et relancez le service Apache.

C'est terminé, toutes les requêtes sur "http://www.domain1.org/mesdocs", ouvriront les pages qui sont dans "/etc/domain1doc".

18.5. Annexe pour le "web−hosting"

```
=== A rajouter dans /etc/bind/named.conf

zone "domain1.org" {
        type master;
        file "/etc/bind/domain1.org.hosts";
        };

zone "domain2.org" {
        type master;
        file "/etc/bind/domain2.org.hosts";
        };



=== fichiers de zone
root@freeduc−sup:/etc/bind# more domain1.org.hosts
$ttl 38400
@                       IN      SOA     domain1.org. hostmaster.domain1.org. (
                        2004052604
                        10800
                        3600
                        604800
                        38400 )
@                         IN      NS      ns1.domain1.org.
ns1                       IN      A       192.168.0.1
www                       IN CNAME ns1
wiki                      IN CNAME ns1
w3                        IN CNAME ns1

root@freeduc−sup:/etc/bind# more domain2.org.hosts
$ttl 38400
@                       IN      SOA     domain2.org. hostmaster.domain2.org. (
                        2004052604
                        10800
                        3600
                        604800
                        38400 )
@                         IN      NS      ns1.domain2.org.
ns1                       IN      A       192.168.1.1
www                       IN CNAME ns1
```

Chapitre 19. Éléments de cours sur le chiffrement

Ce cours se propose de définir les concepts de base du chiffrement et surtout ses objectifs :

- Définition du chiffrement ;

- Les deux méthodes de chiffrement (chiffrement symétrique et chiffrement asymétrique) ;
- Les objectifs du chiffrement (confidentialité, authentification, intégrité, signature électronique) ;
- La notion de certificats ;
- La mise en oeuvre avec le protocole SSL.

19.1. Qu'est−ce−que le chiffrement ?

Le chiffrement consiste à rendre illisible un message en brouillant ses éléments de telle sorte qu'il soit très difficile de reconstituer l'original si l'on ne connaît pas la transformation appliquée.

Le chiffrement est basé sur deux éléments : **une clé et un algorithme**.

- La clé est une chaîne de nombres binaires (0 et 1).
- L´algorithme est une fonction mathématique souvent complexe qui va combiner la clé et le texte à crypter pour rendre ce texte illisible.

Pour "casser" un chiffrement, il n'existe que deux solutions :

- Il faut essayer toutes les clés, ce qui devient impossible actuellement compte tenu de la longueur de celles−ci ;
- "Casser" l'algorithme (qui est, en général, connu par tous), c'est−à−dire rechercher une faille dans la fonction mathématique ou dans son implémentation permettant de trouver la clé plus rapidement.

Le **choix d'un algorithme** dépend donc de :

- La fiabilité de ce dernier (savoir s'il existe des failles importantes) ;
- La longueur de clé qu'il gère ;
- De ce que l'on va en faire : on sera plus vigilant pour un serveur RADIUS permettant l'accès au réseau WIFI de la NASA que pour notre messagerie personnelle...

19.2. Les mécanismes de chiffrement

19.2.1. Le chiffrement symétrique

La même clé est utilisée pour coder et décoder et doit bien évidemment rester secrète.

Un algorithme répandu consiste à effectuer des substitutions et des transpositions en cascade sur les lettres du message. Par exemple : MICROAPPLICATION devient IRMCAPOPIALCINTO si l'on fixe comme **clé** le principe suivant : "la première lettre sera la troisième, la seconde sera la première, la troisième sera la quatrième et la quatrième sera la seconde".

La figure suivante illustre le principe du chiffrement symétrique.

Figure 19−1. Chiffrement symétrique

Quelques algorithmes couramment utilisés (liste non exhaustive) :

- Data Encryption Standard (DES) est une invention du National Bureau of Standards (NSB) américain, qui date de 1977 : c'est le premier algorithme de chiffrement publique gratuit. Les données sont découpées en blocs de 64 bits et codées grâce à la clé secrète de 56 bits propre à un couple d´utilisateurs
- RC2, RC4 et RC5 sont des algorithmes créés à partir de 1989 par Ronald Rivest pour la RSA Security. L'acronyme "RC" signifie "Ron's Code" ou "Rivest's Cipher". Ce procédé permet aux utilisateurs la possibilité de choisir la grandeur de la clé (jusqu'à 1024 bits).
- Advanced Encryption Standard (AES) est un standard approuvé par le ministère américain du commerce en 2001 et vise à remplacer le DES ; il est placé dans le domaine public et accepte des clés d'une taille de 128, 192 ou 266 bits.

Les inconvénients de ce système de chiffrement sont les suivants :

- il faut autant de paires de clés que de couples de correspondants ;
- la non−répudiation n´est pas assurée. Mon correspondant possédant la même clé que moi, il peut fabriquer un message en usurpant mon identité ;
- il faut pouvoir se transmettre la clé au départ sans avoir à utiliser le média à sécuriser !

19.2.2. Le chiffrement asymétrique

C'est une approche radicalement différente apparue en 1976 : **la clé qui sert à coder est différente de celle qui peut déchiffrer**.

La grande nouveauté introduite par cette famille de méthodes, c'est que **la clé chiffrante est publique**. Cette notion de cryptographie à clé publique résulte d'un défi mathématique lancé par Witfield Diffie et Martin Hellman. Trois mathématiciens américains (Ronald Rivest, Adi Shamir et Leonard Adleman) ont réussi à l'époque à trouver une solution, aujourd'hui employée sous le nom de RSA.

La méthode repose sur un constat très simple : il est facile d'effectuer la multiplication de deux nombres premiers, mais il est très difficile de retrouver les facteurs quand le résultat est grand.

La clé publique est donc constituée du produit n de deux nombres premiers p et q, choisis très grands. La clé secrète dépend directement de p et q et ne peut donc etre déduite de la clé publique.

En résumé, chacun dispose d´une clé privée qui est gardée secrète et d´une clé publique qui est destinée à être divulguée. Ces clés sont liées entre elles. Un document encrypté avec l´une ne peut être décodée qu´avec l´autre. Toutefois, la possession de l´une des clés ne permet pas d´en déduire l´autre.

Un autre algorithme utilisant le système de chiffrage asymétrique largement utilisé est le DSA (Digital Signature Algorithm).

Ce système a réellement permit d'assurer des fonctions essentielles telles que la **confidentialité** et l'**authentification**.

19.3. Que permet de faire le chiffrement ?

19.3.1. Garantir la confidentialité d'un message

Le message ne doit être pouvoir lu que par le destinataire.

Pour ce faire, il suffit de récupérer la clé publique P du destinataire (sur le serveur de clé http://www.keyserver.net/ par exemple) puis de crypter le message avec cette clé publique et d´envoyer le résultat au destinataire.

Le destinataire n´a lui, qu´à décoder le message à l´aide de sa clé privée S. Seule la clé privée S correspondant à la clé publique P peut décoder un message encrypté avec cette clé publique P. Comme le destinataire est seul à posséder cette clé privée (secrète), on est sûr de la confidentialité du message.

La figure suivante illustre le principe de confidentialité :

Figure 19−2. Confidentialité

19.3.2. Authentifier l'émetteur d'un message

le chiffrement asymétrique permet de garantir que l´émetteur d´un message est bien celui que l´on croit.

L´émetteur va encrypter le message avec sa clé privée. Le destinataire pourra alors vérifier l´identité de l´émetteur en décryptant le message avec la clé publique de l´émetteur. Comme seul le détenteur de la clé privée est capable de crypter un message déchiffrable par la clé publique, on est sûr de l´identité de l´émetteur.

La figure suivante illustre le principe d'authentification :

Figure 19−3. Authentification

En pratique, on évite de crypter tout le message. On crypte une forme abrégée du message. Cette forme est obtenue à l´aide d´une **fonction mathématique complexe appelée fonction de hachage à sens unique.**

C'est ce que l'on apelle **la signature électronique.**

19.3.3. La signature électronique

La signature électronique permet non seulement **d'authentifier l'emetteur d'un message** mais aussi **de vérifier l'intégrité** de ce dernier (c'est à dire qu'il n'a pas été modifié).

Techniquement, la signature électronique correspond au chiffrement de l'empreinte d'un document via la clé privée du signataire.

Selon le glossaire du CNRS (Centre National de la Recherche Scientifique) http://www.dr15.cnrs.fr/Delegation/STI/Signature/glo−empreinte.html : "L'empreinte électronique est à un document ce que l'empreinte digitale ou génétique est à un individu. L'empreinte d'un document a des propriétés très intéressantes :

- si on change, ne serait−ce qu'une virgule, dans un document, son empreinte change ;
- la probabilité que deux documents aient la même empreinte est à peu de choses près nulle ;
- il n'est pas possible de refabriquer le document à partir de son empreinte.

Techniquement, l'empreinte d'un document est le résultat d'un calcul effectué à l'aide d'un algorithme de hachage approprié (SHA (Secure Hash Algorithm) ou MD5 (Message Digest) sont les plus courants). Il génère pour chaque document un nombre de 128 ou 168 bits, que les anglo−saxons appellent "digest" et qu'en français, on appelle parfois résumé..."

Le schéma suivant illustre le mécanisme de la signature électronique.

Figure 19−4. Sinature électronique

Si les deux opérations donnent le même résultat, alors on est sûr de l'identité du signataire et on est sûr que le document n'a pas été modifié depuis que l'émetteur l'a signé.

Bien entendu, l'ensemble de ces vérifications se font à l'aide de logiciels dédiés qui effectuent ces opérations très rapidement et de façon très assistée.

19.3.4. Mise en oeuvre

La mise en oeuvre de ces principes dans le cadre d'une messagerie électronique.

Voici des adresses de site sur lequel vous trouverez des documentations complètes très pédagogiques sur l'utilisation du célèbre logiciel PGP et de son pendant libre openpgp (ou GnuGP) ; logiciel très utilisé pour le chiffrement et l'établissement de signature numérique lors d'envoi de mèl. :

- http://www.securiteinfo.com/crypto/pgp.shtml
- http://www.openpgp.fr.st/
- http://www.gnupg.org/gph/fr/manual.html
- Et une documentation complémentaire pour gérer pgp ou gpg à partir de Kmail sous Linux : http://docs.kde.org/fr/HEAD/kdenetwork/kmail/pgp.html

Vous avez tous l'adresse électronique de votre professeur qui a déposé sa clé publique relative à cette adresse sur le serveur de clés (en anglais) http://www.keyserver.net/.

Vous devez la récupérer et envoyer à votre professeur un... gentil... message chiffré et signé par votre propre clé privé ; signature que votre professeur doit pouvoir vérifier...

Votre professeur vous répondra à tous un message chiffré et signé...

Mais un autre problème peut se poser : tout est basé sur la confiance que l'on accorde à la clé publique or **comment contrôler que la clé publique appartient bien à celui qui le prétend ?**

C'est le role des **certificats**.

19.4. Les certificats

19.4.1. L'utilité d'un certificat

Pour être sûr que la clé publique provient bien de celui que l'on croit, on utilise une **autorité tierce** (appelé le **tiers de confiance**). Cette autorité est celle qui va **générer une clé publique certifiée** par exemple pour un serveur Web, puis c'est ensuite elle qui **garantira à tout demandeur** (par exemple le client web) que la clé publique envoyée appartient bien à celui qui le prétend (au serveur Web).

La garantie qu'une clé publique provient bien de l'émetteur qu'il prétend être, s'effectue donc via un certificat d'authenticité émanant d'une autorité de certification (AC), le tiers de confiance.

19.4.2. Qu'est−ce qu'un certificat x509 ?

x509 définit la forme d'un certificat (simple fichier informatique) délivré par une **autorité de certification** qui contient :

- la clé publique de son détenteur et des informations sur son identité ;
- le nom distinctif de l'autorité de certification ;
- la signature électronique (chiffrement de l'empreinte par clé privé) de l'autorité de certification.

Pour obtenir plus de précisions sur le certificat x509, vous pouvez consulter la page suivante http://interpki.sourceforge.net/index.php?/x509.html

Comment obtenir un certificat ?

On peut acheter un certificat SSL 128 bits à un prix raisonnable aux alentours de 120 EUR HT /an. (Voir par exemple à cette adresse http://www.netenligne.com/ecommerce/ssl1.asp et http://www.netenligne.com/ecommerce/ssl2.asp pour le tableau comparatif des prix).

Voilà ce que l'on peut lire en ligne : "Pour obtenir un certificat, il faut fournir des données d'identification. Tout propriétaire d'un nom de domaine et du site correspondant peut obtenir un certificat SSL, qu'il s'agisse d'une personne physique, d'une entreprise, d'une association ou d'une administration.

S'il n'y a pas de problème concernant l'identification du demandeur et son droit à obtenir un certificat SSL pour son site, le délai d'obtention est de 1 à 2 jours ouvrés"

Pour un usage interne, on peut se "fabriquer", avec quelques outils logiciels nos propres certificats. C'est d'ailleurs ce que l'on va faire dans le prochain TP.

Comment vérifier un certificat ?

C'est exactement la même méthode que celle utilisée lors de la vérification de la signature électronique d'un document.

La figure suivante illustre un exemple de certificat émis pour un serveur WEB.

Figure 19−5. Certificat

Bien sûr, vous devez "détenir" le certificat contenant la clé publique de l'autorité de certification.

Lorsque vous devez vérifier le certificat d'un serveur WEB, c'est votre navigateur qui s'en charge ; Lors de l'installation d'un navigateur récent, un certain nombre de certificats d'AC sont chargés automatiquement. Si le certificat du site que vous consultez a été validé par une de ces AC, vous n'aurez pas de fenêtres d'alertes... (Voir le TP pour plus de précisions).

Des protocoles, utilisant ces techniques, ont été développés pour sécuriser des services (WEB, telnet, POP, SMTP, etc.).

En particulier, **le protocole SSL (Secure Socket Layer)**, initialement proposé par la société Netscape est aujourd'hui adopté par l'ensemble de la communauté informatique pour l'authentification et le chiffrement des données entre clients et serveurs.

Il existe actuellement un **nouveau standard basé sur SSL, TLS (Transaction Layer Security)** normalisé par l´IETF (Internet Engineering Task Force).

19.5. Le protocole SSL

19.5.1. Principes du protocole SSL

Conçu par Netscape, **SSL est un protocole se plaçant entre la couche application et la couche transport** permettant d´assurer la confidentialité, l´authentification et l´intégrité des données lors des communications.

Il peut être appliqué à des applications comme HTTP, POP, FTP,...

Sous Linux, vous pouvez aller consulter le fichier /etc/services pour connaître les numéros de port correspondants aux nouvelles implémentation de ces services au dessus de SSL.

Figure 19−6. Le protocole SSL

19.5.2. Exemple de fonctionnement du protocole SSL avec un serveur WEB

En ce qui concerne le HTTP, il a été nécessaire de définir une nouvelle méthode d´accès dans les URL baptisée HTTPS pour se connecter au port d´un serveur utilisant le SSL qui porte par défaut le numéro 443.

Le mécanisme est illustré par la figure suivante :

Figure 19−7. HTTP over SSL

1 − le navigateur fait une demande de transaction sécurisée au serveur en envoyant sa requête HTTPS://, il demande donc le certificat garantissant la clé publique du serveur.

2 − le serveur lui envoie son certificat d'authentification délivré par une autorité de certification (normalement organisme officiel). Ce certificat comporte une clé publique.

3 − Le navigateur s'assure tout d'abord que le certificat délivré est valide puis il envoie au serveur une clé secrète codée issue de la clé publique (de 56 ou 128 bits). Seul le serveur sera donc capable de décoder cette clé secrète car il détient la clé privée. Cette clé secrète ainsi créée sera utilisée pour encoder les messages (cryptographie symétrique). L'algorithme à clé secrète utilisé (ex : DES, RC4) est négocié entre le serveur et le client.

4 − Le serveur et le client possède maintenant une clé secrète partagée (la clé de session) et les échanges sont faits par l'intermédiaire de cette clé. Pour assurer l'intégrité des données, on utilise un algorithme de hash (ex : MD5, SHA). S'il y a déconnexion, une nouvelle clé de session sera négociée.

3 remarques et le supplice est terminé puisque vous allez passer à la mise en oeuvre à travers le TP sur HTTPS :

- On voit bien que ce mécanisme permet d'assurer la **confidentialité** (mécanisme de chiffrement), **l'intégrité** (algorithme de hash) et **l'authentification** (certificats).
- Dans le schéma présenté, et à des fins de simplification, il n'a pas été pris en compte l'authentification du client. Cela est possible avec le protocole SSL même si seule l'authentification du serveur est implémentée *par défaut*.
- Le chiffrement asymétrique ne sert finalement qu'à l'authentification et à transmettre la clé de session de manière sécurisée ; c'est ensuite, cette clé de session (symétrique) qui sert à chiffrer les échanges. En effet, le chiffrement symétrique est beaucoup plus rapide que le chiffrement asymétrique.

Chapitre 20. TP sur le serveur WEB sécurisé

L'objectif de ce TP est de développer une sécurité concernant l'authentification d'un serveur HTTP (le navigateur doit être certain de se connecter au "bon" serveur et les données qui transitent doivent être cryptées).

Préalable :

- apache est installé et le service démarre sans problème ;
- vous avez lu le cours sur les notions de base du chiffrement ;
- vous avez fait les 6 TP sur le serveur HTTP et notamment le dernier concernant les serveurs virtuels.

20.1. Présentation du TP

Les accès à des pages web se font à l'aide du protocole http, en empruntant le réseau Internet. Aucune garantie de confidentialité n'est assurée lors de ces accès ; il est relativement simple à un pirate d'intercepter vos requêtes (votre code de carte bleue) et les réponses faites par le serveur.

En outre, vous n'avez pas une certitude absolue d'être en cours de consultation du site que vous croyez.

Afin de palier à ces inconvénients, le protocole https avec l'authentification SSL peut être mis en oeuvre. D'une manière très schématique, il permet d'encapsuler et de crypter le trafic http. Ainsi, il sera quasiment impossible à un pirate qui intercepterait des accès à des pages chargées via le protocole https de décrypter cet échange, et donc de récupérer des informations confidentielles. En outre, https permet de s'assurer que le serveur auquel on accède est bien celui que l'on croit.

HTTPS offre d'autres possibilités qui ne sont pas abordées ici (par exemple, authentifier la personne qui accède au serveur).

Au cours de ce TP, vous allez mettre en place le protocole SSL sur votre serveur WEB

Dans le TP6, vous avez configuré un serveur virtuel auquel l'on accède avec l'url "http://wiki.domain1.org" ; c'est ce serveur que l'on va sécuriser.

20.2. Les paquets à installer

Nous supposons que vous êtes sur une distribution debian. Sinon, il faudra légèrement adapter ce qui suit notamment en ce qui concerne les noms de paquets à installer et l'emplacement des fichiers de configuration.

- libapache−mod_ssl ;
- openssl ;
- libssl ;

Mod_ssl est le module apache qui implémente https (http over ssl). Le serveur https écoute par défaut sur le port 443 au lieu de 80 et correspond à un virtual host configuré par des directives spécifiques à mod_ssl.

OpenSSL fournit notamment une application pour créer des certificats.

20.3. Étape 1 : La création des certificats

Connectez vous sous root et allez dans le répertoire de configuration de votre serveur Apache /etc/apache (on peut évidemment choisir un autre répertoire) et créez un répertoire appelé certs. Vous vous placerez dans ce répertoire avant d'effectuer les manipulations.

20.3.1. Création du certificat serveur

Génération de la clé privé

On génère la clef privé avec la commande suivante en définissant un nom de fichier :

```
openssl genrsa 1024 > servwiki.key
```

La sortie attendue est la suivante :

```
Generating RSA private key, 1024 bit long modulus
        ..................++++++
        ..........................................................++++++
        e is 65537 (0x10001)
```

Si vous souhaitez que cette clé ait un mot de passe (qui vous sera demandé à chaque démarrage d'apache, donc à éviter !), ajoutez "−des3" après "genrsa".

Ceci a pour effet de créer une clé SSL (fichier servwiki.key), ne la perdez pas... **c'est votre clé privé**... (ni éventuellement le mot de passe) !

Vous pouvez observer son contenu : less servwiki.key

```
−−−−−BEGIN RSA PRIVATE KEY−−−−−
MIICXgIBAAKBgQDQG9wvnuLC4aqzaJCAWGA1AxFzg00hjPObhq1mukzsGyuuWBFG
vj/k9vFNYX55DHctb/4cXtsZRWWvgcjtYnCVwRu+DAjFsk//kOMfhplmiv9xQ+ZL
8w/Xrnm8JWdSS+S4LCMnsuIiQtLbhMrQnUV02hAtbIZiSM3k6OjShEZhDQIDAQAB
AoGAHi0cBW+1k+qjFPbBlUq7UJSMUEKmyYmlvVSPCklTZB0gfVxZzPdDTpEcNks/
yo+rLFSD9Vsvy/9LGmLoXruadWlK67PCUnpM5/oRRGgy8t73YKrxflAU5Gtymjvc
ZCf0CAs6wBft3yLU31Qc4WqVM2vTyUH76jebVhxEw8k63OUCQQD/1OmAXV+TxBPG
ZTPFbzUeAE5rQqqOW4aoMNvM61Yn/19h6SzY2MfSQvF1BNns/efCRrqOMeyvPWUG
g1okfogTAkEA0D7pDf/D2Yu5msbOAGF4QBU1erLzpi/s6Rv6VEPYCGnHQlo3jbg9
FZbjHJ4UcYyYaA8jIrkY+FIJM88YlGbWXwJBAILEdvJ5R/CFCkKf2j2yIWmLaIol
En8fw43XI5L0PB7Hxx6KDLVu4XzVYQyahTZBdqR0eMlUNZJBhJE2tO3wi2cCQQCp
JkCFd3es0BrNxqfzlThozRFofcz88za7TldydL0YcFtC4Sb4vWsYizwktZ6jcPEm
rQz8Gl9W7MO+ynwLptB/AkEA1tsnFXoYzI71enmTdugGxbv0RqAd5iQpDYQkDSdn
2LImp/3YnXNJ9qpY91j87tKthh/Oetu6SHlmLg1LOYNIdw==
−−−−−END RSA PRIVATE KEY−−−−−
```

Protégez votre fichier en faisant : chmod 400 servwiki.key

De multiples autres options sont possibles mais il est inutile d'alourdir le sujet (à creuser pour une éventuelle PTI...)

A partir de votre clé, vous allez maintenant créer un fichier de demande de signature de certificat (CSR Certificate Signing Request).

On génère la demande de certificat avec la commande suivante :

```
openssl req −new −key servwiki.key > servwiki.csr
```

Le système va vous demander de saisir des champs ; remplissez les en adaptant sauf le champ "Common Name" qui doit être identique au nom d'hôte de votre serveur virtuel :

```
Country Name (2 letter code) [AU]:FR
State or Province Name (full name) [Some−State]:CORSE
Locality Name (eg, city) []:Ajaccio
Organization Name (eg, company) [Internet Widgits Pty Ltd]:LLB
Organizational Unit Name (eg, section) []:BTSINFO
Common Name (eg, YOUR name) []:wiki.domain1.org
Email Address []:
```

Ce n'est pas la peine de saisir d'autres "extra attributes"...

Ceci a pour effet de créer le formulaire de demande de certificat (fichier servwiki.csr) à partir de notre clé privé préalablement créée.

Ce fichier contient la clé publique à certifier. Un less servwiki.csr nous donne :

```
−−−−−BEGIN CERTIFICATE REQUEST−−−−−
MIIBsTCCARoCAQAwcTELMAkGA1UEBhMCRlIxFTATBgNVBAgTDENvcnNlIGR1IFN1
ZDEQMA4GA1UEBxMHQWphY2NpbzEMMAoGA1UEChMDTExCMREwDwYDVQQLEwhCVFMg
SU5GTzEYMBYGA1UEAxMPcHJvZi5idHNpbmZvLmZyMIGfMA0GCSqGSIb3DQEBAQUA
A4GNADCBiQKBgQDSUagxPSv3LtgDV5sygt12kSbN/NWP0QUiPlksOkF2NkPfwW/m
f55dD1hSndlOM/5kLbSBo5ieE3TgikF0IktjBWm5xSqewM5QDYzXFt031DrPX63F
vo+tCKTQoVItdEuJPMahVsXnDyYHeUURRWLWwc0BzEgFZGGw7wiMF6wt5QIDAQAB
oAAwDQYJKoZIhvcNAQEEBQADgYEAwwI4UvkfhBvyRrUXtjrLfZXLxZlF9o+URmHJ
ysvrLKfesVBEzdA9mqk1OwIwLfe8Fw2fhip2LGqvcCPxDMoIE/0cDkqGRg9iKp7D
DMuy69lPTEB6UtpVKO/1eage0oug6VqdfGAYMMSGyWFuO9FE4UE6HspVodb20wGV
4H8qZuk=
−−−−−END CERTIFICATE REQUEST−−−−−
```

Maintenant, deux choix s'offrent à vous :

- envoyer le fichier servwiki.csr à un organisme (le tiers de confiance ou **l'autorité de certification (CA)**) et ainsi obtenir le certificat dûment signé par la clé privé de l'organisme (après avoir payé),
- **ou bien signer vous même notre certificat.**

C'est ce dernier choix que nous allons voir.

20.3.2. Création du certificat de l'autorité de certification

Pour signer un certificat, vous devez devenir votre propre autorité de certification, cela implique donc de réaliser une clé et un certificat signé.

La création de la clé privé de l'autorité de certification se fait comme précédemment :

```
openssl genrsa -des3 1024 > ca.key
```

Ce qui a pour effet de créer la clé privé de l'autorité de certification. Dans ce cas, il vaut mieux rajouter l'option -des3 qui introduit l'usage d'une "passphrase" (mot de passe long qui peut même contenir du blanc) car c'est cette clé privé qui signera tous les certificats que l'on emmettra ; cette "passphrase" sera donc demandée à chaque utilisation de la clé.

Ensuite, à partir de la clé privé, on crée un certificat x509 pour une durée de validité d'un an auto-signé :

```
openssl req -new -x509 -days 365 -key ca.key > ca.crt
```

Il faut saisir la passphrase... puisqu'on utilise "ca.key" que l'on a protégée. Et on donne les renseignements concernant cette fois-ci l'autorité de certification (c'est à dire nous-même).

Attention : le nom de "Common Name" doit être différent de celui qui a été donné pour la clé.

```
Country Name (2 letter code) [AU]:FR
State or Province Name (full name) [Some-State]:CORSE
Locality Name (eg, city) []:Ajaccio
Organization Name (eg, company) [Internet Widgits Pty Ltd]:LLB
Organizational Unit Name (eg, section) []:BTSINFO
Common Name (eg, YOUR name) []:cert_CA
Email Address []:
```

C'est notre certificat d'autorité de certification qui va permettre de signer les certificats créés.

20.3.3. La signature du certificat serveur par le CA (Certificate Autority)

La commande qui signe la demande de certificat est la suivante :

```
openssl x509 -req -in servwiki.csr -out servwiki.crt -CA ca.crt -CAkey ca.key\
-CAcreateserial -CAserial ca.srl
```

Le certificat signé est le fichier "servwiki.crt". La sortie de la commande attendue est la suivante :

```
Signature ok
subject=/C=FR/ST=CORSE/L=Ajaccio/O=LLB/OU=BTSINFO/CN=wiki.domain1.org
Getting CA Private Key
Enter pass phrase for ca.key:
```

Vous avez maintenant un certificat pour votre serveur se nommant servwiki.crt.

```
less servwiki.crt

-----BEGIN CERTIFICATE-----
MIICVDCCAb0CAQEwDQYJKoZIhvcNAQEEBQAwdDELMAkGA1UEBhMCRlIxFTATBgNV
BAgTDENvcnNlIGR1IFN1ZDEQMA4GA1UEBxMHQWphY2NpbzEMMAoGA1UEChMDTExC
MREwDwYDVQQLEwhCVFMgSU5GTzEbMBkGA1UEAxMSc2VydmV1ci5idHNpbmZvLmZy
MB4XDTA0MDIwODE2MjQyNloXDTA0MDMwOTE2MjQyNlowcTELMAkGA1UEBhMCRlIx
FTATBgNVBAgTDENvcnNlIGR1IFN1ZDEQMA4GA1UEBxMHQWphY2NpbzEMMAoGA1UE
ChMDTExCMREwDwYDVQQLEwhCVFMgSU5GTzEYMBYGA1UEAxMPcHJvZi5idHNpbmZv
LmZyMIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDSUagxPSv3LtgDV5sygt12
kSbN/NWP0QUiPlksOkF2NkPfwW/mf55dD1hSndlOM/5kLbSBo5ieE3TgikF0Iktj
BWm5xSqewM5QDYzXFt031DrPX63Fvo+tCKTQoVItdEuJPMahVsXnDyYHeUURRWLW
wc0BzEgFZGGw7wiMF6wt5QIDAQABMA0GCSqGSIb3DQEBBAUAA4GBALD640iwKPMf
pqdYtfvmLnA7CiEuao60i/pzVJE2LIXXXbwYjNAM+7Lov+dFT+b5FcOUGqLymSG3
kSK6OOauBHItgiGI7C87u4EJaHDvGIUxHxQQGsUM0SCIIVGK7Lwm+8e9I2X0G2GP
9t/rrbdGzXXOCl3up99naL5XAzCIp6r5
-----END CERTIFICATE-----
```

Il faut maintenant **installer le certificat de l'autorité de certification dans chaque navigateur client**. C'est lui qui va valider le certificat reçu par le client lors de la requête https://.

20.3.4. Installation du certificat d'autorité de certification

Pour installer sur votre navigateur le certificat de l´autorité de certification (fichier ca.crt) :

- "Arrangez−vous" pour avoir le certificat disponible à partir du client (disquette, ftp, ssh, répertoire partagé...)
- Sous Windows, un clic droit sur le fichier devrait vous permettre de lancer l'assistant d'installation du certificat dans Internet Explorer. **Vous devez l'installer en tant qu'autorité de certification.**Vérifiez en suite que le certificat s'y trouve bien sous le nom de "cert_CA".
- sur Mozilla :
    - Edition/préférence/Confidentialité et Sécurité/Certificats
    - Bouton de commande : gestion des certificats
    - Onglet : autorité
    - Bouton de commande : importer
    - etc...

Il ne reste plus maintenant qu'à configurer apache.

20.4. Étape 2 : configuration d'Apache

Vérifiez dans un premier temps que le module ssl est bien pris en compte (non commenté dans /etc/apache/modules.conf).

```
cat /etc/apache/modules.conf | grep ssl
LoadModule ssl_module /usr/lib/apache/1.3/mod_ssl.so
```

Si le module est commenté, décommentez−le et relancez le serveur.

Éditez le fichier de configuration d'apache /etc/apache/httpd.conf et vérifiez et/ou modifiez et/ou rajoutez :

```
#On vérifie que le serveur écoute bien sur le port 443
  Listen *:443
        ...
#On installe le certificat de l'autorité de certification
 SSLCertificateFile    /etc/apache/certs/ca.crt
 SSLCertificateKeyFile /etc/apache/certs/ca.key

#On configure notre virtualHost (à adapter...)
 <VirtualHost wiki.domain1.org:443>
 ServerName  wiki.domain1.org
 DocumentRoot /le/chemin

#On installe le certificat pour ce serveur Web
 SSLCertificateFile    /etc/apache/servwiki.crt
 SSLCertificateKeyFile /etc/apache/certs/servwiki.key

 </VirtualHost>
```

Si vous voulez ne pouvoir accéder à ce serveur que par le protocole HTTPS, il faut supprimer les lignes correspondante à ce virtualhost écoutant sur le port 80 dans ce fichier.

Relancez le serveur.

Vous pouvez maintenant "passer" à la phase de test...

À partir d'un navigateur sur votre poste client :

```
https://wiki.domain1.org
```

Votre page devrait s'afficher normalement.

Si vous avez une alerte de sécurité, c'est que vous avez mal réalisé une étape. Selon l'alerte, reprenez l'étape correspondante.

Vous pouvez maintenant revoir, dans la partie cours, le schéma concernant le protocole SSL et l'application HTTP (fig 1.7).

Chapitre 21. Installation d'un serveur SAMBA

Partage de ressources sous GNU/Linux pour les clients GNU/Linux ou Windows avec le protocole SMB.

21.1. Introduction

Avec SAMBA vous allez mettre en place un service de partage de disque pour des clients réseau. Ceux−ci peuvent être sous Linux ou sous Windows. Nous verrons surtout la configuration du service serveur sous Linux, et la configuration des clients sous Windows.

Samba est un produit assez populaire. Il existe de plus en plus d'outils de configuration en environnement graphique qui simplifient les tâches sur un serveur en exploitation (partage de ressources, création de comptes utilisateurs). Comme nous n'en sommes pas là, nous allons réaliser les opérations manuellement.

Vous devez savoir ce qu'est un domaine Microsoft, un groupe de travail, comment fonctionne la résolution de nom NetBIOS avec le protocole NetBIOS, ce qu'est un serveur WINS, un serveur d'authentification (contrôleur de domaine).

21.2. Éléments d'installation et de configuration de SAMBA

21.2.1. Environnement de SAMBA

SAMBA est installé sur la Freeduc−sup. Si vous n'utilisez pas cette distribution, installez les paquets. Il ne devrait normalement pas y avoir de problèmes de dépendances.

Le paquet installe principalement samba et samba−common :

- le programme **nmbd** qui assure la résolution de nom NetBIOS et **smbd** qui assure le partage de ressource SMB/CIFS dans /`usr/sbin`,
- le script d'initialisation dans `/etc/init.d`,
- un fichier de configuration `/etc/samba/smb.conf`,
- une documentation complète dans `/usr/share/doc`,
- le service de journalisation (log) dans `/var/log/samba`,
- des outils comme smbpasswd pour la création des comptes `samba` et `nmblookup` pour vérifier le fonctionnement de la résolution de noms NetBIOS.

La commande **`dpkg−reconfigure samba`** vous demande si samba doit être lancé en mode autonome, choisissez « oui », si un fichier `/etc/samba/smbpasswd` doit être créé, choisissez également « oui ». La dernière option vous permet d'avoir une base de données de compte créée automatiquement à partir de la base de compte du fichier `/etc/passwd`.

Faites tout de suite une sauvegarde du fichier `/etc/smb.conf`.

Dans la suite du document, vous remplacerez `$NOM_HÔTE` par le nom d'hôte de votre machine qui servira également de nom NetBIOS.

21.2.2. Le fichier de configuration sous Linux

Voici le fichier de configuration qui nous servira de base de travail. Il va permettre de :

- définir `$NOM_HÔTE` comme contrôleur de domaine,
- mettre en place l'authentification des utilisateurs,
- partager des disques et une imprimante pour un client Windows,
- partager du dossier personnel d'un utilisateur sous Linux comme étant son répertoire personnel sous Windows.

Le fichier de configuration comprend essentiellement deux parties :

- une partie << générale >> qui définit le comportement général du serveur et la stratégie adoptée pour les services communs (CPD, mode d'authentification, service WINS),
- une partie `share`, qui définit les ressources partagées et les permissions d'accès.

21.2.3. Les étapes de la configuration du serveur

Nous allons réaliser les opérations suivantes :

- Vérifier et valider le fichier de configuration,
- Déclarer les ressources partagées,
- Créer un compte d'utilisateur pour SAMBA.

Il n'y aura plus qu'à tester la configuration à partir d'un client.

Attention, un compte système n'est pas un compte SAMBA. Faites bien la distinction entre les deux.

21.2.4. Première étape − Installer le fichier de configuration

Configurer l'environnement de samba par le fichier `/etc/samba/smb.conf` et activez le service avec la commande **`/etc/init.d/samba start`**. Cette opération doit être réalisée chaque fois que le fichier de configuration est modifié. Vérifiez la configuration à l'aide de la commande **`testparm | more`**.

Corrigez les erreurs éventuelles de configuration.

21.2.5. Deuxième étape − Déclarer les ressources partagées

Cette opération est réalisée dans la partie `share` du fichier `smb.conf`. Chaque fois que vous ajoutez ou modifiez une ressource, relancez le service serveur.

Pour l'authentification sur un domaine, vous devez créer un répertoire `netlogon`, par exemple :

```
mkdir −p /home/samba/netlogon
```

et déclarer la ressource dans la section `share` du fichier `/etc/smb.conf`.

21.2.6. Troisième étape − Créer un compte d'utilisateur autorisé

La création d'un compte d'utilisateur SAMBA et de son mot de passe est réalisée à l'aide de la commande **`smbpasswd`**.

```
smbpasswd −a MonCompte MonMotDePasse
```

ajoute le compte SAMBA `MonCompte` avec le mot de passe `MonMotDePasse` dans le fichier `/etc/samba/smbpasswd`.

Voir **`man smbpasswd`** ou **`smbpasswd −−help`** pour le mode d'utilisation de la commande. Le compte système doit être préalablement créé avec la commande unix **`adduser`**.

Trois remarques :

- Les manipulations peuvent paraître fastidieuses si vous avez un grand nombre de comptes utilisateurs.
- Si vous disposez de nombreux comptes d'utilisateurs sur votre système Linux, vous disposez d'un script qui permet de créer automatiquement les comptes SAMBA à partir du fichier `/etc/passwd`. Voir **`man mksmbpasswd`** pour le mode d'utilisation de la commande. Il est également possible de créer sans difficulté un script qui, a partir d'un fichier texte crée les comptes systèmes et les comptes SAMBA.

Toutes les indications sont dans la documentation de SAMBA.

21.2.7. La configuration d'un client Windows

La configuration du client Windows ne doit pas poser de difficulté.

Configurez le client pour les réseaux Microsoft afin que l'utilisateur soit authentifié par le serveur de votre domaine. Ici le contrôleur de domaine est le serveur SAMBA. Vérifiez la configuration du protocole TCP/IP, relancez la machine. Vous pourrez vous authentifier sur le serveur SAMBA.

Toutes les commandes **`net use`**, **`net share`** ou les outils comme le voisinage réseau vous permettent d'accéder aux ressources du serveur (disques partagés, imprimantes, disque personnels).

Un problème à éviter :

Le compte utilisateur SAMBA dispose de moins de privilèges que le compte root. Si vous partagez un disque et que vous faites les manipulations sous le compte root, faites attention aux droits, car si root est propriétaire (chmod 700), le client SAMBA ne pourra pas accéder au disque.

21.3. Annexe : exemple de fichier de configuration de SAMBA :

```
Annexe 1 : Exemple de fichier de configuration de Samba :
Le fichier ci−dessous est simplifié, vous trouverez de nombreuses autres options
dans la documentation.
================================================================================
# Date: 2003/10/03 12:48:57

# Global parameters
[global]
        workgroup = INFOGESTION
        netbios name = main_serveur
        server string = %h server (Samba %v)
        encrypt passwords = Yes
        passwd program = /usr/bin/passwd %u
        passwd chat = *Enter\snew\sUNIX\spassword:* %n\n *Retype\snew\sUNIX\spassword:* %n\n .
```

```
        syslog = 0
        max log size = 1000
        socket options = IPTOS_LOWDELAY TCP_NODELAY SO_SNDBUF=4096 SO_RCVBUF=4096
        logon script = logon.cmd
        logon path = \\%N\profiles\%u
        logon drive = h:
        logon home = \\homeserver\%u
        domain logons = Yes
        os level = 33
        preferred master = True
        domain master = True
        dns proxy = No
        wins server = localhost
        invalid users = root

[netlogon]
        path = /home/samba/netlogon
        read only = yes
        browsable = no


[homes]
        comment = Home Directories
        read only = No
        create mask = 0700
        directory mask = 0700
        browseable = No

[partage]
        comment = Ressource partagées
        path = /tmp
        read only = No
        create mask = 0700
        printable = Yes
        browseable = No

[tmp]
        comment = Partage
        path = /tmp
        create mask = 0700
        directory mask = 0700
        guest ok = Yes

[printers]
        comment = All Printers
        path = /tmp
        create mask = 0700
        printable = Yes
        browseable = No
```

==========

Chapitre 22. Travaux pratiques : installation d'un serveur SAMBA

Partage de ressources sous GNU/Linux pour les clients GNU/Linux ou Windows avec le protocole SMB.

Ce document décrit comment utiliser un serveur samba comme serveur d'identification et d'authentification pour des clients Windows. Le serveur simule un contrôleur de domaine NT4 Server ou 2000.

Vous utiliserez 2 postes en réseau. Le premier est sous Linux, le second sous Windows. On désire installer et configurer le service de partage de fichiers SAMBA sous Linux. Le client Windows doit permettre l'identification des utilisateurs sur le serveur en utilisant les mots de passe cryptés.

Cet atelier permet la mise en oeuvre du protocole SMB. Il permet également d'envisager la mise en place du partage de fichiers et d'imprimantes.

22.1. Déroulement des opérations

Les opérations vont se dérouler en 6 étapes :

- Configuration du fichier `/etc/smb.conf` et démarrage des services,
- Création d'un compte utilisateur,
- Création du fichier d'authentification pour SAMBA /etc/smbpasswd,
- Création de ressources disques partagées en lecture et en lecture/écriture,
- Configuration du client Windows 9x,
- Procédure de test.

22.2. Configuration du fichier smb.conf et démarrage des services

Le fichier de configuration smb.conf est dans le répertoire `/etc/samba`. Faites une copie de sauvegarde de ce fichier puis ouvrez l'original avec un éditeur. Modifiez−le afin que les utilisateurs puissent accéder au répertoire `/tmp` du serveur en `rw` et à leur répertoire personnel en `rw` également.

Vous utiliserez et adapterez l'exemple donné dans la fiche de cours.

22.3. Création d'un compte utilisateur

Vous allez tout d'abord :

- créer le compte système
- créer le compte samba.

Créez un compte système à l'aide de la commande **`adduser`**.

Pour ce compte système vous créerez un compte samba à l'aide de la commande **`smbpasswd`**.

22.4. Vérification de la configuration sur le serveur SAMBA

Démarrez le service. Vous pouvez utiliser la commande **`testparm`** pour valider la configuration du serveur. Vérifiez également la table des processus et les traces dans le fichier log.

Le fichier `DIAGNOSIS.txt` de la documentation de samba, donne une procédure en 10 points pour vérifier que tout fonctionne. Localisez ce fichier, (en général dans `/usr/doc` ou dans `/usr/share/doc/samba`) ouvrez−le avec un éditeur et réalisez la procédure de test qui y est décrite.

22.5. Procédure de test à partir d'un client Linux

Si le serveur fonctionne correctement et que vous utilisez une Freeduc−Sup, vous pouvez utiliser le module externe (plugin) `smb` directement à partir de konqueror.

Lancez konqueror à partir d'un autre client linux et utilisez l'url suivante :
`smb://@IP_Du_Serveur/un_partage_Samba_public` ou **`smb://loginname@Nom_Du_Serveur/`**. Vous devriez avoir une fenêtre équivalente à celle donnée ci−dessous.

Figure 22−1. Accès à un serveur SAMBA à partir d'un client Linux

En ligne de commande (man smbmount smbclient mount), il est fortement déconseillé de mettre le mot de passe en clair, car outre le fait qu'il soit visible lors de sa saisie sur la console, il apparaitra en clair dans la liste des processus avec un simple ps. Pour un montage automatique du système de fichiers partagé dans fstab par exemple, on utilisera un fichier credentials avec l'option du même nom et on veillera à positionner les droits minimum.

```
[jmj@pastorius jmj]$ smbmount //eminem/homes ~/smbmnt −o username=jmj
Password:
[root@pastorius root]# mount −t smbfs //eminem/outils /mnt/admin −o username=admin
Password:
[jmj@pastorius jmj]$ smbclient //eminem/homes −U jmj
Password:
Domain=[EMINEM] OS=[Unix] Server=[Samba 3.0.7]
smb: \>
```

Concernant la dernière commande, après le prompt smb, vous pouvez ensuite taper help pour obtenir l'aide.

Exemple d'utilisation de tar avec les partages de fichiers samba :

```
smbclient //monpc/monpartage "" −N −Tc sauvegarde.tar users/docs
```

22.6. Procédure de test à partir d'un client Windows

La procédure de tests sera réalisée à partir d'un client w98. Pour d'autres types de clients, il sera peut être nécessaire de créer des comptes d'ordinateurs, ou adapter la configuration du serveur SAMBA. Il faudra se référer à la documentation située en général dans `/usr/share/doc/samba`.

Configurez votre client Windows pour qu'il puisse faire partie de votre domaine NT (Panneau de configuration, icône Réseau) déclaré sur votre serveur Linux.

Au démarrage du PC, vous devez avoir, sur le client, une fenêtre qui vous demande de vous identifier dans le domaine défini. Vérifiez l'accès.

Une fois la session ouverte vous devez pouvoir utiliser les outils et commandes suivantes :

- Explorateur,
- Voisinage réseau,
- Démarrer, exécuter, \\NomDuServeurSamba
- Consultez sur le serveur les fichiers `/var/log/samba/log.%m`
- Vérifiez les accès en lecture / écriture sur les espaces disques partagés.

Modification de l'environnement serveur

Créez sur le serveur les espaces supplémentaires `/mnt/apps` et `/mnt/partage`. Le premier est en lecture uniquement, le deuxième en lecture / écriture. Modifiez `smb.conf`, relancez le service serveur, testez les accès.

22.7. Automatisation de création de compte.

On donne, dans un fichier texte << personnes >>, une liste de personnes. Le fichier a la structure suivante :

```
Nom Prénom
```

par exemple :

```
TUX Junior
TUX Tuxinette
TUX Padre
...
```

En général un fichier d'importation n'est pas aussi simple car on peut avoir des prénoms composés, ou des noms comprenant des << espaces >>. Les champs sont distingués par des séparateurs comme un point−virgule par exemple. Il faudra dans ce cas traiter différement le fichier.

Le principe est simple :

- On crée un script qui va créer automatiquement les comptes systèmes,
- Le compte est représenté par la concaténation du nom et du prénom
- Le mot de passe par défaut est la concaténation du nom et du prénom
- Les groupes sont préalablement créés.

Vous pouvez avoir des différences entre les distributions type RedHatetc... ou Debian. La commande **`passwd`** par exemple ne supporte pas l'option `−−stdin`. Vous avez donc ci−dessous deux esmples d'applications qui tiennent compte de ces différences. On donne le script suivant qui crée les comptes systèmes :

```
cat persons | while true ; do
# Si la ligne est vide on quitte,
# il ne faut donc pas de ligne vide dans le fichier
read ligne
        if [ "$ligne" == "" ] ; then
        exit 0
        fi

#On récupère les paramètes prénom, nom
set −− $ligne
        echo $2 $1
useradd $1$2 −G $1$2
echo $1$2 | (passwd −−stdin $1$2)
done
```

Testez et vérifiez le fonctionnement du script. Modifiez le script pour qu'il crée également les comptes SAMBA.

Version pour Debian

```
cat persons | while true ; do
read ligne
        if [ "$ligne" == "" ] ; then
        exit 0
        fi
set −− $ligne
        echo $2 $1
addgroup $1$2
useradd  $1$2 −G $1$2
echo $1$2:$1$2 | chpasswd
done
```

22.8. Administration graphique

Vous pouvez utiliser des outils graphiques d'administration de SAMBA comme swat ou webmin. Pour utilisez swat, décommentez la ligne dans `/etc/inetd.conf` :

```
swat stream tcp nowait root \
  /usr/sbin/tcpd /usr/sbin/swat
```

Activez les services apache et inetd. Ouvrez le navigateur et saisissez :

```
http://localhost:901
```

Ouvrez une session avec le compte root.

Pour utiliser webmin, activez les services apache et webmin. Dans un navigateur allez sur l'url :

```
https://localhost:10000
```

Ouvrez une session avec le compte root et comme mot de passe << knoppix >> qui a été mis par défaut sur la Freeduc−Sup. Dans l'onglet Serveurs, vous pourrez administrer votre serveur SAMBA.

Chapitre 23. Eléments de cours sur le service DHCP

L'adressage IP dynamique − Fiche de cours

L'adressage IP dynamique − Fiche de cours

Eléments de cours sur le service DHCP

23.1. Résumé

Présentation, installation et configuration d'un serveur DHCP

Mots clés : << Serveur DHCP >>, << Agent relais DHCP >>

Description et objectifs de la séquence

L'atelier propose

- d'installer un serveur DHCP sous Linux,
- d'installer un client DHCP sous Linux
- d'installer un client DHCP sous Windows
- de réaliser une phase de test avec les commandes winipcfg et ipconfig de Windows

Les éléments sur l'analyse de trame, notamment les trames bootp, seront retraités lors des TP sur la métrologie.

23.2. Rôle d'un service DHCP

Un serveur DHCP (*Dynamic Host Configuration Protocol*) a pour rôle de distribuer des adresses IP à des clients pour une durée déterminée.

Au lieu d'affecter manuellement à chaque hôte une adresse statique, ainsi que tous les paramètres tels que (serveur de noms, passerelle par défaut, nom du réseau), un serveur DHCP alloue à un client, un bail d'accès au réseau, pour une durée déterminée (durée du bail). Le serveur passe en paramètres au client toutes les informations dont il a besoin.

Tous les noeuds critiques du réseau (serveur de nom primaire et secondaire, passerelle par défaut) ont une adresse IP statique ; en effet, si celle−ci variait, ce processus ne serait plus réalisable.

Ce processus est mis en oeuvre quand vous ouvrez une session chez un fournisseur d'accès Internet par modem. Le fournisseur d'accès, vous alloue une adresse IP de son réseau le temps de la liaison. Cette adresse est libérée, donc de nouveau disponible, lors de la fermeture de la session.

23.2.1. Pourquoi mettre en place un réseau TCP/IP avec des adresses IP dynamiques

L'affectation et la mise à jour d'informations relatives aux adresses IP fixes peuvent représenter une lourde tâche. Afin de faciliter ce travail et de simplifier la distribution des adresses IP, le protocole DHCP offre une configuration dynamique des adresses IP et des informations associées.

Avantages de DHCP dans l'administration d'un réseau ?

1. Le protocole DHCP offre une configuration de réseau TCP/IP fiable et simple, empêche les conflits d'adresses et permet de **contrôler** l'utilisation des adresses IP de façon **centralisée**. Ainsi, si un paramètre change au niveau du réseau, comme, par exemple l'adresse de la passerelle par défaut, il suffit de changer la valeur du paramètre au niveau du serveur DHCP, pour que toutes les stations aient une prise en compte du nouveau paramètre dès que le bail sera renouvelé. Dans le cas de l'adressage statique, il faudrait manuellement reconfigurer toutes les machines.

2. **économie d'adresse** : ce protocole est presque toujours utilisé par les fournisseurs d'accès Internet qui disposent d'un nombre d'adresses limité. Ainsi grâce à DHCP, seules les machines connectées en ligne ont une adresse IP. En effet, imaginons un fournisseur d'accès qui a plus de 1000 clients. Il lui faudrait 5 réseaux de classe C, s'il voulait donner à chaque client une adresse IP particulière. S'il se dit que chaque client utilise en moyenne un temps de connexion de 10 mn par jour, il peut s'en sortir avec une seule classe C, en attribuant, ce que l'on pourrait appeler des "jetons d'accès" en fonction des besoins des clients.
3. Les postes itinérants sont plus faciles à gérer
4. Le changement de plan d'adressage se trouve facilité par le dynamisme d'attribution.

Avec DHCP, il suffit d'attribuer une adresse au serveur. Lorsqu'un ordinateur client DHCPdemande l'accès au réseau en TCP−IP son adresse est allouée dynamiquement à l'intérieur d'une plage d'adresses définie sur le serveur .

L'administrateur de réseau contrôle le mode d'attribution des adresses IP en spécifiant une **durée de bail** qui indique combien de temps l'hôte peut utiliser une configuration IP attribuée, avant de devoir solliciter le renouvellement du bail auprès du serveur DHCP.

L'adresse IP est libérée automatiquement, à l'expiration du bail, pour un ordinateur client DHCP retiré d'un sous−réseau, et une nouvelle adresse est automatiquement définie pour ce dernier, lorsque cet ordinateur est reconnecté à un autre sous−réseau. Ni l'utilisateur ni l'administrateur de réseau n'ont besoin de fournir de nouvelles informations relatives à la configuration. Cette fonctionnalité est non négligeable, tant pour les utilisateurs de portables fixés ou non à différentes stations d'accueil que pour les ordinateurs fréquemment déplacés.

L'inconvénient :

Le client utilise des trames de **broadcast** pour rechercher un serveur DHCP sur le réseau, cela charge le réseau. Si vous avez une entreprise avec plusieurs centaines de personnes qui ouvrent leur session le matin à 8 h ou l'après midi à 14 h, il peut s'en suivre de graves goulets d'étranglement sur le réseau. L'administrateur devra donc réfléchir sérieusement à l'organisation de son réseau.

23.2.2. Protocole DHCP(Dynamic Host Configuration Protocol)

Le protocole **DHCP** (*Dynamic Host Configuration Protocol*) (RFC 1533 1534) est une extension de BOOTP (RFC 1532), il fournit une **configuration dynamique** des adresses IP et des informations associées aux ordinateurs configurés pour l'utiliser (clients DHCP). Ainsi chaque hôte du réseau obtient une configuration IP dynamiquement au moment du démarrage, auprès du **serveur DHCP**. Le serveur DHCP lui attribuera notamment une adresse IP, un masque et éventuellement l'adresse d'une passerelle par défaut. Il peut attribuer beaucoup d'autres paramètres IP notamment en matière de noms (l'adresse des serveurs DNS, l'adresse des serveurs WINS)

23.3. Fonctionnement de DHCP

Un client DHCP est un ordinateur qui demande une adresse IP à un serveur DHCP.

Comment, alors, un client DHCP, qui utilise le protocole TCP/IP mais qui n'a pas encore obtenu d'adresse IP par le serveur, peut−il communiquer sur le réseau ?

23.3.1. Attribution d'une adresse DHCP

Lorsqu'un client DHCP initialise un accès à un réseau TCP/IP, le processus d'obtention du bail IP se déroule en 4 phases :

1 − Le client émet un message de demande de bail IP (DHCPDISCOVER) qui est envoyé sous forme d'une diffusion sur le réseau avec adresse IP source 0.0.0.0 et adresse IP destination 255.255.255.255 et adresse MAC.

2 − Les serveurs DHCP répondent en proposant une adresse IP avec une durée de bail et l'adresse IP du serveur DHCP (DHCOFFER)

3 − Le client sélectionne la première adresse IP (s'il y a plusieurs serveurs DHCP) reçue et envoie une demande d'utilisation de cette adresse au serveur DHCP (DHCPREQUEST). Son message envoyé par diffusion comporte l'identification du serveur sélectionné qui est informé que son offre a été retenue ; tous les autres serveurs DHCP retirent leur offre et les adresses proposée redeviennent disponibles.

4 − Le serveur DHCP accuse réception de la demande et accorde l'adresse en bail (DHCPACK), les autres serveurs retirent leur proposition.

Enfin le client utilise l´adresse pour se connecter au réseau.

Vous trouverez des éléments très précis sur le protocole DHCP dans les pages du manuel de Linux. (dhcp3d, `dhcpd.conf` et `dhclient.conf`).

23.3.2. Renouvellement de bail IP

Lorsqu'un client redémarre, il tente d'obtenir un bail pour la même adresse avec le serveur DHCP d'origine, en émettant un DHCPREQUEST. Si la tentative se solde par un échec, le client continue à utiliser la même adresse IP s'il lui reste du temps sur son bail.

Les clients DHCP d'un serveur DHCP Windows (NT/2000) tentent de renouveler leur bail lorsqu'ils ont atteint 50% de sa durée par un DHCPREQUEST. Si le serveur DHCP est disponible il envoie un DHCPACK avec la nouvelle durée et éventuellement les mises à jour des paramètres de configuration.

Si à 50% le bail n'a pu être renouvelé, le client tente de contacter l'ensemble des serveurs DHCP (diffusion) lorsqu'il atteint 87,5% de son bail, avec un DHCPREQUEST, les serveurs répondent soit par DHCPACK soit par DHCPNACK (adresse inutilisable, étendue désactivée...).

Lorsque le bail expire ou qu'un message DHCPNACK est reçu le client doit cesser d'utiliser l'adresse IP et et demander un nouveau bail (retour au processus de souscription). Lorsque le bail expire et que le client n'obtient pas d'autre adresse la communication TCP/IP s'interrompt.

Remarque : Si la demande n'aboutit pas et que le bail n´est pas expiré, le client continue à utiliser ses paramètres IP.

23.4. Configuration d'un serveur DHCP

Définir une plage d'adresses qui peuvent être louées à des hôtes qui en font la demande.

En général, on donne :

- Une adresse de début (la première qui sera attribuée)
- Une adresse de fin (la dernière)
- Une ou plusieurs plages d'adresses à exclure de la location (ceci permet de faire cohabiter un modèle de configuration IP dynamique avec un modèle statique)
- Un masque de sous−réseau

Tous ces éléments sont attribués pour une durée de bail à fixer. Si, au bout de cette durée, l'hôte ne sollicite pas à nouveau une adresse au serveur, cette adresse est jugée disponible pour un autre hôte.

Il est possible de connaître les baux actifs (les locations en cours), on voit alors à quelle adresse MAC est attribuée une adresse IP.

23.5. Mise en oeuvre d'un client DHCP

Les clients DHCP doivent être configurés seulement après la configuration du serveur. Etant donné qu'un ordinateur ne peut fonctionner simultanément comme client et serveur DHCP, l'ordinateur fonctionnant comme **serveur DHCP doit être configuré avec une adresse IP fixe**.

Lors de la configuration du client DHCP, il faut cocher la case « Obtenir une adresse IP depuis un serveur DHCP » dans la fenêtre des propriétés de Microsoft TCP/IP. Il n'est pas nécessaire alors de préciser une adresse IP ou un masque de sous−réseau.

Voici, par exemple, la configuration TCP/IP d'un ordinateur Windows XP qui sollicite une configuration IP auprès d'un serveur DHCP :

Figure 23−1. Client DHCP sous Windows XP

Les commandes IPCONFIG (Windows NT/200*x*) et **Winipcfg** (Windows 9*x*) permettent de visualiser la configuration IP complète du poste de travail :

Figure 23−2. WinIPCFG sous Windows 9x

IPCONFIG

ipconfig /all : affiche les paramètres IP complets, cela va nous permettre de vérifier la bonne affectation d'adresse.

ipconfig /renew : déclenche l'envoi d'un message DHCPREQUEST vers le serveur DHCP pour obtenir des options de mise à jour

ipconfig /release : déclenche l'envoi d'un message DHCPRELEASE pour abandonner le bail. Commande utile lorsque le client change de réseau.

23.6. Rôle de l'agent de relais DHCP

Comme les clients contactent les serveurs DHCP à l'aide d'une diffusion, dans un inter−réseau, vous devrez théoriquement installer un serveur DHCP par sous−réseau. Si votre routeur prend en charge la RFC 1542, il peut faire office d'agent de relais DHCP, et ainsi relayer les diffusions de demande d'adresse IP des clients DHCP dans chaque sous−réseau.

Si votre routeur ne prend pas en charge la RFC 1542, une machine serveur peut être configurée comme agent de relais DHCP, il suffira de lui spécifier l'adresse du serveur DHCP. Les demandes des clients DHCP seront relayées vers le serveur DHCP par l'agent de relais DHCP qui transmettra les offres aux clients.

Figure 23−3. Agent de relais DHCP dans un réseau routé

Chapitre 24. Travaux pratiques : installation d'un serveur DHCP

24.1. Indications pour la réalisation du TP

L'atelier propose

- d'installer un serveur DHCP sous Linux,
- d'installer un client DHCP sous Linux
- d'installer un client DHCP sous Windows
- de réaliser une phase de test avec les commandes winipcfg et ipconfig de Windows

Matériel nécessaire :

Deux machines en dual boot Linux / Windows en réseau.

Les éléments sur l'analyse de trame, notamment les trames bootp, seront retraités lors des TP sur la métrologie.

24.1.1. Installation du serveur

Les paquets sont déjà installés.

Attention : vous pouvez avoir sur votre distribution, plusieurs serveurs DHCP.

dhcpxd est conforme à la RFC 2131. Il fournit un exemple de configuration assez détaillé.

dhcp3, intègre l'inscription auprès d'un DNS Dynamique. C'est ce package que nous allons utiliser dans le TP. Par contre si vous n'avez pas de DNS dynamique sur le réseau, vous devrez mettre en entête du fichier `dhcpd.conf`, la ligne :

```
ddns−update−style none;
```

24.1.2. Configuration du serveur

La configuration consiste à créer 2 fichiers :

- `/etc/dhcp3/dhcpd.conf`, ce fichier sert à la configuration même du serveur (plage d'adresses, paramètres distribués),
- `/var/lib/dhcp3/dhcpd.leases`, ce fichier va servir à l'inscription des clients. Chaque client DHCP, génère l'écriture d'un enregistrement dans ce fichier. Cela permet le suivi, les statistiques de l'activité du serveur.

24.1.2.1. Le fichier de configuration dhcpd.conf

On n'abordera pas tous les paramètres. Vous trouverez un exemple de fichier commenté qui permet de réaliser cet atelier. Vous pouvez créer ce fichier avec un éditeur.

```
$>more dhcpd.conf

# ici il s'agit du réseau 192.168.0.0
subnet 192.168.0.0 netmask 255.255.255.0 {

#La plage d'adresse disponible pour les clients
range 192.168.0.10 192.168.0.20;

# Les clients auront cette adresse comme passerelle par défaut
option routers    192.168.0.254;

# Ici c'est le serveur de noms, on peut en mettre plusieurs
option domain−name−servers   192.168.0.1;
```

```
# Enfin on leur donne le nom du domaine
option domain-name   "freeduc-sup.org";

# Et l'adresse utilisée pour la diffusion
option broadcast-address 192.168.0.255;

# Le bail à une durée de 86400 s par défaut, soit 24 h
# On peut configurer les clients pour qu'ils puissent demander
# une durée de bail spécifique
default-lease-time  86400;

# On le laisse avec un maximum de 7 jours
max-lease-time 604800;

#Ici on désire réserver des adresses à des machines
group {
#use-host-decl-names indique que toutes les machines dans l'instruction « group »
# auront comme nom, celui déclaré dans l'instruction host.
use-host-decl-names true ;

# ici définir les machines
host m1 {
hardware ethernet 00:80:23:a8:a7:24;
fixed-address 192.168.0.125;
} # End m1

host m2 {
hardware ethernet a0:81:24:a8:e8:3b;
fixed-address 192.168.0.126;
} # End m2

    } # End Group
}     # End dhcp.conf
```

24.1.2.2. Création d'un fichier d'inscription

Ce fichier doit parfois être créé, sans quoi le serveur DHCP ne pourra pas démarrer. Il suffit de créer un fichier vide. Pour cela, saisissez la commande **touch */var/lib/dhcp3/dhcpd.leases***. Le fichier est créé. Voici ce qu'il peut contenir après l'inscription du premier client :

```
[root@master /etc]# more  /var/lib/dhcp3/dhcpd.leases

lease 192.168.0.10 {

 starts 1 2002/12/14 18:33:45;

 ends 1 2002/12/14 18:34:22;

 hardware ethernet 00:40:33:2d:b5:dd;

 uid 01:00:40:33:2d:b5:dd;

 client-hostname "CHA100";

}
```

On distingue les informations suivantes : Début du bail, Fin du bail, adresse MAC du client, le nom d'hôte du client. Attention ce nom est différent du nom Netbios utilisé sur les réseaux Microsoft.

24.1.2.3. Activation du serveur

Le serveur est configuré, il n'y a plus qu'à le mettre en route. Utilisez la commande suivante pour arrêter ou activer le service : **/etc/init.d/dhcpd3 start | stop**.

Le script lance le serveur en mode daemon. Vous pouvez le lancer en avant plan avec la commande **dhcpd3 -d**. Cela permet de voir les messages et déterminer s'il y a des dysfonctionnement éventuels.

```
root@master:/etc/dhcp3# dhcpd3 -d

Internet Software Consortium DHCP Server V3.0.1rc9

Copyright 1995-2001 Internet Software Consortium.

All rights reserved.

For info, please visit http://www.isc.org/products/DHCP

Wrote 1 leases to leases file.

Listening on LPF/eth0/00:d0:59:82:2b:86/192.168.0.0/24

Sending on   LPF/eth0/00:d0:59:82:2b:86/192.168.0.0/24

Sending on   Socket/fallback/fallback-net
```

CTRL C pour arrêter.

24.1.3. Installation des clients

24.1.3.1. Le client sous Windows

L'installation est assez simple si vous avez déjà une carte réseau et le protocole TCP/IP installé. Utilisez les commandes suivantes: Panneau de configuration/Réseau/Protocole TCP IP/Propriétés/Onglet "adresse ip"/ Cochez Obtenir automatiquement une adresse IP

La configuration est terminée, vous pouvez relancer la machine. Le client interrogera un serveur DHCP pour qu'il lui délivre un bail (sorte d'autorisation de séjour sur le réseau) contenant au minimum une adresse Ip et le masque correspondant .

24.1.3.2. Le client sous Linux

Vous allez réaliser une configuration manuelle

Allez dans le répertoire `/etc/network`, ouvrez le fichier **interfaces**. C'est ici qu'est la configuration des cartes installées sur la machine. Remplacez `static` par `dhcp` dans la configuration de l'interface `eth0`. Mettez tous les paramètres de cette interface (address, netmask, network....) en commentaire.

La configuration de la carte est terminée, vous pouvez tester en relançant le service réseau.

Vous pouvez egalement tester dynamiquement en ligne de commande:

```
root@m1:# dhclient eth0
```

24.1.4. Procédure de test

Sur Windows vous allez pouvoir utiliser (selon les versions) les commandes **`IPCONFIG`** et **`Winipcfg`**.

Utilisez **`ipconfig /?`** pour voir comment utiliser la commande

Vous pouvez utiliser l'interface graphique **`winipcfg`** sous Windows 9x uniquement. Allez dans Démarrer puis Exécuter et saisissez **`winipcfg`**. Une fois la fenêtre activée vous pouvez utiliser les fonctions de libération et de renouvellement de bail. Si vous avez plusieurs cartes sur la station, la liste déroulante << Cartes Ethernet Informations >> vous permet d'en sélectionner une.

24.2. Réalisation du TP

1. Installez un serveur DHCP minimal sous Linux et vérifiez le bon démarrage du service
2. Installez un client DHCP sous Linux, vérifiez le bon démarrage du service réseau et l'inscription dans le fichier dhcp.leases du serveur. Testez le renouvellement du bail. Il suffit de relancer le service réseau.
3. Installez un client DHCP sous Windows, vérifiez le bon démarrage du service réseau et l'inscription dans le fichier `dhcp.leases` du serveur. Testez le renouvellement du bail.
4. Modifiez l'étendue du serveur. Vérifiez le bon fonctionnement de la distribution d'adresses aux clients.
5. Modifiez la configuration du serveur afin qu'il distribue également l'adresse de la passerelle par défaut, l'adresse du serveur de nom. Testez la configuration.
6. Modifiez la configuration du serveur DHCP afin de réserver une adresse au client, vérifiez que le processus a bien fonctionné.

Chapitre 25. Travaux pratiques : installation d'un agent relais DHCP

25.1. Routeur et agent relais DHCP (RFC 1542)

Les trames arp, bootp ne traversent pas les routeurs. Sur un réseau segmenté par des routeurs il est donc impossible de servir tous les segments avec le même serveur DHCP. Il faut donc mettre un serveur DHCP sur chaque segment, ou alors utiliser un agent de relais DHCP.

Un agent relais DHCP relaie les messages DHCP échangés entre un client et un serveur DHCP situés sur des sous−réseaux différents.

Il est généralement installé sur un routeur pour pouvoir diriger les messages vers le serveur DHCP, mais ce n'est pas obligatoire. L'agent doit connaître l'adresse du serveur DHCP mais ne peut pas être lui−même client DHCP.

Serveur DHCP et agent de relais ont des adresses ip statiques. Le dialogue traverse le routeur et se fait en unicast.

Figure 25−1. Dialogue client DHCP, agent de relai DHCPet serveur DHCP

Après avoir envoyé une trame de broadcast, le client DHCP dialoque avec l'agent de relai DHCP en unicast (1). L'agent demande une adresse au serveur DHCP dont il connaît l'adresse (2). Le serveur retourne à l'agent une adresse (3) qui est donnée au client DHCP par l'agent (4).

Le principal problème du service DHCP est la mise à jour des serveurs de noms d'hôtes. Bind sous Linux permet la mise à jour dynamique (RFC 2136) grâce à un serveur "updater" qui peut ajouter ou supprimer dynamiquement des enregistrements de ressources. Il faut pour corriger cela installer un serveur DNS dynamique (DDNS) qui accepte les inscriptions des clients DHCP.

25.2. La maquette

Construisez une maquette en adaptant le schéma ci−dessous :

Figure 25−2. Maquette agent relais DHCP

Configurez les interfaces réseau de chaque machine, vérifiez que le routeur est opérationnel et que les paquets traversent bien.

25.3. Installation

Les packages : vous avez normalement les paquets sur la distribution Linux (client, serveur et agent de relais) :

Vous allez devoir installer ces produits pour les configurer et disposer de la documentation de ces produits.

Description:

Sous Linux il existe un agent relais DHCP (dhcprelay). Ce produit de l'ISC (Internet Software Consortium) permet de router des requêtes BOOTP et DHCP provenant de clients d'un réseau sur lequel il n'y a pas de serveur DHCP vers un autre segment sur lequel un serveur pourra répondre.

Mode de fonctionnement :

L'agent relais DHCP écoute les requêtes et les réponses BOOTP et DHCP. Quand une requête arrive, l'agent route la requête vers la liste de serveurs spécifiée sur la ligne de commande. Quand une réponse arrive d'un serveur, l'agent transmet la réponse (broadcast ou unicast cela dépend de la réponse) sur le segment d'où provenait la requête (broadcast) ou directement vers le client (unicast).

Ligne de commande :

```
dhcrelay3 [−p port] [−d] [−q] [−i if0 [... −i ifN ] ]server0 [ ...serverN ]
L'agent
− écoute  sur  toutes  les  interfaces  à  moins que certaines
  soient spécifiées   avec l'option −i,
− utilise, comme le  protocole  Bootp, le port 67 par défaut
  (voir /etc/services ) modifiable avec l'option −p,
− fonctionne  en avant−plan avec l'option −d (option debug),
  sinon en arrière−plan,
− n'affiche pas les informations de démarrage avec l'option −q,
− utilise  les  serveurs  spécifiés sur la ligne de commande
  server0, ...serveurN.
```

Vous allez installer le service serveur DHCP. Inspirez vous de l'exemple ci−dessous :

```
ddns−update−style none;

authoritative;

log−facility local7;
subnet 172.16.11.0 netmask 255.255.255.0 {
 range 172.16.11.2 172.16.11.253;
 option routers 172.16.11.254;
 #option domain−name−servers 192.168.90.77;
 #option domain−name "pat107.org";
  option broadcast−address 172.16.11.255;
 default−lease−time 1200;
 max−lease−time 2400;

}


subnet 172.16.12.0 netmask 255.255.255.0 {
 range 172.16.12.2 172.16.12.253;
 option routers 172.16.12.254;
 option broadcast−address 172.16.12.255;
```

```
 default-lease-time 1200;
 max-lease-time 2400;
}
```

Vous adapterez le fichier de configuration du serveur afin qu'il puisse délivrer des adresses pour les deux étendues d'adresses. Chaque segment représentant une étendue.

Vérifiez que le serveur démarre sans erreurs ni warning avec l'option "-d" (debug). Ne le lancez pas en mode daemon.

```
roo:~# dhcpd3 -d
Internet Systems Consortium DHCP Server V3.0.1rc14
Copyright 2004 Internet Systems Consortium.
All rights reserved.
For info, please visit http://www.isc.org/sw/dhcp/
Wrote 0 deleted host decls to leases file.
Wrote 0 new dynamic host decls to leases file.
Wrote 1 leases to leases file.
Listening on LPF/eth0/00:08:c7:19:25:75/172.16.11.0/24
Sending on   LPF/eth0/00:08:c7:19:25:75/172.16.11.0/24
Sending on   Socket/fallback/fallback-net
```

Vérifiez égalemement que le service est actif et que le port est bien ouvert avec la commande netstat :

```
Proto Recv-Q Send-Q Adresse locale Adresse distante  Etat        PID/Program name
udp    64232      0 0.0.0.0:67     0.0.0.0:*                   2093/dhcpd3
```

Installer l'agent relais DHCP et activer le service, toujours en mode debug. La commande "dpkg-reconfigure " peut également vous permettre de configurer l'agent et indiquer à quel serveur l'agent doit passer les requêtes.

Sur la ligne de commande, on indique à quel serveur doit s'adresser l'agent :

```
root@PAT109:~# dhcrelay3 172.16.11.1 -d
Internet Systems Consortium DHCP Relay Agent V3.0.1rc14
Copyright 2004 Internet Systems Consortium.
All rights reserved.
For info, please visit http://www.isc.org/sw/dhcp/
Listening on LPF/eth0/00:03:0d:08:63:bf
Sending on   LPF/eth0/00:03:0d:08:63:bf
Sending on   Socket/fallback
```

regardez, avec la commande netstat sur quel port par défaut l'agent attend les requêtes.

Démarrez le poste client et regardez le dialogue sur les consoles du serveur et de l'agent. Le client doit obtenir une adresse ip.

Voici un extrait du dialogue sur l'agent :

```
# Transmission de la requête cliente au serveur
forwarded BOOTREQUEST for 00:0a:e4:4e:64:4e to 172.16.11.1
# Transmission de l'adresse reçu du serveur au client
forwarded BOOTREPLY for 00:0a:e4:4e:64:4e to 172.16.12.10
```

Voici un extrait du dialogue sur le serveur :

```
# Le serveur reçoit une requête de l'agent
DHCPREQUEST for 172.16.12.10 from 00:0a:e4:4e:64:4e via 172.16.12.1
# Il fournit une adresse ip et la valide
DHCPACK on 172.16.12.10 to 00:0a:e4:4e:64:4e via 172.16.12.1
```

Réalisez une capture de trame du dialogue agent/serveur sur le routeur.

Analysez la capture

Entre le relais DHCP et le serveur DHCP a-t-on utilisé des adresses de diffusion MAC ?

Comment le serveur DHCP sait-il dans quelle plage d'adresse (étendue) il doit puiser l'adresse ?

Fixez un bail à 1 mn et étudiez le mécanisme de renouvellement automatique en examinant une capture de 3 minutes.

Modifiez la configuration du serveur afin de faire de la réservation d'adresse pour le client. Vérifiez le fonctionnement.

Une fois que tout fonctionne, activez tous les services en mode daemon et vérifiez le fonctionnement de la maquette.

Chapitre 26. Installation d'un serveur DNS

La résolution de noms − Fiche de cours

La résolution de noms − Fiche de cours

Ce document décrit la procédure d'installation et de configuration d'un serveur de noms sous GNU/Linux

26.1. Description et objectifs de la séquence

Avant d'installer un service quel qu'il soit, il faut s'assurer du bon fonctionnement de la résolution de noms sur le réseau. Pour cela vous avez le choix entre l'utilisation des fichiers hosts ou du service DNS. C'est ce dernier qui sera utilisé. Vous devez être familiarisé avec l'installation de GNU/Linux.

26.2. Qu'est ce que le service de résolution de noms de domaine

Le service de résolution de noms d'hôtes DNS (Domain Name Services), permet d'adresser un hôte par un nom, plutôt que de l'adresser par une adresse IP. Quelle est la structure d'un nom d'hôte ?

```
                Nom_d_hôte          ou bien          Nom_d_hôte.NomDomaine
Exemple :       ns1                 ou bien          ns1.foo.org
```

Le nom de domaine identifie une organisation dans l'internet, comme, par exemple, yahoo.com, wanadoo.fr, eu.org. Dans les exemples, nous utiliserons un domaine que l'on considère fictif : << foo.org >>. Chaque organisation dispose d'un ou plusieurs réseaux. Ces réseaux sont composés de noeuds, ces noeuds (postes, serveurs, routeurs, imprimantes) pouvant être adressés.

Par exemple, la commande **`ping ns1.foo.org`**, permet d'adresser la machine qui porte le nom d'hôte `ns1`, dans le domaine (organisation) `foo.org`.

Quelle différence entre la résolution de noms d'hôtes avec un serveur DNS et les fichiers `hosts` ?

Avec les fichiers `hosts`, chaque machine dispose de sa propre base de données de noms. Sur des réseaux importants, cette base de données dupliquée n'est pas simple à maintenir.

Avec un service de résolution de noms, la base de données est localisée sur un serveur. Un client qui désire adresser un hôte regarde dans son cache local, s'il en connaît l'adresse. S'il ne la connaît pas il va interroger le serveur de noms.

Tous les grands réseaux sous TCP/IP, et internet fonctionnent (schématiquement) sur ce principe.

Avec un serveur DNS, un administrateur n'a plus qu'une seule base de données à maintenir. Il suffit qu'il indique sur chaque hôte, quelle est l'adresse de ce serveur. Ici il y a 2 cas de figures possibles :

- soit les hôtes (clients) sont des clients DHCP (Dynamic Host Configuration Protocol), cette solution est particulière et n'est pas abordée ici. Cette technique est l'objet d'un autre chapitre.
- soit les clients disposent d'une adresse IP statique. La configuration des clients est détaillée dans ce document.

Normalement un service DNS nécessite au minimum deux serveurs afin d'assurer un minimum de redondance. Les bases de données des services sont synchronisées. La configuration d'un serveur de noms secondaire sera expliquée. Nous verrons également en TP le fonctionnement de la réplication des bases de données (bases d'enregistrements de ressources). On peut parler de bases de données réparties et synchronisées.

26.3. Présentation des concepts

26.3.1. Notion de domaine, de zone et de délégation

Un << domaine >> est un sous−arbre de l'espace de nommage. Par exemple `.com` est un domaine, il contient toute la partie hiérarchique inférieure de l'arbre sous jacente au noeud `.com`.

Un domaine peut être organisé en sous domaines. `.pirlouit.com` est un sous domaine du domaine `.com`. Un domaine peut être assimilé à une partie ou sous−partie de l'organisation de l'espace de nommage. Voir la diapositive sur les Domaines, zones et délégations.

Figure 26−1. Les domaines

Une "zone" est une organisation logique (ou pour être plus précis, une organisation administrative) des domaines. Le rôle d'une zone est principalement de simplifier l'administration des domaines. Le domaine ".com" peut être découpé en plusieurs zones,

z1.com, z2.com...zn.com. L'administration des zones sera déléguée afin de simplifier la gestion globale du domaine. Voir la diapositive sur les zones.

Figure 26−2. Les zones

La délégation consiste à déléguer l'administration d'une zone (ou une sous−zone) aux administrateurs de cette zone. Voir la diapositive sur la délégation.

Figure 26−3. La délégation

Attention à ces quelques remarques :

- Un domaine est une organisation de l'espace de nommage. Il peut être attaché à un domaine parent, et/ou peut avoir un ou plusieurs sous−domaines enfants.
- Les zones correspondent à des organisations administratives des domaines. Un domaine peut être administré par plusieurs zones administratives, mais il est possible aussi qu'une zone serve à l'administration de plusieurs domaines. Prenons l'exemple d'un domaine "MonEntreprise.fr", membre de ".fr". Il peut être composé de trois sous−domaines France.MonEntreprise.fr, Italie.MonEntreprise.fr, Espagne.MonEntreprise.fr et de deux zones d'administration. Une en France pour les sous−domaines France.MonEntreprise.fr, Italie.MonEntreprise.fr (il n'y a pas de délégation), et une pour Espagne.MonEntreprise.fr, il y a délégation.
- L'adressage IP correspond à une organisation physique des noeuds sur un réseau IP.
- L'organisation de l'espace de nommage est complètement indépendante de l'implantation géographique d'un réseau ou de son organisation physique. L'organisation physique est gérée par des routes (tables de routage). L'espace de nommage indique pour un nom de domaine N, quelles sont les serveurs de noms qui ont autorité sur cette zone. Elles ne donnent pas la façon d'arriver à ces machines.
- Les seules machines connues au niveau de l'espace de nommage, sont les serveurs de nom "déclarés". Ces informations sont accessibles par des bases de données "whois".
- La cohérence (le service de résolution de noms) entre l'organisation de l'espace de nommage global et les organisations internes des réseaux sur internet est réalisée par les serveurs de noms.

26.3.2. le domaine in−addr.arpa

Le principe de la résolution de noms, consiste à affecter un nom d'hôte une adresse IP. On parle de résolution de noms directe. Le processus inverse doit pouvoir également être mis en oeuvre. On parle de résolution de noms inverse ou reverse. Le processus doit fournir, pour une adresse IP, le nom correspondant. Pour cela il y a une zone particulière, in−addr.arpa, qui permet la résolution inverse d'adresse IP. Voir la diapositive sur la résolution inverse.

Figure 26−4. La résolution inverse

Par exemple, pour le réseau 192.68.1.0, on créera une zone inverse dans le domaine in−addr.arpa. La zone de recherche inverse dans le domaine deviendra : 1.68.192.in−addr.arpa. Cette zone devra répondre pour toutes les adresses déclarées dans la tranche 192.168.1.0 à 192.168.1.254.

On inscrira dans cette zone tous les noeuds du réseau pour lesquels on désire que la résolution inverse fonctionne. Un serveur de noms peut, pratiquement, fonctionner sans la définition de cette zone tant que le réseau n'est pas relié à l'internet. Si cela était le cas, il faudrait déclarer cette zone, sans quoi, des services comme la messagerie électronique, ne pourrait fonctionner correctement, notamment à causes des règles anti−spam. (Voir www.nic.fr)

26.3.3. Fichiers, structure et contenus

Sur linux nous allons utiliser deux types de fichiers :

- le fichier `/etc/bind/named.conf`, qui décrit la configuration générale du serveur DNS,
- les fichiers qui contiennent les enregistrements de ressources pour la zone dans `/etc/bind`. On crée, en général, un fichier pour la résolution directe d'une zone, et un fichier pour la résolution inverse.

Les enregistrements ont une structure et un rôle que nous verrons. Le daemon se nomme named, prononcer << naime dé >>.

26.3.4. Principaux types d'enregistrements

Les types d'enregistrements qui enrichissent une base de données DNS sont de plusieurs types, dont voici les principaux :

- Enregistrement de type SOA (*Start Of Authority*) : indique l'autorité sur la zone. Ces enregistrements contiennent toutes les informations sur le domaine. Par exemple le délai de mise à jour des bases de données entre serveurs de noms primaires et secondaires, le nom du responsable du site
- Enregistrements de type NS (*Name Server*) : ces enregistrements donnent les adresses des serveurs de noms pour le domaine.
- Enregistrement de type A (*Adresse*) : ces enregistrements permettent de définir les noeuds fixes du réseau (ceux qui ont des adresses IP statiques). Serveurs, routeurs, switchs ...
- Enregistrements de type MX (*Mail eXchanger*) : ils servent pour déclarer les serveurs de messagerie. Il faudra déclarer une enregistrement de type MX pour la réalisation du TP sur la messagerie.
- Enregistrements de type CNAME (*Canonical Name*) : ils permettent de définir des alias sur des noeuds existants. Par exemple www.foo.org peut être la même machine que web.foo.org. Dans ce cas, << www >> est un alias (CNAME) de << web >>. Cela permet de différencier le nommage des machines des standards de nommages des services (www, ftp, news, smtp, mail, pop...).
- Enregistrement de type PTR (Pointeur) : ils permettent la résolution de noms inverse dans le domaine in−addr.arpa.

Ces enregistrements caractérisent des informations de type IN − INternet. Voir l'annexe pour avoir un fichier exemple.

26.3.5. Structure des enregistrements

Structure d'un enregistrement SOA : chaque fichier de ressource de zone commence par un enregistrement de type SOA. Voici un exemple d'enregistrement SOA :

```
$TTL 38400
foo.org. IN SOA ns1.foo.org. hostmaster.foo.org. (
        20001210011     ; numéro de série
        10800           ; rafraîchissement
        3600            ; nouvel essai
        604800          ; Obsolescence après une semaine
        86400 )         ; TTL minimal de 1 jour
```

Caractéristiques des différentes informations :

SOA Start Of Authority, enregistrement qui contient les informations de synchronisation des différents serveurs de nom.

foo.org, donne le nom de la zone. Le nom de la zone, ici "foo.org", peut être remplacé par "l'@", arobase.

hostmaster.foo.org : la personne qui est responsable de la zone. Le premier point sera remplacé par l'arobase (@) pour envoyer un courrier électronique. Cela deviendra hostmaster@foo.org. En général postmaster, est un alias de messagerie électronique vers l'administrateur du DNS.

1. Numéro de série sous la forme AAAAMMJJNN, sert à identifier la dernière modification sur le serveur de noms maître. Ce numéro sera utilisé par les serveurs de nom secondaires pour synchroniser leurs bases. Si le numéro de série du serveur de noms primaire est supérieur à celui des serveurs de noms secondaire, alors le processus de synchronisation suppose que l'administrateur à apporté une modification sur le serveur maître et les bases sont synchronisées.
2. Rafraîchissement : Intervalle de temps donné en seconde pour indiquer au serveur la périodicité de la synchronisation.
3. Retry : intervalle de temps avant réitération si l'essai précédent n'a pas fonctionné.
4. Expire : temps au bout duquel le serveur ne remplit plus sa mission s'il n'a pu contacter le serveur maître pour mettre à jour ses données.
5. TTL : Time To Live, durée de vie des enregistrements. Plus la durée de vie est courte, plus l'administrateur est susceptible de considérer que ses bases sont à jour, par contre cela augmente le trafic sur le réseau.

Enregistrement de type NS pour le domaine foo.org :

```
foo.org.        IN NS   ns1.foo.org.    ; noter le point final "."
foo.org.        IN NS   ns2.foo.org.    ; foo.org peut être remplacé par "@"
                                        ; IN signifie enregistrement de type INternet
```

Le << . >> final signifie que le nom est pleinement qualifié. On aurait pu mettre :

```
@               IN NS   ns1
                IN NS   ns2
```

"@" signifie "foo.org" et pour le serveur de nom, comme "ns1" n'est pas pleinement qualifié, cela équivaut à "ns1.foo.org".

Enregistrements de type A : nous devons décrire la correspondance Nom / Adresse

```
ns1.foo.org.            IN      A       192.168.0.1
```

```
ns2.foo.org.            IN      A       192.168.0.2
localhost.foo.org.      IN      A       127.0.0.1
```

S'il y avait d'autres hôtes sur la zone, il faudrait les définir ici.

Enregistrements de type CNAME : Ce sont les alias (Canonical Name). Une requête du type http://www.foo.org sera adressée à ns1.foo.org, puisque www est un alias de ns1.

```
www                     IN      CNAME   ns1.foo.org.
ftp                     IN      CNAME   ns1.foo.org.
```

Enregistrement de type PTR : il serviront à la résolution de noms inverse.

```
1.0.168.192.in-addr.arpa.       IN      PTR     ns1.foo.org.
2.0.168.192.in-addr.arpa.       IN      PTR     ns2.foo.org.
```

26.3.6. La délégation

La délégation consiste à donner l'administration d'une partie du domaine à une autre organisation. Il y a transfert de responsabilité pour l'administration d'une zone. Les serveurs de la zone auront autorité sur la zone et auront en charge la responsabilité de la résolution de noms sur la zone. Les serveurs ayant autorité sur le domaine auront des pointeurs vers les serveurs de noms ayant autorité sur chaque zone du domaine.

26.3.7. Serveur primaire et serveur secondaire

Le serveur maître (primaire) dispose d'un fichier d'information sur la zone. Le ou les serveurs esclaves (secondaires) obtiennent les informations à partir d'un serveur primaire ou d'un autre serveur esclave. Il y a " transfert de zone". Les serveurs maîtres et esclaves ont autorité sur la zone.

26.3.8. Le cache

L'organisation d'internet est assez hiérarchique. Chaque domaine dispose de ses propres serveurs de noms. Les serveurs peuvent être sur le réseau physique dont ils assurent la résolution de nom ou sur un autre réseau. Chaque zone de niveau supérieur (edu, org, fr...) dispose également de serveurs de nom de niveau supérieur. L'installation du service DNS, installe une liste de serveurs de noms de niveaux supérieurs. Cette liste permet au serveur de résoudre les noms qui sont extérieurs à sa zone. Le serveur enrichit son cache avec tous les noms résolus. Si votre réseau réseau n'est pas relié à internet, vous n'avez pas besoin d'activer cette liste.

Ce fichier est un peu particulier. Il est fournit avec les distributions. Il est utilisé par le serveur de noms à l'initialisation de sa mémoire cache. Si vos serveurs sont raccordés à internet, vous pourrez utiliser une liste officielle des serveurs de la racine (ftp.rs.internic.net).

26.4. Installation et configuration d'un serveur DNS

Processus de configuration

L'application est déjà installée. Pour mettre en place le service de résolution de noms sur un serveur GNU/Linux, on va procéder successivement aux opérations suivantes :

1. vérifier les fichiers déjà installés,
2. configurer les fichiers des zones administrées,
3. configurer les fichiers de transaction sécurisée pour rndc,
4. démarrer et tester le service serveur.

26.4.1. Fichiers déjà installés

Vous devez normalement avoir déjà les fichiers suivants :

1. `/etc/bind/named.conf`, fichier de déclaration des fichiers de ressources
2. `/etc/bind/db.127`, zone locale reverse
3. `/etc/bind/db.0`, zone locale de broadcast
4. `/etc/bind/db.255`, zone locale de broadcast
5. `db.local`, zone directe locale
6. `db.root`, fichiers des serveurs racine

Le contenu de tous ces fichiers et commentaires se trouve en annexe.

Vous avez également des fichiers particuliers : `rndc.key`, `rndc.conf`. rndc, est un outil qui permet de passer des commandes à distance à un serveur de nom. Nous porterons une attention toute particulière à ces fichiers, à leur rôles et à l'utilité de rndc.

Il va suffire de rajouter les fichiers manquants à la zone administrée.

26.4.2. rndc, le fichier de configuration, le fichier de clé

rndc est un outil qui permet de réaliser des transactions sécurisées avec un serveur de nom. Le mode de fonctionnement est dit à "clé partagée", c'est à dire que le client rndc et le serveur bind doivent avoir la même clé. Vous devrez donc configurer le fichier de configuration de rndc et le fichier named.conf avec les mêmes paramètres.

Ces fichiers et exemples sont également fournis en annexe. La clé doit être strictement identique dans les 2 fichiers. Si vous avez un message d'erreur à l'utilisation de rndc, vérifiez bien ces paramètres.

rndc supporte plusieurs paramètres pour passer des commandes au serveur de nom (halt, querylog, refresh, reload, stat...). Utilisez la commande "man rndc" pour en savoir plus.

Dans le fichier rndc, vous allez avoir besoin d'au moins 3 paramètres. rndc utilisera ces paramètres si rien n'est spécifié sur la ligne de commande. Dans les autres cas, vous pouvez passer les paramètres sur la ligne de commande.

Note : vous pouvez vous passer du système de clé mais ce n'est pas conseillé. Commentez tout ce qu'il y a dans le fichier named.conf et qui concerne la clé s'il y a déjà des choses. Renommez le fichier rndc.conf en rndc.conf.orig, ça devrait fonctionner. Vous pouvez tester cela en faisant un **`/etc/init.d/bind restart`**. Vous ne devriez pas avoir de message d'erreur.

```
#Description du serveur et de la clé utilisés par défaut.
#Ici on utilise par défaut le serveur local, avec la clé key-name
   options {
     default-server  localhost;
     default-key     "<key-name>";
   };
```

Il est possible de dire quelle clé utiliser en fonction d'un serveur donné.

```
   server localhost {
     key  "<key-name>";
   };
```

Enfin il reste à définir la ou les clés avec leur noms et leurs valeurs.

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

Pour créer une nouvelle clé, utilisez la commande :

```
dnssec-keygen -a hmac-md5 -b <bit-length> -n HOST <key-file-name>

#Ici on génère une clé de 512 bits dans un fichier maCLE
dnssec-keygen -a hmac-md5 -b 512 -n HOST maCLE
```

Le fichier `named.conf` doit connaître la clé utilisée par le client,

```
// secret must be the same as in /etc/rndc.conf
key "key" {
        algorithm       hmac-md5;
        secret
"c3Ryb25nIGVub3VnaCBmb3IgYSBtYW4gYnV0IG1hZGUgZm9yIGEgd29tYW4K";
};
```

mais doit également comprendre les paramètres qui définissent les machines clientes autorisées à passer des commandes avec une directive `controls`.

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

# Ici on peut passer des commandes à partir de n'importe quelle machine
controls {
    inet 127.0.0.1 allow { any; } keys { "key"; };
};

# Ici on peut passer des commandes localement
controls {
    inet 127.0.0.1 allow { localhost; } keys { "key"; };
};
```

26.4.3. Procédure de configuration du serveur

L'installation a copié les fichiers. Sur une configuration simple vous allez avoir 3 fichiers à créer ou à modifier sur le serveur primaire :

- `/etc/bind/named.conf` (fichier de configuration globale du service DNS du serveur de noms primaire),
- `/etc/bind/db.foo.org` qui contiendra la description de la correspondance nom−adresse de toutes les machines du réseau
- `/etc/bind/db.foo.org.rev` qui contiendra la correspondance inverse adresse−nom (pour la résolution inverse de nom in−addr.arpa).

26.4.4. Configurer les fichiers

Vous pouvez configurer le serveur manuellement, c'est à dire créer les fichiers à l'aide d'un éditeur de texte ou à l'aide d'un outil de configuration graphique. En général on n'installe jamais d'interface graphique sur un serveur pour des questions de sécurité. Nous allons donc créer les fichiers complètement. La configuration est réalisable également à distance avec des requêtes HTTP grâce à des outils comme webmin.

26.4.5. Configuration du DNS manuellement

Le fichier racine pour la configuration du serveur de noms est le fichier `/etc/bind/named.conf`. Ce fichier est lu au démarrage du service et donne la liste des fichiers qui définissent la base de données pour la zone.

26.4.6. Le fichier `named.conf`

Voir annexe.

26.4.7. Le fichier db.foo.org

Voir annexe.

Le paramètre @, signifie qu'il s'agit du domaine "foo.org" (le nom tapé après le mot " zone " dans le fichier de configuration named.conf). Le paramètre "IN", signifie qu'il s'agit d'un enregistrement de type internet. Notez la présence d'un point (.) après le nom des machines pleinement qualifiés. Sans celui−ci, le nom serait " étendu ". Par exemple, ns1.foo.org (sans point) serait compris comme ns1.foo.org.foo.org (on rajoute le nom de domaine en l'absence du point terminal). Le point (.) terminal permet de signifier que le nom est pleinement qualifié.

26.4.8. Le fichier db.foo.org.rev

Voir annexe.

26.5. Compléments pratiques

26.5.1. Démarrer ou arrêter le service le service

Le service (daemon) qui active la résolution de noms s'appelle named prononcer << naime dé >>, mais le script s'appelle bind, ou sur certaines distributions bind9. Je noterai ici bind.

Si vous voulez l'arrêter ou le redémarrer dynamiquement vous pouvez utiliser les commandes suivantes :

```
# La commande stop utilise souvent rndc.
# rndc doit donc être préalablement configuré.
/etc/init.d/bind stop
/etc/init.d/bind start
```

Relancer le service serveur de cette façon peut parfois poser problème. En effet cette procédure régénère le cache du serveur. Le service prend également un nouveau PID. Si vous voulez éviter cela, ce qui est généralement le cas, préférez la commande **`kill −HUP` *`PID de Named`***. Vous trouverez le PID de named dans `/var/run`.

26.5.2. Finaliser la configuration

Les fichiers de configuration sont créés. Il ne reste plus qu'à tester. Il faut au préalable configurer le serveur pour que tous les processus clients utilisent le service de résolution de nom. Il vous faut modifier le fichier `/etc/resolv.conf` :

```
# nameserver AdresseIpDuServeurDeNom
# Exemple
nameserver 192.168.0.1
```

Vous pouvez également configurer d'autres clients pour qu'ils utilisent votre serveur de nom.

26.5.3. Procédure de configuration des clients

La description de la configuration de tous les clients possibles n'est pas détaillée. Vous trouverez ci−dessous des éléments pour un client windows 9x et pour un client GNU/Linux.

26.5.4. Avec windows

Il s'agit d'un client windows. Chaque client dispose du protocole TCP/IP, d'une adresse IP. Il faut configurer le client pour lui signifier quel est le serveur de noms qu'il doit consulter. Pour cela il faut aller dans : panneau de configuration − réseau − tcp/ip − Onglet << Configuration DNS >>. Vous allez pouvoir définir les paramètres suivants :

- le nom d'hôte de la machine locale dans le réseau
- le nom de domaine auquel appartient l'hôte (dans cet exemple c'est foo.org)

Ces 2 paramètres sont facultatifs dans l'atelier qui nous intéresse. Par contre le paramètre << Ordre de recherche DNS >> est important. Mettez dessous :

- L'adresse IP du serveur de noms que vous avez configuré,
- Cliquez sur ajouter
- Entrez l'adresse IP du serveur de noms
- Validez puis relancer la machine

Ce paramètre, définit à la machine locale, l'adresse de l'hôte de destination qui est chargé de la résolution des noms d'hôtes dans le réseau. Cela permet de dire qu'un serveur de noms doit avoir une adresse IP statique sur le réseau.

26.5.5. Avec GNU/Linux

Vous pouvez modifier (en tant que root) le fichier de configuration du << resolver >> (`/etc/resolv.conf`). Exemple (ça tient en deux lignes) :

```
# Fichier /etc/resolv.conf
search foo.org
nameserver 192.168.1.1 # mettre votre DNS
```

26.6. Procédure de tests

Attention aux fichier `hosts` et au fichier `host.conf`. Prenez le temps de regarder ce qu'il y a dedans. Faites une copie de sauvegarde de ces fichiers et renommez les. Vérifiez au besoin leur utilité avec les commandes **`man host.conf`** et **`man hosts`**.

Vous pouvez tester votre configuration avant même d'avoir configuré un client. Sur la même machine vous allez utiliser un service client du serveur (commande **ping**) qui utilisera un service serveur (DNS).

Test sur le serveur de noms : Tapez la commande **`ping ftp.foo.org`**. Si la commande répond, le serveur fonctionne. En effet ftp est un alias de ns1 dans la zone foo.org.

Test sur le client : Avant de lancer une commande, vous devez vérifier que vous n'avez pas de fichier `hosts` local, sinon vous devez le supprimer.

Pourquoi ? L'utilisation de fichiers hosts et d'un serveur de noms n'est pas exclusif. Dans bien des environnements, le fichier `hosts` est consulté avant le serveur de noms (notamment windows, GNU/Linux à moins que ce ne soit précisé). Si vous avez un fichier `hosts` sur la machine, vous pouvez avoir des résultats qui ne sont pas ceux attendus.

26.6.1. Vérifier la résolution de noms :

Pensez à bien vérifier le nom d'hôte de votre machine avec la commande **hostname**, au besoin, sous root, modifiez ce nom, toujours avec cette commande. Fermez les sessions et rouvrez les, vous aurez le bon nom d'hôte qui s'affichera sur votre console.

Mettons que le réseau soit configuré de la façon suivante :

```
Nom d'hôte      Alias (CNAME)           Adresse IP Serveur
ns1             www
                ftp
                mail
                ns1                     192.68.1.1
Client 1        Cli1                    192.68.1.2
```

Pour vérifier le fonctionnement de la résolution de noms à partir du client cli1, vous pouvez utiliser les commandes suivantes :

```
− ping ns1
```

```
− ping cli1
```

Vous pouvez également tester la résolution des alias (CNAME) avec les commandes :

```
ping            mail.foo.org
ping            www.foo.org
ping            ftp.foo.org
ping            ns1.foo.org
```

C'est bien la même adresse IP (voir le cache arp) qui répond, la machine a donc bien plusieurs noms.

Si vous voulez vérifier que c'est bien le serveur de noms qui réalise la résolution, il existe plusieurs solutions. La plus simple est d'arrêter le service serveur avec la commande **`/etc/init.d/bind stop`**, puis de refaire les manipulations. Aucune machine n'est atteignable en utilisant son nom, mais cela est toujours possible en utilisant l'adresse IP.

26.7. Dépannage et outils

Les sources de dysfonctionnement des services de nom peuvent être nombreuses et parfois complexes à résoudre. Voici quelques outils et méthodes qui peuvent être utilisées.

26.7.1. Les erreurs de chargement de bind

Si vous avez une erreur similaire à celle−ci :

```
Problème de clés entre named et rndc
root@knoppix:/etc/bind# /etc/init.d/bind9 stop
Stopping domain name service: namedrndc: connection to remote host closed
This may indicate that the remote server is using an older version of
the command protocol, this host is not authorized to connect,
or the key is invalid.
```

Le problème est lié à rndc, et souvent à des clés qui sont différentes ou mal définies entre `named.conf` et `rndc.conf`. Vérifiez donc bien tous les paramètres.

Vérifiez dans les journaux (en général `/var/log/daemon`) qu'il n'y a pas d'erreur de chargement de named. Voici un exemple de log.

```
# Log après chargement des zones
Apr  8 23:12:46 knoppix named[1066]: starting BIND 9.2.1
Apr  8 23:12:46 knoppix named[1066]: using 1 CPU
Apr  8 23:12:46 knoppix  []

named[1068]: loading configuration from '/etc/bind/named.conf'
named[1068]: no IPv6 interfaces found
named[1068]: listening on IPv4 interface lo, 127.0.0.1#53
named[1068]: listening on IPv4 interface eth0, 192.168.90.100#53
named[1068]: command channel listening on 127.0.0.1#953
named[1068]: zone 0.in−addr.arpa/IN: loaded serial 1
named[1068]: zone 127.in−addr.arpa/IN: loaded serial 1
named[1068]: zone 255.in−addr.arpa/IN: loaded serial 1
named[1068]: zone localhost/IN: loaded serial 1
named[1068]: zone foo.org/IN: loaded serial 2003040101
Apr  8 23:12:46 knoppix named[1068]: running
```

Ou encore avec la commande ps :

```
root:# ps aux | grep named
root  1066  0.0  1.6 10312 2136 ?  S    23:12   0:00 /usr/sbin/named
root  1067  0.0  1.6 10312 2136 ?  S    23:12   0:00 /usr/sbin/named
root  1068  0.0  1.6 10312 2136 ?  S    23:12   0:00 /usr/sbin/named
root  1069  0.0  1.6 10312 2136 ?  S    23:12   0:00 /usr/sbin/named
root  1070  0.0  1.6 10312 2136 ?  S    23:12   0:00 /usr/sbin/named
```

Vous pouvez également faire des tests successifs pour tester la résolution de nom.

```
#Vérification avec des ping

root@ns1:~# ping −c1 ns1.foo.org
PING ns1.foo.org (192.168.90.100): 56 data bytes
64 bytes from 192.168.90.100: icmp_seq=0 ttl=64 time=0.1 ms
−−− ns1.foo.org ping statistics −−−
1 packets transmitted, 1 packets received, 0% packet loss
round−trip min/avg/max = 0.1/0.1/0.1 ms



root@ns1:~# ping −c1 www.foo.org
PING ns1.foo.org (192.168.90.100): 56 data bytes
64 bytes from 192.168.90.100: icmp_seq=0 ttl=64 time=0.1 ms
−−− ns1.foo.org ping statistics −−−
1 packets transmitted, 1 packets received, 0% packet loss
```

```
round-trip min/avg/max = 0.1/0.1/0.1 ms
```

26.7.2. nslookup, dig

La commande **nslookup** est de moins en moins utilisée, nous la verrons donc pas. Nous allons voir l'utilisation de dig.

Ces commandes sont très largement utilisées par les administrateurs de réseau pour résoudre les problèmes liés aux services de résolution de nom.

Tests avec dig :

```
# Test sur une zone
root@knoppix:/etc/bind# dig any foo.org
; <<>> DiG 9.2.1 <<>> any foo.org
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 32752
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 1

;; QUESTION SECTION:
;foo.org.                       IN      ANY

;; ANSWER SECTION:

foo.org.                604800  IN      SOA     ns1.foo.org.   \
        root.ns1.foo.org. 2003040102 604800 86400 2419200 604800
foo.org.                604800  IN      NS      ns1.foo.org.

;; ADDITIONAL SECTION:
ns1.foo.org.            604800  IN      A       192.168.90.100

;; Query time: 7 msec
;; SERVER: 192.168.90.100#53(192.168.90.100)
;; WHEN: Tue Apr  8 23:30:05 2003
;; MSG SIZE  rcvd: 100



# Récupération de l'enregistrement SOA d'une zone

root@knoppix:/etc/bind# dig soa foo.org

; <<>> DiG 9.2.1 <<>> soa foo.org
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 15982
;; flags: qr aa rd ra; QUERY:1, ANSWER:1, AUTHORITY:1, ADDITIONAL:1

;; QUESTION SECTION:
;foo.org.                       IN      SOA

;; ANSWER SECTION:

foo.org.                604800  IN      SOA     ns1.foo.org.     \
          root.ns1.foo.org. 2003040102 604800 86400 2419200 604800

;; AUTHORITY SECTION:
foo.org.                604800  IN      NS      ns1.foo.org.

;; ADDITIONAL SECTION:
ns1.foo.org.            604800  IN      A       192.168.90.100

;; Query time: 2 msec
;; SERVER: 192.168.90.100#53(192.168.90.100)
;; WHEN: Tue Apr  8 23:30:43 2003
;; MSG SIZE  rcvd: 100



#Vérification de la résolution de nom sur www.foo.org

root@knoppix:/etc/bind# dig www.foo.org

; <<>> DiG 9.2.1 <<>> www.foo.org
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 52961
;; flags: qr aa rd ra; QUERY:1, ANSWER:2, AUTHORITY:1, ADDITIONAL:0

;; QUESTION SECTION:
;www.foo.org.                   IN      A

;; ANSWER SECTION:
www.foo.org.            604800  IN      CNAME   ns1.foo.org.
ns1.foo.org.            604800  IN      A       192.168.90.100
```

```
;; AUTHORITY SECTION:
foo.org.                604800  IN      NS      ns1.foo.org.

;; Query time: 3 msec
;; SERVER: 192.168.90.100#53(192.168.90.100)
;; WHEN: Tue Apr  8 23:31:49 2003
;; MSG SIZE  rcvd: 77



# Vérification de la résolution de nom inverse.

root@ns1:/etc/bind# dig ptr 100.90.168.192.in-addr.arpa
; <<>> DiG 9.2.1 <<>> ptr 100.90.168.192.in-addr.arpa
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 30642
;; flags: qr aa rd ra; QUERY:1, ANSWER:1, AUTHORITY:1, ADDITIONAL:0
;; QUESTION SECTION:
;100.90.168.192.in-addr.arpa.   IN      PTR

;; ANSWER SECTION:
100.90.168.192.in-addr.arpa. \
                    604800 IN  PTR  ns1.90.168.192.in-addr.arpa.

;; AUTHORITY SECTION:
90.168.192.in-addr.arpa.     \
                    604800 IN      NS      ns1.90.168.192.in-addr.arpa.

;; Query time: 7 msec
;; SERVER: 192.168.90.100#53(192.168.90.100)
;; WHEN: Tue Apr  8 23:45:39 2003
;; MSG SIZE  rcvd: 77
```

26.7.3. Le cache du DNS

Le cache permet également de détecter certaines causes d'erreur. Le problème est qu'il est en mémoire. Pour le récupérer sous la forme d'un fichier utilisez la commande **`kill -INT PID de named`**. Vous récupérez un fichier `/var/named/named_dump.db` que vous pouvez exploiter.

26.7.4. Les journaux

Si vous êtes en phase de configuration, pensez (ce doit être un réflexe) à consulter les fichiers de journalisation, notamment `/var/log/messages`. Cette opération permet dans bien des cas de corriger des erreurs qui se trouvent dans les fichiers de configuration. Voici comment procéder :

- Arrêt du serveur
- Nettoyage du fichier `> /var/log/messages`
- Démarrage du serveur
- Consultation des logs : **`cat /var/log/daemon.log | more`**

Pour les fichiers logs, utilisez, si le fichier est trop gros la commande **tail** :

```
# tail -N NomFichier
# Affiche les N dernières lignes d'un fichier
# Par exemple, affiche les 250 dernières lignes d'un fichiers
# tail -n 250 /var/log/daemon.log | more
```

26.8. Remarques

Si vous désirez mettre en place la résolution de noms sur un réseau local, il n'y a pas grand chose de plus à réaliser. Il faut rajouter les enregistrements de type MX pour la messagerie, cette opération sera réalisée pendant la configuration du service de messagerie. Il faut également mettre en place un service de synchronisation des bases de données avec un serveur secondaire pour assurer le service d'un serveur de noms de backup.

Si vous désirez vous relier sur internet, le processus est plus complexe. Il faudra approfondir la description des enregistrements et la structure des fichiers.

Par convention, on considère que chaque domaine dispose d'au moins 1 serveur de noms primaire et un serveur de noms secondaire afin d'assurer une redondance en cas de panne d'un serveur. Les clients réseau seront configurés pour utiliser indifféremment le serveur de noms primaire ou les serveurs de nom secondaires. Il en résulte une duplication de la base de données du DNS primaire sur les serveurs secondaires. La base de données est rafraîchie en fonction des paramètres de l'enregistrement SOA. Ce procédé met en oeuvre un principe de base de données répartie. Vous trouverez quelques éléments dans les annexes qui suivent.

26.9. Annexes

26.9.1. Annexe 1 − extraits de fichiers de configuration

Les extraits ci−dessous d'une zone fictive `foo.org` peuvent servir d'exemple pour bâtir une zone.

Si on respecte les conventions utilisées sur internet, voici ce que l'on devrait avoir :

- le serveur ftp est accessible par l'adresse `ftp.foo.org`
- le serveur http par l'adresse `www.foo.org`
- le serveur de noms primaire par `ns1.foo.org`
- le serveur de messagerie `mail.foo.org`
- le serveur de news `news.foo.org`, etc.

ftp, www, mail sont des alias (*canonical* name ou CNAME) de la machine `ns1` dans le domaine `foo.org`

Nous aurons donc sur le serveur de noms 5 enregistrements dans la zone `foo.org` qui concernent la machine `ns1.foo.org` : un enregistrement de type A pour déclarer `ns1` quatre enregistrements de type CNAME pour la machine `ns1`.

Nous aurons également, dans la zone reverse in−addr.arpa, 1 enregistrement de type pointeurs (PTR) pour chaque enregistrement de type A dans la zone directe. Enfin, pour le serveur de messagerie, il faut également un enregistrement de type MX.

Tous les fichiers concernant la zone locale, et un fichier `named.conf` sont déjà installés sur votre machine.

```
; db.local
; Résolution directe pour la zone locale
; BIND data file for local loopback interface
;
$TTL    604800
@       IN      SOA     localhost. root.localhost. (
                              1         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache TTL
;
@       IN      NS      localhost.
@       IN      A       127.0.0.1



; db.127
; Résolution inverse pour l'adresse de loopback
; BIND reverse data file for local loopback interface
;
$TTL    604800
@       IN      SOA     localhost. root.localhost. (
                              1         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache TTL
;
@       IN      NS      localhost.
1.0.0   IN      PTR     localhost.



; db.0
; Résolution inverse pour la zone de broadcast
; BIND reverse data file for broadcast zone
;
$TTL    604800
@       IN      SOA     localhost. root.localhost. (
                              1         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache TTL
;
@       IN      NS      localhost.



; db.255
; Résolution inverse pour la zone de broadcast;
; BIND reverse data file for broadcast zone
```

```
;
$TTL    604800
@       IN      SOA     localhost. root.localhost. (
                              1         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache TTL
;
@       IN      NS      localhost.



; db.root
; fichier des serveurs de noms de l'internet
; vous pouvez le consulter sur votre disque.



; db.foo.org
; fichier directe pour la zone foo.org
; BIND data file for local loopback interface
;
$TTL    604800
@       IN      SOA     ns1 root.ns1 (
                     2003040102         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache TTL
;
@       IN      NS      ns1
ns1     IN      A       192.168.90.100 ; @ ip du serveur de nom
www     IN      CNAME   ns1
ftp     IN      CNAME   ns1



; db.foo.org.rev
; fichier de résolution inverse pour la zone foo
; BIND data file for local loopback interface
;
$TTL    604800
@       IN      SOA     ns1 root.ns1 (
                     2003040102         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache TTL
;
@       IN      NS      ns1
100     IN      PTR     ns1




; fichier named.conf du serveur primaire
// C'est le fichier principal de configuration des DNS
// C'est ici que sont réalisées,, pour chaque zone, les déclarations
// des fichiers de ressources.

options {
        directory "/var/cache/bind";
// Serveurs à prévenir pour les transferts de zone
        forwarders {0.0.0.0;};
};

// Ici les paramètres pour les clés rndc
// Les paramètres doivent être strictement identiques à celui
// du fichier rndc.key ou rndc.conf
// Si vous avez des messages d'erreur à l'utilisation
// de cette commande, vérifier le contenu des fichiers.


key "rndc-key" {
algorithm       hmac-md5;
secret "c3Ryb25nIGVub3VnaCBmb3IgYSBtYW4gYnV0IG1hZGUgZm9yIGEgd29tYW4K";
};

# Autoriations rndc sur la machine.
controls {
   inet  127.0.0.1 allow {localhost;} keys {"rndc-key";};
};

// Indication pour les serveurs racines
zone "." {
        type hint;
```

```
        file "/etc/bind/db.root";
};


// be authoritative for the localhost forward
// and reverse zones, and for
// broadcast zones as per RFC 1912
zone "localhost" {
        type master;
        file "/etc/bind/db.local";
};

zone "127.in-addr.arpa" {
        type master;
        file "/etc/bind/db.127";
};

zone "0.in-addr.arpa" {
        type master;
        file "/etc/bind/db.0";
};

zone "255.in-addr.arpa" {
        type master;
        file "/etc/bind/db.255";
};


//zone directe de foo
zone "foo.org" {
        type master;
        file "/etc/bind/db.foo.org";
};

//Zone reverse pour 192.168.90.
zone "90.168.192.in-addr.arpa" {
        type master;
        file "/etc/bind/db.foo.org.rev";
};




; fichier name.conf du serveur secondaire
; L'entête de bouge pas
; Tout ce qui concerne localhost non plus car chaque DNS
; est primaire pour les zones locales
; on ajoute la déclarations des autres zones, le fichier
; de stockage et l'adresse IP du serveur primaire pour
; pouvoir réaliser les transferts de zone.

; Déclaration de la zone foo.org

zone "foo.org" {
        type slave;
        file "/etc/bind/db.foo.org";
        masters {192.168.90.1;};
};

zone "90.168.192.in-addr.arpa" {
        type slave;
        file "/etc/bind/db.foo.org.rev";
        masters {192.168.90.1;};
};



; fichier rndc.conf
/* $Id: cours-dns.xml,v 1.9 2004/11/07 14:20:07 jmj Exp $ */
/*
 * Exemple de fichier de rndc.conf, pris pour les TP
 */

options {
        default-server  localhost;
        default-key     "rndc-key";
};

server localhost {
        key     "rndc-key";
};

key "rndc-key" {
algorithm       hmac-md5;
secret "c3Ryb25nIGVub3VnaCBmb3IgYSBtYW4gYnV0IG1hZGUgZm9yIGEgd29tYW4K";
};
```

```
; fichier rndc.key

key "rndc-key" {
        algorithm hmac-md5;
        secret "49khYQyHfO4AqYO9K7by6Q==";
};
```

26.9.2. Annexe 2 − Serveur primaire et serveur secondaire

Pour configurer le serveur secondaire, vous n'avez pas grand chose à faire. Copiez le fichier `named.conf` du primaire sur le secondaire. Voyez l'exemple ci−dessus. Le dns secondaire téléchargera (processus de transfert de zone) les fichiers de ressources du dns primaire. Attention, le dns secondaire pour une zone est toujours dns primaire pour la zone locale `localhost`.

On remplace la définition `masters` par `slave` sauf pour la zone locale et les fichiers `db.local` et `db.127` qui sont lus localement. Ensuite vous avez rajouté l'adresse du serveur à partir duquel le transfert de zone doit s'effectuer.

Activer les serveurs de noms et analyser les traces (log) sur les 2 serveurs. Corrigez toutes les erreurs jusqu'à ce que cela fonctionne. Vous devriez obtenir la trace selon laquelle il y a eu un transfert de zone entre le serveur maître et le serveur esclave. Exemple :

```
Apr  6 plibre named[8821]: send AXFR query 0 to 195.115.88.38
```

*Expérience 1 :*Vous pouvez expérimenter un échange entre un serveur de noms primaire et un serveur esclave. Modifiez sur le serveur primaire le N° de série comme si vous aviez modifié les fichiers de ressources de `ns1` et relancez le service. Vérifiez le transfert de zone a mis à jour la base de données répartie.

Expérience 2 : Vous pouvez expérimenter une autre procédure d'échange, mais cette fois sans relancer le serveur de noms secondaire. Modifiez d'abord sur les deux serveurs le délai de rafraîchissement et mettez le à 2 ou 3 minute. Relancez les services. Modifiez sur le serveur primaire le N° de série dans l'enregistrement SOA, comme si vous aviez modifié les fichiers de ressources de ns1 et relancez le service. Si vous attendez, vous verrez la synchronisation s'opérer (trace dans les fichiers de logs). Vous découvrez ainsi le mode de fonctionnement de synchronisation des serveurs de noms sur internet.

Remarque : si vous voulez, sur ces serveurs assurer la gestion de plusieurs domaines, il suffit de rajouter les définitions de ressources pour ces domaines, puis de déclarer ces zones dans `/etc/named.conf`.

Notez également que la notion d'autorité est différente de la notion de serveur maître ou serveur esclave. En effet si vous avez en charge la gestion de deux zones (Z1 et Z2), vous pouvez mettre deux serveurs ayant autorité sur ces zones (ns1 et ns2), par contre ns1 peut être serveur maître pour Z1 et secondaire pour Z2, et ns2 peut être serveur maître pour Z2 et esclave pour Z1.

26.9.3. Annexe 3 − Mise en place d'une délégation de zone

Prenons l'exemple d'une zone `sd` d'adresse 192.168.254.0, rattachée à `foo.org`. Nous allons mettre en place une délégation de zone pour `sd`. La résolution des noms de la zone `sd.foo.org` est prise en charge par les serveurs de noms de la zone `sd`, nous n'avons donc pas à nous en occuper. Par contre nous devons déclarer ces serveurs afin de maintenir la cohérence de la hiérarchie.

Configuration de la délégation : sur le serveur de noms de la zone `foo.org` il faut rajouter les enregistrements qui décrivent les serveurs de noms de la zone `sd.foo.org` dans le fichier de zone `db.foo.org`.

```
sd              86400   NS      ns1.sd.foo.org.
                86400   NS      ns2.sd.foo.org.
```

Et les enregistrements qui déterminent les adresses de ces serveurs de noms.

```
ns1.sd.foo.org. IN      A       192.168.254.1
ns2.sd.foo.org. IN      A       192.168.254.2
```

La délégation de la zone in−addr.arpa : Dans la pratique, cette délégation est différente car la zone inverse ne dépend pas de la zone supérieure, mais d'une autre entité (in−addr). Le processus est donc un peu différent.

Pourquoi ? parce que cette zone reverse est gérée par l'entité qui gère l'espace 192.168.0 à 192.168.255 et il est fort probable que ce n'est pas la zone `foo` qui assure la résolution inverse pour tous les réseaux compris entre 192.168.0 et 192.168.255.

Ceci dit, cela n'empêche pas de réaliser cela sur une maquette. Il est possible de mettre en place cette résolution inverse. Nous allons donc considérer que la zone `foo.org` assure la résolution de noms inverse du réseau 192.168.254. Ce reviendrait à considérer que dans la réalité, la zone `sd` serait un sous domaine de `foo`. La configuration ici est simple, les masques de sous−réseaux utilisés ici sont ceux par défaut des classes (255.255.255.0) pour la classe C. Le principe pour la zone inverse est identique à celui de la zone directe. Il suffit de rajouter dans le fichier `db.0.168.192` les enregistrements suivants :

```
sd.foo.org.                         IN NS    ns1.sd.foo.org.
                                    IN NS    ns2.sd.foo.org.
1.0.168.192.in-addr.arpa   86400  IN  PTR  ns1.sd.foo.org.
2.0.168.192.in-addr.arpa   86400  IN  PTR  ns2.sd.foo.org.
```

26.9.4. Annexe 3 − Outils de diagnostique et contrôle

http://www.dnsreport.com/tools/dnsreport.ch Le plus complet pour les tests. Il explique assez bien les problème et les modifications à faire pour les résoudres. Avec le serveur eclis ont peut arriver à n'avoir que 2 warnings: "Multiple MX records" et "Mail server host name in greeting" qui corresponds respectivement au fait que l'on a pas de serveur de mail de secour et que l'on fait du virtual hosting. Note: ce test utilise un des root name serveur au hazard. Il faut faire le test plusieur fois pour avoir un apperçu complet de la situation.

http://www.zonecut.net/dns/ Affiche la situation vu depuis un root name serveur pris au hazard et ce avec une jolie représentation grphique. Là aussi il faut tester plusieurs fois pour avoir un aperçu complet. Les quelques lignes de résumé à la fin sont intéressantes.

http://www.squish.net/dnscheck/dnscheck.cgi Le seul qui fait un test exaustif de tout les root name serveur et qui fait un résumé pertinant à la fin. Il est un peut lent mais ça va finalement plus vite que de tester 10 fois les autres outils pour passer en revue tout les root name serveurs... Ca démontre bien, avec des probabilités en poucent, pourquoi on obtient pas toujours les mêmes résultats quand il y a un problème avec les DNS.

Chapitre 27. Travaux dirigés : installation du service DNS

27.1. Présentation − le contexte

Vous utilisez deux machines M1 et M2.

M1 sera serveur primaire de votre zone, il est également serveur HTTP, serveur FTP, serveur de messagerie et serveur de news.

M2 sera client de M1 pour les trois premières parties du TP et serveur secondaire pour la quatrième partie.

Vous prendrez l'adresse de réseau 192.168.x.0. `x` est variable pour chacun des binômes du groupe. La valeur sera donnée par votre enseignant. Vous remplacerez `x` par la valeur fournie tout au long de ce document (TD et TP).

Votre domaine est `couleur` ou `couleur` est une variable que vous donnera votre enseignant. Couleur prendra une des valeurs `rouge`, `vert`, `bleu`...

On considère que M1 est serveur web, serveur ftp, serveur de messagerie et serveur de news.

Voici les noms qui sont assignés :

- Serveur de noms primaire : ns1
- Serveur HTTP : www
- Serveur ftp : ftp
- Serveur de noms secondaire : ns2
- Serveur de mail : mail
- Serveur de news : news

Rédigez les éléments des fichiers `named.conf` des serveurs primaires et secondaires de votre zone. Vous rédigerez également les fichiers de ressources de la zone `couleur.org`.

Vous utiliserez les documents fournis dans la fiche de cours et en annexe.

Chapitre 28. Travaux pratiques : installation du service DNS

28.1. Présentation

Vous utilisez deux machines M1 et M2. Le TP comporte quatre parties.

1. Première partie : préparation de votre environnement réseau client et serveur
2. Deuxième partie : configuration de la résolution de noms pour la zone directe :

   M1 sera serveur de noms

   M2 sera client de M1

   Test de la configuration à l'aide des commandes ping, et de requêtes ftp, http
3. Troisième partie : configuration de la résolution de noms pour la zone reverse

   Test de la configuration à l'aide de dig

4. Quatrième partie : mise en place du serveur secondaire, modification de l'enregistrement SOA du serveur primaire
5. Test du transfert de zone

Vous utiliserez les documents réalisés en TD.

28.2. Préparation de votre environnement réseau client et serveur

Ouvrez une session et passez administrateur

Renommez sur les deux machines les fichiers `/etc/hosts` (**`mv /etc/hosts /etc/hosts.original`**) afin d'éviter les effets de bords sur la résolution de noms.

28.3. Installation du serveur de noms primaire

Procédure de configuration du serveur

Vérifiez que vous avez bien les fichiers de configuration de la zone locale, sinon vous devez commencer par là. Vous compléterez ensuite la configuration pour votre zone. Faites une copie de sauvegarde de ces fichiers.

- `/etc/bind/named.conf` (fichier de configuration du serveur de noms primaire),
- `/etc/bind/db.couleur.org` qui contiendra la description de la correspondance nom−adresse de toutes les machines de votre zone.
- `/var/db.couleur.org.rev` qui contiendra la correspondance inverse adresse−nom (pour la résolution inverse de nom in−addr.arpa).

Vérifiez et validez la configuration des clés de `rndc.conf` et `named.conf`.

28.3.1. Configuration du service serveur DNS manuellement

Faites la configuration du serveur (fichiers `named.conf`, `ressources`, `rndc`).

Démarrer le serveur.

Vérifier le bon fonctionnement (traces dans les journaux, processus) de named et rndc.

Corrigez tant qu'il y a des erreurs.

28.3.2. Configuration du service client manuellement

- Les services clients de M1 et M2 doivent être configurés pour utiliser le service de résolution de noms.
- Modifiez sur les deux machines le fichier `/etc/resolv.conf`.
- Relancez le service réseau.
- Testez la configuration
- Vérifiez que la résolution de noms fonctionne sur :

  www.couleur.org

  ftp.couleur.org

  mail.couleur.org ....
- Corrigez tant que cela ne fonctionne pas.
- Vérifiez à 'aide la commande **ping**, de requêtes FTP ou HTTP à partir d'un client, que le serveur de noms retourne bien les enregistrements.

  Vérifiez à l'aide la commande **dig**, que le serveur répond bien sur différents types de requêtes (**dig `any`**, **dig `www`**, **dig `soa`**).

28.4. Configuration de la zone reverse

Configurez à l'aide des fichiers fournis en annexe, la zone inverse (reverse). Ceci consiste à rajouter une déclaration dans le fichier `named.conf`. Créez le fichier de ressource correspondant.

Relancez le service named, vérifiez les journaux, corrigez les éventuelles erreurs.

Vérifiez à l'aide de **dig** que les requêtes de type **dig `ptr`** fonctionnent.

28.5. Installation du serveur de noms secondaire

Sur M2 vérifiez que vous avez bine les fichiers de déclaration pour la zone locale. Ajoutez et déclarez les zones directe et inverses pour votre zone.

Activez le serveur secondaire, vérifiez que le service est actif et vérifiez également dans les journaux qu'il n'y a pas d'erreurs.

Vous devez avoir dans `/var/log/daemon`, une trace qui confirme le transfert de zone.

N'allez pas plus loin tant que cela n'est pas en parfait état de fonctionnement.

28.5.1. Procédure de test du serveur secondaire

Arrêtez sur le serveur primaire le service named.

Configurez un client pour qu'il puisse utiliser aussi bien le serveur primaire que le serveur secondaire. Ajouter pour cela un enregistrement de déclaration du serveur secondaire sur le client.

Testez le fonctionnement du serveur secondaire, à partir d'un client, en utilisant des requêtes sur `www.couleur.org` ou `ftp.couleur.org`.

C'est le serveur secondaire qui doit répondre, le serveur primaire étant inactif.

Vérifier cela avec la commande **dig**.

28.6. Test de l'enregistrement SOA

Modifiez au minimum le temps de rafraîchissement des enregistrements du serveur Primaire. (2 ou 3 mn). Modifiez également le N° de série. Relancez le serveur primaire et vérifiez dans les logs que le transfert de zone s'effectue bien.

Faites une modification sur votre fichier de ressources `db.couleur.org` et modifiez le N° de série. Attendez quelques minutes, vous devriez trouver une trace de synchronisation des bases de données des serveurs, sans avoir eu besoin de relancer aucun serveur.

Chapitre 29. Installation d'un serveur NFS

Le partage de fichiers pour les clients Unix

Le partage de fichiers pour les clients Unix

29.1. Résumé

Pourquoi un service NFS alors que celui−ci est très peu utilisé sur les environnements Windows et qu'il n'existe à ma connaissance pas de produits libres client ou serveur pour Windows. Pour deux raisons :

La première est que le service NFS est très largement employé dans les environnement Unix/Linux. Si vous avez des machines sous Linux vous utiliserez NFS. Il est donc nécessaire de connaître les procédures de configuration et d'utilisation de ce service.

La deuxième concerne Windows. Vous aurez sans doute un jour envie ou besoin d'installer le produit Windows Services For Unix (WSFU) de Microsoft. Ce produit disponible déjà sous Windows NT4 Server et mis à jour pour Windows 2000, offre de nombreux outils d'administration de type Unix pour Windows, dont un service NFS.

Nous allons voir, dans un environnement Linux, comment utiliser le service NFS.

29.2. Installation des produits clients et serveurs

Sur une debian, et donc sur la Freeduc−Sup, vous pouvez installer les outils nécessaires sur un serveur grâce à **apt−get**.

Sur le serveur, il faut installer portmap, nfs−common, et nfs−kernel−server.

```
apt-get install portmap nfs-common nfs-kernel-server
```

Sur le client, il est faut installer nfs−common et portmat.

```
apt-get install portmap nfs-common
```

En ce qui concerne les sécurités, sachez que NFS utilise le wrapper tcp (tcpd). Il est possible de configurer la sécurité via les fichiers `/etc/hosts.allow` et `/etc/hosts.deny`. Les protocoles à ouvrir sur le serveur sont statd, nfsd, lockd, rquotad et mountd. Sur le client, il faut permettre à statd d'accèder à localhost.

Vous pouvez activer NFS par la commande **`/etc/init.d/nfs-kernel-server start`**. Il vous faudra au préalable avoir défini les ressources à partager (exporter).

Les programmes sur lequel s'appuie le service NFS utilisent les RPC (Remote Procedure Call). Ils s'inscrivent donc auprès du service portmap qui met à jour sa table de service rpc. Voici un extrait de ce que donne la commande **`rpcinfo -p`**

```
 program vers proto   port
    100000    2   tcp    111  portmapper
    100000    2   udp    111  portmapper
    100003    2   udp   2049  nfs
    100003    3   udp   2049  nfs
    100003    2   tcp   2049  nfs
    100003    3   tcp   2049  nfs
    100021    1   udp  33065  nlockmgr
    100021    3   udp  33065  nlockmgr
    100021    4   udp  33065  nlockmgr
    100021    1   tcp  38399  nlockmgr
    100021    3   tcp  38399  nlockmgr
    100021    4   tcp  38399  nlockmgr
    100005    1   udp    967  mountd
    100005    1   tcp    970  mountd
    100005    2   udp    967  mountd
    100005    2   tcp    970  mountd
    100005    3   udp    967  mountd
    100005    3   tcp    970  mountd
```

Voici maintenant les processus qui doivent être actifs sur le serveur NFS.

portmap gére le catalogue des programmes RPC,

mountd est chargé des opérations de montage/démontage d'arborescence,

nfsd exécute les primitives d'accès aux fichiers − requêtes émanant des clients.

29.2.1. Les fichiers de configuration du serveur NFS

`/etc/exports` décrit ce que le serveur exporte, vers quelles machines le serveur exporte, avec quelles autorisations. Il s'agit d'un fichier texte, qui est éditable avec n'importe quel éditeur. Il centralise la liste de l'ensemble des ressources offertes par cette machine. Notez cependant que le nom de partage est automatiquement celui de la ressource (on ne peut pas partager sous un autre nom), et qu'enfin, on peut ponctuellement partager une ressource sans passer par ce fichier (à la volée : voir **exportfs**).

Exemple de fichier `/etc/exports` :

```
# Ressource  Options   Liste_de_Clients
# Exporte /tmp vers la machine "cli" avec possibilité Read Write (rw)
# rw est l'option par défaut
/tmp  cli(rw)
#Exporte "/tmp" en lecture seule vers toutes les machines du réseau
/tmp  *(ro)
```

29.2.2. Les fichiers de configuration du client NFS

Il n'y a pas de fichier particulier. Il suffit que les programmes soient installés (portmap et nfs−common). Pensez à lancer le **portmap**, sinon le montage restera en attente.

```
/etc/init.d/portmap start
```

Les répertoires exportés par un serveur peuvent être << montés >> manuellement ou à la demande. Nous verrons comment configurer un fichier sur le poste client, afin qu'un dossier soit << monté >> automatiquement au démarrage du client. Il s'agit dans ce cas de l'utilisation permanente de la ressource.

29.2.3. Exemple Unix de montage NFS

Prenons la configuration précédente (fichier `/etc/exports` ci−dessus)

Le client `cli1` monte (importe) `/tmp` de `ns1` sur le répertoire local `/tempo` en utilisant la commande suivante

`$mount -t nfs` ***`ns1:/tmp /tempo`***. `-t` indique le type de SGF − arborescence NFS −

Une fois montée, l'accès à la ressource est transparent.

En fin d'utilisation, le client démonte l'arborescence `/tmp` en utilisant la commande **`$umount`** ***`/tempo`***

A chaque opération de montage ou démontage, le fichier local `/etc/mtab` est mis à jour. Il contient la liste des systèmes de fichiers montés (arborescence NFS ou non).

Attention : NFS utilise un cache. Si vous ne voulez pas perdre de données, utiliser une procédure de << démontage >> des disques ou alors un << shutdown >> du poste client. Dans les autres cas, vous risquez de perdre les informations logées en cache.

29.2.4. Configuration du serveur

Vérifiez que le noyau supporte le système de fichiers nfs:

Utilisez la commande **more *`/proc/filesystems`***, voici ce que vous pouvez obtenir.

```
nodev   pipefs
        ext2
nodev   ramfs
        msdos
        vfat
        iso9660
nodev   usbfs
nodev   nfs
```

Le système de fichiers nfs doit apparaître. S'il n'apparaît pas, c'est que le système n'est pas compilé avec le support de NFS, ou alors il est compilé pour le charger comme un module. Si c'est le cas, vous pouvez charger le module avec la commande :

```
insmod nfs
```

ou encore

```
modprobe nfs
```

Le module doit apparaître avec la commande **lsmod**, et le fichier `/proc/filesystems` est normalemnt modifié (insmod force l'insertion d'un module dans le noyau, un module pouvant étre un pilote de périphérique par exemple, ou des fonctionnalités de la machine. Modprobe insert le module avec autodétection, afin de vous rendre compte de son bon fonctionnement, ou en donnant les bons paramètres. Lsmod liste les modules présents dans votre noyau, et rmmod permet la suppression d'un module présent dans le noyau).

29.2.4.1. Le fichier `/etc/exports`

Ce fichier est utilisé par les daemons pour déterminer les volumes qui seront exportés (accessibles), et quels seront les permissions à accorder sur ces volumes. Il existe autant de lignes que de points de montage. La structure d'une ligne est de la forme :

```
PointDeMontage  client1(option) clientn(option)
```

− PointDeMontage est le volume local à exporter,

− Client1 ... Clientn définissent les ordinateurs qui ont le droit d'accéder au volume exporté,

− Option: définit le type d'accès et les permissions.

Exemple de fichier avec la commande **more *`/etc/exports`***

```
/tmp   *.archinet.edu(rw)
/usr/local/man *.archinet.edu(ro)
```

Le dossier `/tmp` est exporté en lecture et écriture pour tous les ordinateurs du domaine archinet.edu. Le dossier `/usr/local/man` en lecture uniquement.

Voici quelques options de montage, utiliser **man *`exports`*** pour avoir la liste exhaustive :

Secure : requiert une authentification

Insecure : ne requiert pas d'authentification

ro | rw : lecture uniquement ou lecture écriture

Noaccess : permet d'exclure une partie de l'arborescence pour des clients donnés

Voici un extrait de la page de manuel :

Exemple

```
# fichier /etc/exports d'exemple
/               maître(rw) confiance(rw,no_root_squash)
/projects       proj*.local.domain(rw)
/usr            *.local.domain(ro) @trusted(rw)
/home/joe       pc001(rw,all_squash,anonuid=150,anongid=100)
/pub            (ro,insecure,all_squash)
```

Commentaire :

La première ligne exporte l'ensemble du système de fichiers vers les machines maître et confiance. En plus des droits d'écriture, toute conversion d'UID est abandonnée pour l'hôte confiance.

La deuxième et la troisième ligne montrent des exemples de noms d'hôtes génériques, et de sous−réseaux (`@trusted').

La quatrième ligne montre une entrée pour le client PC/NFS présenté plus haut.

La dernière ligne exporte un répertoire public de FTP, à tous les hôtes dans le monde, en effectuant les requêtes sous le compte anonyme. L'option `insecure` permet l'accès aux clients dont l'implémentation NFS n'utilise pas un port réservé.

29.2.4.2. L'identité des utilisateurs

Un des problèmes de NFS va être la gestion des utilisateurs et de leurs droits. Le serveur NFS va tenter d'identifier les users de la machine cliente par rapport au système habituel d'authentification de la machine. Tant que le client et le serveur ne se mettent pas d'accord sur une base commune d'identification, les confusions, voire les attaques, sont possibles. En effet, comment gérer les droits de l'utilisateur Pierre qui vient de la machine client Pluton si notre serveur ne connait aucun utilisateur Pierre ? Et d'abord, comment sait on quel utilisateur est connecté ?

La réponse est simple : par son id. Or, si on a deux systèmes d'authentification différents et autonomes (par exemple, par `/etc/passswd`, sur chacune des machines), qui dit que l'id de Pierre sur la machine cliente ne sera pas celle de Paul sur la machine serveur ?

En fait, c'est encore pire : Si on ne met pas un système commun d'identification (et pour être sur, d'authentification), on est certain de confondre les utilisateurs. DE MANIERE GENERAL, IL VAUT MIEUX UTILISER NFS DANS UN ENVIRONNEMENT DE CONFIANCE

NFS offre de commandes qui permettent de manipuler les identifiants des utilisateurs afin de plier ceux−ci à notre système de sécurité (contrôle et modification des identifiants donnés par les clients).

29.2.5. Configuration et utilisation du client Unix/Linux

29.2.5.1. Programmes client

Comme vu plus haut, n'oubiez pas d'avoir **portmap** actif sur votre client. Les programmes clients à utiliser sont : **mount** et **showmount**

29.2.5.2. Le fichier `/etc/fstab`

Ce fichier contient une table des volumes montés sur le système. Il est utilisé par les daemons mount, umount, fsck. Les volumes déclarés sont montés au démarrage du système. Voici un extrait de fichier :

Exemple

```
/dev/hda1       /               ext2      defaults   1 1
/dev/hda2       swap            swap      defaults   0 0
/dev/fd0        /mnt/floppy     ext2      noauto     0 0
/dev/cdrom      /mnt/cdrom      iso9660   user,noauto,ro 0 0
ns1:/usr/local/man  /doc        nfs       rsize=8192,wsize=8192,timeo=14,intr
```

La dernière ligne indique que le volume `/usr/local/man`, situé sur le serveur `ns1`, et qui contient les pages du manuel est un volume nfs, monté sous le nom de local de `/doc`.

Ce fichier évite d'avoir à << monter >> manuellement des systèmes de fichiers ou d'avoir à indiquer les points de montage, bien que cela puisse s'avérer parfois nécessaire (utilisation ponctuelle d'une ressource...). L'option `auto` permet de préciser si le montage doit etre fait automatiquement au démarrage de la machine. L'option `noauto` permet d'indiquer le montage tel qu'il doit être fait, lors d'une demande manuelle de montage (pratique pour les disquettes, CD et autres lecteurs amovibles).

29.2.5.3. Montage manuel de système de fichiers

La commande souvent utilisée est de la forme **mount `-t TypeDeSGF` *`NomDeMontage VolumeMonté`***.

Vous pourrez avoir toutes les options avec la commande **man *`mount`*** ou une aide plus brève avec **mount `--help`**.

Exemple de montage : **`mount -t nfs` *`ns1:/usr/local/man /doc`***.

La forme standard de la commande **mount** est **mount `-t type` *`périphérique répertoire`*** avec :

Type : type de sgf (fat, vfat, nfs, ext2, minix....) pour nous c'est nfs

Périphérique : nom du fichier exporté sous la forme NomServeur:NomDossierExporté

Répertoire : nom du répertoire local de montage/

Le type de fichier que vous montez est de type nfs, vous utiliserez l'exemple de la commande ci−dessous :

mount −t nfs *serveurNFS:/usr/share/doc /mnt/doc*

Commentaire : la ligne de commande monte le répertoire exporté `/usr/share/doc` du serveur serveurNFS, sur le répertoire local du client `/mnt/doc`.

Le `mtab` est modifié chaque fois que l'utilisateur << monte >> ou << démonte >> un système de fichiers. Le système tient à jour une table des volumes montés.

Attention, l'accès à la commande **mount** n'est, par défaut, autorisée que pour root.

Il faut rajouter l'option `user` dans le fichier `/etc/fstab`, afin qu'un autre utilisateur puisse accéder à cette commande.

Exemple : /dev/cdrom /mnt/cdrom iso9660 noauto,ro

devient /dev/cdrom /mnt/cdrom iso9660 user,noauto,ro

La prise en compte des modifications est dynamique.

La commande **mount** sans paramètres donne la liste des volumes montés. La commande consulte la table maintenue à jour dans le fichier `mtab`.

29.2.5.4. La commande showmount

Cette commande permet d'interroger un hôte distant sur les services NFS qu'il offre, et notamment les volumes qu'il exporte.

showmount −e AdresseIP_ou_NomIP lancée à partir d'un client nous affichera la liste des ressources offertes par *sAdresseIP_ou_NomIP* (=serveur).

Sur le serveur, **showmount −a** nous affichera la liste des clients connectés sur chacune de nos ressources.

De même, sur le serveur, la command **showmount −e** affiche le liste des partages en cours.

29.2.5.5. Autres commandes d'administration

rpcinfo : (par exemple **rpcinfo −p** consulte le catalogue des applications RPC (nfsd, mountd sont des applicatifs RPC parmi d'autres).

nfsstat : fournit des statistiques d'utilisation de NFS.

La commande **exportfs** permet elle aussi d'obtenir la liste des partages en cours, de relancer le service (pour la prise en compte d'éventuelles modifications du fichier `/etc/exports`, voir même d'effectuer un partage à la volée (sans passer par `/etc/exports`).

exportfs −v affiche les partages en cours.

exportfs −r active les changements fait dans le fichier de configuration de partage NFS (il fait relire le fichier `/etc/exports` par le programme serveur).

exportfs `machine:/repertoire` offre à la volée à machine (qui peut être aussi bien un nom de machine qu'étoile, ou un réseau) le partage `/repertoire`. On peut passer des options avec −o.

Exemple

```
#exportfs −v
  /tmp   pluton(rw,root_squash)
#exportfs −o rw,no_root_squash 192.168.*:/opt/sav
#exportfs −v
  /tmp   pluton(rw,root_squash)
  /opt/sav      192.168.*(rw,no_root_squash)
```

Chapitre 30. Travaux pratiques : partages NFS

30.1. Première partie

Vous allez configurer un service de partage de disque pour un client Unix. Vous serez, au cours du TP, serveur pour un autre binôme puis client du serveur d'un autre binôme. Vous allez créer deux répertoires partagés qui seront accessibles par le client :

- `/tmp` sur le serveur sera accessible en lecture/écriture
- `/usr/share/doc` sur le serveur sera accessible en lecture pour le client.

Ces répertoires seront montés respectivement sur les répertoires locaux `/mnt/tempo` et `/mnt/doc`

Vous pourrez utiliser les commandes **man *exports***, **man *mount***, **man *showmount***, **man *fstab***, **man *rpcinfo***.

1. Créez sur le serveur le fichier `/etc/exports`, et déclarez les répertoires exportés.

   Activez le service `portmap`. Vérifiez qu'il est bien actif.

   Voici un exemple de ce que vous pouvez obtenir avec **rpcinfo −p** :

```
 program no_version protocole  no_port
     100000    2   tcp    111  portmapper
     100000    2   udp    111  portmapper
     100011    1   udp    725  rquotad
     100011    2   udp    725  rquotad
     100003    2   udp   2049  nfs
     100005    1   udp   1026  mountd
     100005    1   tcp   1047  mountd
     100005    2   udp   1026  mountd
     100005    2   tcp   1047  mountd
```

2. Vérifiez sur le serveur les fichiers exportés avec la commande **showmount −e**

   Attention, si vous montez une arborescence sur un répertoire local, et que ce répertoire contenait des fichiers, ces derniers seront masqués le temps du montage.
3. Créez sur le client les points de montage, montez les dossiers exportés du serveur et testez les accès à partir du client.
4. Vérifiez les permissions d'accès lecture et lecture/écriture.
5. À partir du client, créez un fichier sur le `fs` (*file system*) accessible en écriture.
6. Ouvrez une autre session sur le serveur dans un autre terminal et essayez de démonter les répertoire montés. Que se passe t−il, pourquoi ?
7. Vérifiez sur le serveur les fichiers exportés avec la commande **showmount −a**.
8. Démontez les systèmes de fichiers.

30.2. Deuxième partie

1. Éditez et modifiez le fichier sur le client afin d'inclure les systèmes de fichiers nfs exportés par le serveur. Utilisez l'exemple que vous avez dans `/etc/fstab`.
2. Rajoutez les lignes nécessaires en vous servant de l'exemple ci−dessous.

```
serveurNFS:/usr/share/doc /mnt/doc nfs  user
```

3. Vérifiez que les modifications que vous avez apportées dans le fichier `fstab` fonctionnent.
4. Supprimez l'option `user` sur les lignes que vous avez mises dans le fichier `fstab`, enregistrez. Essayez ensuite de monter l'arborescence en utilisant un compte autre que << root >>. Que se passe−t−il ?
5. Restaurez l'environnement.

30.3. Troisième partie

1. Créez un utilisateur Linus sur la machine client, et donnez lui l'UID 1100. Créez un utilisateur Larry sur le serveur (UID 1100).
2. Réalisez un partage de `/tmp` sur le serveur, que le client montera dans /mnt.
3. Sur le client, Linus crée un fichier dans `/mnt`. Vérifiez son propriétaire
4. Sur le serveur, vérifiez le contenu du répertoire `/tmp`. A qui appartient le fichier crée ?
5. De façon théorique, pour quel utilisateur cela est il encore plus dangereux ? Quels sont les UIDs que vous connaissez tous à l'avance ?

   Consultez la page de man de exports (c'est un fichier de configuration, donc dans le chapitre 5 : man 5 exports, et plus largement man man).
6. Dans un environnement de grande confiance, vous voulez donner le droit à un root distant d'être reconnu comme root sur le partage. Comment faire ? Testez.
7. Interdisez ensuite seulement à root d'être confondu avec le root local, mais laissez les autres utilisateurs être eux mêmes. Testezxi. (Ceci est le comportement par défaut).
8. Sachant que l'UID de l'utilisateur nobody, qui existe certainement sur votre machine, est certainement 65534, trouvez une solution pour donner à toute personne se connectant l'identité de nobody. Créez un nouvel utilisateur BillG sur votre client, en vérifiant bien que son UID n'existe pas dans la sécurité locale du serveur.
9. Montez le répertoire partagé à partir du client, et testez la création et l'accès aux fichiers avec les utilisateurs BillG,Linus, et root. Que remarquez vous ?
10. Comment faire pour que tous les utilisateurs soient transformés en nobody ? Testez.
11. Vous devez maintenant offrir un partage de `/tmp`, mais chaque machine qui se connectera sur ce partage doit être différenciée (peu importe l'utilisateur de cette machine). On vous propose de créer sur le serveur des utilisateurs

client1, client2 et client3. Chaque machine qui se connectera se verra affublée de chacun de ces comptes (tout utilisateur de la machine client PC1 sera l'utilisateur client1 sur le serveur, tout utilisateur du pc client PC2 sera reconnu comme utilisateur client2 sur le serveur, et ainsi de suite. Créez le fichier `/etc/exports` correspondant à ces contraintes. Testez, et validez votre solution.

Chapitre 31. Installation d'un service de messagerie

Les protocoles SMTP, POP et IMAP − Fiche de cours

Les protocoles SMTP, POP et IMAP − Fiche de cours

Comment installer un serveur SMTP, un client IMAP et un client POP3

31.1. Le service de messagerie électronique

La messagerie électronique est une application très importante et des plus utiles des réseaux. Plus rapide et moins onéreuse que la plupart des autres moyens de communication (télécopie, téléphone, courrier postal, coursier...) la messagerie électronique est un vecteur de plus en plus important dans la communication aussi bien interne qu'externe. Dans l'univers des réseaux TCP/IP, la messagerie SMTP (*Simple Mail Transport Protocol*) est de loin la plus utilisée, notamment avec sendmail qui est le standard en matière de serveur SMTP sur les machines Unix.

Le logiciel libre Postfix est un gestionnaire de messagerie simple à configurer et conçu pour une sécurité optimale. De plus il est peu gourmand en ressources système et constitue donc une véritable alternative à Sendmail. Le choix de Postfix est légitime tant pour le traitement de flux importants de messages que pour de petites installations.

L'objectif de ce cours est de préparer l'installation et la mise en exploitation de Postfix en lieu et place de Sendmail.

31.2. Terminologie

31.2.1. MHS, MTA, UA, DUA

Le MHS (*Message Handler System*) est le système global de messagerie,

Le MTA (*Message Transfert Agent* est composé d'agents. Un agent de routage (sendmail, MS eXchange...) et un agent de transport (SMTP, UUCP).

L'agent de routage a pour but d'acheminer le message, en fonction de l'adresse vers son destinataire. Pour nous, avec l'environnement Linux, l'agent de routage est sendmail. L'agent de transport reçoit un message et une direction. Il ne prend aucune décision sur la route à utiliser. Pour nous, protocole de transport peut être SMTP ou UUCP. Le logiciel Sendmail assure les deux fonctions de transport et de routage.

L'UA ou MUA, Message User Agent, est le programme utilisé par le client pour composer, envoyer et recevoir les messages. Pour la composition et l'envoi des messages il existe des programmes comme mail sous Linux. D'autres programmes sont utilisés comme Eudora, Netscape, kmail... On appelle souvent l'UA un << mailer local >> si on utilise des outils comme Eudora, Oulook, Mutt, Kmail, ou un << web mail >> si on utilise un navigateur comme Mozilla, Netscape ou Internet explorer pour consulter sa messagerie. Ces outils utilisent des protocoles différents. Les protocoles utilisés sont SMTP ou UUCP pour envoyer, et POP3, IMAP, POP3s, IMAPs pour recevoir.

Il existe également un agent (DUA − *Delivery User Agent*) pour la remise physique du courrier entrant dans la boîte aux lettres de l'utilisateur (BAL). Sur Linux nous utilisons procmail. Cette remise locale (*local delivery*) est réalisé par un agent (mail, procmail...) dans des boîtes aux lettres (`mailbox`) pour mémorisation, (`/var/mail/dupont`, `/var/spool/mail/dupond`).

Figure 31−1. Message Handler System

31.3. Historique et évolution de sendmail

Sendmail est le routeur de courrier depuis 1982. Il répond aux préconisations de la RFC 822. En 1993, né le standard MIME − RFC 1521 (*Multipurpose Internet Mail Extensions*), puis en 1994 les extensions du service SMTP (RFC 1652, 1869) pour le transfert caractères 8 bits.

31.3.1. MIME

Le but de MIME est de standardiser les méthodes de transfert de données 8 bits, structurer le corps du message en contenus (body−parts), standardiser les différents contenus possibles. Un en−tête est rajouté à ceux définis dans le RFC 822 : Mime−version:1.0

31.3.1.1. Le standard MIME

MIME supporte plusieurs type d'encodage comme :

1. Texte 7 bits, US−ASCII
2. Quoted−Printable (Caractère non US−ASCII remplacé par une séquence =XY, XY étant le code hexadécimal du caractère.)
3. Base 64 (Texte, image, son)
4. 8Bits (les lignes sont composées de caractères 8 bits, il faut préciser l'alphabet : iso−latin1)
5. Binary

La strucure d'un message MIME est standardisée par des en−têtes supplémentaires qui décrivent la structure et le type de contenu (format des données) du message.

Exemple de déclaration décrivant la structure :

1. Multipart/mixed
2. Multipart/parallel (plusieurs parties avec affichage en parallèle.)
3. Multipart/digest (d'autres messages inclus dans le message)
4. Multipart/alternative (partie du message affichée suivant l'environnement du correspondant.)

Exemple de déclaration décrivant le format des données

1. Text/plain : charset=iso−8859−1
2. Text/richtext
3. Image/gif
4. Image/jpeg
5. Audio/basic
6. Video/mpeg
7. Application/octet−stream : exemple word
8. Application/postscript

31.3.1.2. Exemple de message

```
From mascret  Mon Mar 19 08:02:46 2001
Return−Path: <Marcel.Giry@unilim.fr>
Delivered−To: alix.mascret@beaupeyrat.com
Received: from limdns2.unilim.fr (limdns2.unilim.fr [164.81.1.5])
        by pegase.beaupeyrat.com (Postfix) with ESMTP id AC04237B05
        for <salvaco@beaupeyrat.com>; Mon, 19 Mar 2001 08:02:44 +0100 (CET)
Received: from pctest (modem8.unilim.fr [164.81.1.208])
          by limdns2.unilim.fr (8.9.1a/jtpda−5.3.2) with ESMTP id IAA04253
          ; Mon, 19 Mar 2001 08:02:39 +0100
Message−Id: <4.2.0.58.20010319080303.00950a70@pop.unilim.fr>
X−Sender: xalan@pop.unilim.fr (Unverified)
X−Mailer: QUALCOMM Windows Eudora Pro Version 4.2.0.58
Date: Mon, 19 Mar 2001 08:05:13 +0100
To: salvaco@beaupeyrat.com,
 xalan@univlim.fr
From: Ximian Alan <xalan@univlim.fr>
Subject: Controle IUT2
Mime−Version: 1.0
Content−Type: multipart/mixed;
  boundary="====================_811307==_"
Status: RO
X−Status: A
```

31.3.1.3. L'importance d'un bon UA

MIME permet l'utilisation de plusieurs types de données (text, audion compressés...) et plusieurs format (rtf, doc, gz, zip...). Il est important de posséder un UA de bonne qualité.

1. Reconnaître et afficher du texte US−ASCII,
2. Reconnaître les autres jeux de caractères et permettre de sauvegarder les contenus non reconnus dans un fichier pour traitement ultérieur
3. Reconnaître et afficher les contenus de type message/RFC822
4. Reconnaître le type Multipart/mixed
5. Reconnaître le type Multipart/alternative
6. Traiter les Multipart non reconnus comme Multipart/mixed
7. Décoder les contenus de Application/* si l'encodage quoted−printable ou base64 est utilisé, puis offrir de sauver le résultat dans un fichier.

31.4. Pourquoi Postfix

Le serveur de messagerie standard sur les systèmes Unix est le serveur Sendmail. Sendmail a fait ses preuves. L'inconvénient est son mode de configuration. Toutes les fonctions de messagerie sont réalisées par un seul programme. Sa structure est dite monolithique et la configuration (fichier `sendmail.cf`) en est d'autant plus compliquée. Ce phénomène s'accroit avec l'amplification de l'utilisation du service de messagerie (augmentation de fréquence/volume) et avec l'exposition aux tentatives de piratage des serveurs de messagerie. Il existe d'autres serveurs de messagerie sur Unix (QMail, Z−mailer...) tous présentent des inconvénients au niveau utilisation de la bande passante, inter−opérabilté, respect des RFC, facilité de configuration, sécurité...

L'objectif de postfix est d'apporter une solution à ces différents problèmes.

31.4.1. Buts premiers : un nouveau MTA sous Unix

1. bénéficier de l'expérience de sendmail
2. facile à administrer : ce qui est facile à comprendre est plus facile à sécuriser.
3. rapide et évolutif : le trafic SMTP de 1999 n'est pas celui de 1980. Il faut pouvoir faire un logiciel supportant les sites énormes (ISP, Accès des grosses entreprises, ...)
4. compatibilité sendmail maximale

Il assure égalemetn un compatibilité et le support :

1. des MUA existants (pine,mutt,mail...)
2. des gestionnaires de liste (majordomo, sympa...)
3. des formats de boîte aux lettres (mh, mbox, qmail−dir, ...)
4. des agents d'acheminement local (procmail, deliver, cyrus...)
5. des configurations (UUCP, réécriture, mailertable, ...)
6. des utilisateurs (alias, .forward, ...)
7. des RFCs

31.4.2. L'Auteur

L'Auteur − Wietse Venema − est connu pour ses contributions à la sécurité et aux logiciels libres. Il est également auteur de TCP_Wrapper et d'un portmap sécurisé. Il est co−auteur avec D. Farmer de SATAN. Il travaille au Watson Research Center d'IBM.

Postfix est un logiciel libre. Le site officiel est www.postfix.org.

31.5. Architecture de postfix

Postfix (voir bigpicture.gif) est architecturé autour d'un module de réception des messages (voir inbound.gif) et de celui qui permet de délivrer ces messages (voir outbound.gif).

Figure 31−2. Architecture de Postfix

Figure 31−3. Réception des messages

Figure 31−4. Traitement des messages

31.5.1. La réception des messages (entrées)

Quand un message doit être traité par un système Postfix, le passage obligé est la file `incoming`.

Si le message est posté localement, il est déposé dans un répertoire en accès << écriture possible pour tout le monde >>. Le démon pickup le traitera à partir de là. Ce démon procède à une première phase d'analyse des couriers (headers) afin de protéger le reste du système.

Si le message provient d'un réseau, le message est traité par un serveur SMTP. Certaines règles de sécurité et de contrôles sont déjà effectuées.

Les messages peuvent être générés par Postfix lui même ou par un robot afin de prévénir l'administrateur des erreurs, adresses introuvables, tentatives de violations des règles, problèmes de protocoles...

Les messages peuvent être redistribués par des entrées dans les fichiers d'alias ou des fichiers `.forward`.

Le démon cleanup représente la période finale de traitement d'un message, notamment la vérification de l'entête du message (complétude user@fqdn), la réécriture d'adresse, le dépôt du message dans la file incoming, l'avertissement du gestionnaire de liste.

31.5.2. Délivrer les messages

Quand un message est arrivé dans la file `incoming`, l'étape suivante consiste à le délivrer. Ceci est pris en charge par le gestionnaire de file qui est le coeur du système de Postfix. Il contacte un agent (local, smtp, lmtp, pipe) chargé de délivrer les messages en lui communiquant des paramètres (localisation du message, nom/adresse de l'emetteur, nom(s)/adresse(s) du/des destinataires, machine hôte de destination...

Le gestionnaire de liste maintient une liste séparée pour les courriers ne pouvant être délivrés immédiatement (deferred).

Les messages ne pouvant être définitivement délivrés (bounces) génèrent une trace d'information dans les journaux.

Sur Linux, l'agent de traitement local des messages est le plus souvent procmail. Il doit pouvoir traiter des structures de boîtes aux lettres conformes au standard Unix, utiliser les aliases, les redirections `.forward`...

L'agent de traitement pour l'acheminement distant des messages s'appuie sur le protocole SMTP et utilise le port 25.

Les différents démons sont activés << à la demande >> par un super serveur (master daemon) un peu à la façon d'inetd.

31.5.3. Une fonction / un programme

Chaque grande fonction de postfix est prise en charge par un programme indépendant.

1. Lecture des messages locaux
2. Réception SMTP
3. Réécriture d'adresse
4. Envoi SMTP
5. Délivrance locale
6. Traitement des erreurs (bounces)
7. Gestion des files

31.5.4. Apports en termes de sécurité :

Cette option présente plusieurs avantages.

1. Décomposition = programmes plus petits et plus lisibles
2. Plus difficile à casser ou circonvenir
3. Chroot plus facile
4. Les programmes ne se font pas confiance : isolation de chaque fonction

31.5.5. Communication interprocessus par sockets Unix ou file (FIFO)

1. Portabilité aisée
2. Messages courts dans les sockets
3. Ne pas faire confiance aux données

31.5.6. Semi résidence

1. Les démons sont réutilisés et contrôlés par un super démon << master >> qui les crée à la demande.
2. Nombre maximum pour chaque fonction : contrôle précis du fonctionnement, sécurité contre le << déni de service >> (DOS)
3. Temps d'inactivité paramétrable

31.5.7. Files d'attente multiples

1. `maildrop` : messages locaux postés par sendmail
2. `incoming` : messages en cours de réécriture et de nettoyage
3. `active` : messages en cours ou en attente de transport
4. `deferred` : messages en attente
5. `defer` : arborescence d'attente (hachée pour éviter les trop gros répertoires −− problème dans Sendmail)

31.6. Configuration et fichiers de configuration de Postfix

Les outils d'administration et de maintenance sont dans `/usr/sbin`. Voici les principaux.

1. postalias sert à maintenir la base de données des alias
2. newaliases (`/usr/bin`) assure la compatibilité avec sendmail pour la base de donnes des alias
3. postcat affiche le contenu des files d'attentes.
4. postconf affiche les paramètres de Postfix contenus dans fichier `main.cf`
5. postlog, sert à gérer les logs (réalisation de scripts)
6. postqueue, permet de gérer et administrer les files d'attentes.

Les journaux (logs) sont dans `/var/log`

31.6.1. Configuration − extrait du fichier `/etc/postfix/master.cf`

Il définit les démons à lancer, leur nombre et les << transports >>

```
# ==========================================================================
# service type  private unpriv  chroot  wakeup  maxproc command args
#               (yes)   (yes)   (yes)   (never) (50)
# ==========================================================================
smtp      inet  n       −       y       −       −       smtpd
smtps     inet  n       −       y       −       −       smtpd \
          −o smtpd_tls_wrappermode=yes −o smtpd_sasl_auth_enable=yes
submission inet n       −       y       −       −       smtpd \
          −o smtpd_enforce_tls=yes −o smtpd_sasl_auth_enable=yes
pickup    fifo  n       n       y       60      1       pickup
cleanup   unix  −       −       y       −       0       cleanup
qmgr      fifo  n       −       y       300     1       qmgr
#qmgr     fifo  n       −       y       300     1       nqmgr
tlsmgr    fifo  −       −       y       300     1       tlsmgr
rewrite   unix  −       −       y       −       −       trivial−rewrite
bounce    unix  −       −       y       −       0       bounce
defer     unix  −       −       y       −       0       bounce
flush     unix  −       −       y       1000?   0       flush
smtp      unix  −       −       y       −       −       smtp
showq     unix  n       −       y       −       −       showq
error     unix  −       −       y       −       −       error
local     unix  −       n       n       −       −       local
virtual   unix  −       n       n       −       −       virtual
lmtp      unix  −       −       n       −       −       lmtp
[...]
```

31.6.2. Le fichier de configuration `/etc/postfix/main.cf`

Si postfix n'a pas été préalablement configuré, vous n'avez pas de fichier de configuration `main.cf`. Vous pouvez utiliser la commande :

```
dpkg−reconfigure postfix
```

Utilisez les paramètres suivants pour une configuration minimale :

```
#$NOM_MACHINE est le nom d'hôte de votre machine
local only
$NOM_MACHINE
Append Domain           no
Destination             $NOM_MACHINE
Local Network           127.0.0.0/8
Use Procmail            Yes
Siez Mail Box           0
Char Def Local Adress   +
```

Le fichier `main.cf` contient tous les paramètres de postfix. Ceux−ci peuvent être affichés avec la commande **postconf**.

Voici un exmple de `main.cf` que vous pourrez réutiliser pour les TP

```
# Vous avez un fichier complet et commenté
# /usr/share/postfix/main.cf.dist

command_directory = /usr/sbin
daemon_directory = /usr/lib/postfix
program_directory = /usr/lib/postfix

smtpd_banner = $myhostname ESMTP $mail_name (Debian/GNU)
setgid_group = postdrop
biff = no
2bounce_notice_recipient = postmaster

# appending .domain is the MUA's job.
append_dot_mydomain = no
myhostname = NomHote.foo.org
```

```
mydomain = foo.org
mydestination = $myhostname, localhost.$mydomain $mydomain
myhostname = NomHote.foo.org
myorigin = $mydomain
myorigin = /etc/mailname

alias_maps = hash:/etc/aliases
alias_database = hash:/etc/aliases
# /etc/mailname contient l'équivalent de $MYHOSTNAME

mynetworks = 127.0.0.0/8 192.168.0.0/24
mailbox_command = procmail −a "$EXTENSION"
mailbox_size_limit = 0
recipient_delimiter = +
relay_domains = $mydestination
relayhost = $mydomain
smtpd_recipient_restrictions = permit_mynetworks,check_relay_domains
```

Pour une configuration initiale remplir myhostname, mydomain, myorigin, mydestination, relayhost.

31.6.3. Le fichier de configuration des aliases `/etc/aliases`

Il sert à la création des aliases, par exemple jean.dudognon sera l'alias du compte système jddgn. Le courrier sera adressé à jean.dudognon@domaine.dom, mais sera délivré dans la boîte du compte jddgn, c'est à dire physiquement dans `/var/spool/mail/jddgn`.

Le fichier `/etc/postfix/aliases` est de type texte. C'est celui−ci que vous modifiez. Après chaque modification du fichier source utiliser la commande **newaliases** ou **postaliases *`hash:/etc/postfix/aliases`*** qui met à jour le fichier de bases de données `/etc/postfix/aliases.db`.

31.6.4. Surveillance et maintenance de postfix

La maintenance est réalisée à l'aide des commandes externes. Autrement les transactions sont journalisées par le démon syslogd.

```
Oct 31 11:23:26 uranus postfix/master[2745]: daemon started

    uranus postfix/smtpd[2753]: connect from unknown[192.168.1.1]
    uranus postfix/smtpd[2753]: 82BF05769B: client=unknown[192.168.1.1]
    uranus postfix/cleanup[2754]: 82BF05769B:
           message−id=<20011031102453.82BF05769B@uranus.foo.org>
    uranus postfix/qmgr[2749]: 82BF05769B:
           from=<mlx@foo.org>, size=318, nrcpt=1 (queue active)
    uranus postfix/local[2756]: 82BF05769B:
           to=<mlx@foo.org>, relay=local, delay=94,
           status=sent ("|/usr/bin/procmail /etc/procmail.rc")
    uranus postfix/smtpd[2753]: disconnect from unknown[192.168.1.1]
```

31.7. Structure des messages

Un messages est schématiquement composé de deux parties, une entête et un corps. Ces deux parties sont séparées par une ligne blanche.

```
L'entête est découpée ainsi :
- FROM:         expéditeur
- TO:           destinaire(s)
- CC:           copie à
- BCC:          copie  aveugle
- REPLY−TO:     adresse de réponse
- ERROR−TO:     adresse en cas d 'erreurs
- DATE:         date expédition
- RECEIVED      informations de transferts
- MESSAGE−ID:   identificateur unique de msg
- SUBJECT:      sujet
```

31.8. Le dialogue entre le client et le serveur

Le dialogue est défini par le protocole SMTP selon un schéma client/serveur. Sur le client, un démon (programme sendmail ou smtpd par exemple) attend les requêtes TCP sur le port 25 d'un client (le programme mail par exemple).Le dialogue est en ASCII. Pour tester utilisez la commande **`telnet` *`Serveur_SMTP`* `25`** ou encore **`sendmail −v −bs`**.

Exemple de dialogue : la chaîne << >>> >> n'apparaît pas, c'est juste pour distinguer les commandes client.

```
[mlx@uranus mlx]$ telnet localhost 25
Trying 127.0.0.1...
Connected to localhost.localdomain (127.0.0.1).
Escape character is '^]'.
220 uranus.foo.org ESMTP Postfix
>>> EHLO uranus
```

```
250-uranus.foo.org
250-PIPELINING
250-SIZE 10240000
250-ETRN
250 8BITMIME
>>> MAIL FROM:<mlx@uranus.foo.org>
250 Ok
>>> RCPT TO:<mlx@foo.org>
250 Ok
>>>DATA
354 End data with <CR><LF>.<CR><LF>
Message de test
.
250 Ok: queued as C21B15769B
>>> QUIT
221 Bye
Connection closed by foreign host.
You have new mail in /var/spool/mail/mlx
```

31.9. PostOFFICE

Le service POP − Postoffice Protocole est utilisé par les logiciels clients (netscape, Eudora, Outlook...) pour relever le courrier sur les serveurs de messagerie. Le client pop utilise un couple Nom d'utilisateur/mot de passe pour la phase identification/authentification par le serveur. Le service pop3d reste en écoute sur le port 110. Il est généralemnt lancé par le démon inetd ou xinetd. Pour activer le service pop3 il suffit de décommenter la ligne correspondante dans le fichier `/etc/inetd.conf`. Voici des exemples de lignes que vous pouvez avoir dans votre fichier `inetd.conf` :

```
#:MAIL: Mail, news and uucp services.
imap2   stream  tcp     nowait  root   /usr/sbin/tcpd /usr/sbin/imapd
imaps   stream  tcp     nowait  root   /usr/sbin/tcpd /usr/sbin/imapd
pop3    stream  tcp     nowait  root   /usr/sbin/tcpd /usr/sbin/ipop3d
pop3s   stream  tcp     nowait  root   /usr/sbin/tcpd /usr/sbin/ipop3d
#imaps   stream  tcp nowait  root    /usr/sbin/tcpd  \
                 /usr/sbin/sslwrap  -nocert -addr 127.0.0.1 -port 143
#pop3s   stream  tcp nowait  root    /usr/sbin/tcpd  \
                 /usr/sbin/sslwrap  -nocert -addr 127.0.0.1 -port 110
```

ou d'activer le service << disable = no >> dans `/etc/xinetd.d/pop3`, si vous utilisez xinetd.d.

31.10. IMAP (*Internet Message Access Protocol*)

Pop a été conçu pour la consultation << hors ligne >>, (*off line*). IMAP permet la consultation hors ligne, mais également << en ligne >>, selon un processus interactif entre le client et le serveur. Les messages ne sont plus rapatriés sur le client. Ils restent en dépôt sur le serveur jusqu'à ce que l'utilisateur demande explicitement la suppression ou le transfert.

Ce procédé est particulièrement interessant pour les utilisateurs mobiles. Ils peuvent consulter leur messages à partir de machines ou de lieux non définis à l'avance.

Comme la connexion au serveur est permanente pendant la durée du traitement, il présente l'inconvénient d'un surcoût financier du à la liaison téléphonique.

Ces services utilisent des protocoles/ports différents. Ils peuvent cohabiter simultanément sur le même serveur. Un utilisateur peut utiliser selon ses besoins l'un ou l'autre des services POP ou IMAP.

Vous devrez installer et configurer sur les postes clients, un client IMAP (Netscape messenger, Kmail...).

Des logiciels d'interface sur le serveur comme IMP (www.imp.org), permettent de transformer le serveur IMAP en serveur << webmail >>. Les clients pourront alors utiliser n'importe quel navigateur pour consulter leur boîte aux lettres. Le CRU, (Comité Réseau des Universités − www.cru.fr), a fait une étude sur les principaux produits qui pouvaient être utilisés.

31.11. Remarques sur pop3 et imap

Les ports utilisés par les services pop3, pop3s, imap, impas, sont déclarés dans le fichiers `/etc/services`. Voici ce que donne le lancement d'inetd :

```
root@knoppix:/home/knoppix# netstat -atup | grep LISTEN
tcp        0      0 *:imaps     *:*         LISTEN      376/inetd
tcp        0      0 *:pop3s     *:*         LISTEN      376/inetd
tcp        0      0 *:pop3      *:*         LISTEN      376/inetd
tcp        0      0 *:imap2     *:*         LISTEN      376/inetd
```

Dans un soucis de sécurité, les applications récentes ne supportent plus les transactions << en clair >> sur le réseau. Cela signifie que les applications sont compilées pour utiliser les protocoles d'encryptage TLS/SSL. Vous devrez en tenir compte dans la configuration de vos clients et vous utiliserez pop3s et imaps.

Pour en savoir plus, vous pouvez consulter :

```
/usr/share/doc/ipopd/README.Debian
```

```
/usr/share/doc/libc-client2003debian/README.Debian
/usr/share/doc/libc-client2003debian/md5.
/usr/share/doc/libc-client2002/md5.txt
/usr/share/doc/libc-client2003debian/imaprc.txt
```

Chapitre 32. Travaux pratiques : configuration d'un système de messagerie

Les protocoles SMTP, POP et IMAP − TP

Les protocoles SMTP, POP et IMAP − TP

Comment installer un serveur SMTP, un client IMAP et un client POP3

32.1. Installation de postfix

Vous allez installer successivement :

1. le serveur de messagerie postifx puis tester son fonctionnement,
2. un serveur de remise de courrier pop3 et imap
3. un client pop3 et imap puis tester son fonctionnement,

Pour imap on utilisera uw−imapd/uw−imapd−ssl, pour pop3, on utilisera ipopd. Pour les préconfigurer vous utiliserez les commandes :

```
dpkg-reconfigure uw-imapd
dpkg-reconfigure ipopd
```

Vous sélectionnerez pop3 et pop3/ssl, imap4 et imap/ssl. Attention pour imap4, c'est imap2 qu'il faut sélectionner. C'est bizarre mais c'est comme ça ;−) imap3 est devenu obsolète.

La procédure de configuration, génère des certificats dans `/etc/ssl/certs/`

32.2. DNS − préparation préalable

Vous allez déjà préparer votre serveur de nom. Le serveur de nom primaire sera également serveur SMTP (enregistrement MX − Mail eXchanger). Si votre serveur de nom s'appelle `ns1`, rajouter les enregistrements suivants dans le fichier de configuration de votre zone :

```
# On définit la machine qui achemine le courrier pour
# user@ns1.VotreDomaine.Dom
@       IN      MX      10      ns1.VotreDomaine.Dom
#On définit un alias pour le courrier envoyé à
mail    IN      CNAME           ns1
#On définit un alias pour le courrier envoyé à partir de
smtp    IN      CNAME           ns1
#On définit un alias pour le serveur pop et pour imap
pop     IN      CNAME           ns1
imap    IN      CNAME           ns1
```

Relancer le service dns. Les commandes suivantes doivent fonctionner à partir d'un client du domaine :

```
ping ns1.VotreDomaine.Dom
ping smtp.VotreDomaine.Dom
ping mail.VotreDomaine.Dom
ping pop.VotreDomaine.Dom
ping imap.VotreDomaine.Dom
```

32.3. Configuration du serveur postifx.

Vous allez successivement configurer un serveur SMTP Postifx, tester la configuration, installer les serveur pop et imap, tester le fonctionnemnt de l'ensemble.

32.3.1. Installation du serveur SMTP

Configurer votre machine pour un service minimum (pas de liste, pas de réécriture d'adresse...).

Utilisez l'exemple de configuration de `main.cf` et la liste des variables à configurer donnés dans la fiche de cours, afin de mettre en place un service minimum.

Activez le service avec la commande **`/etc/init.d/postfix start`**. Vérifiez le bon démarrage du serveur dans le fichier de log et dans la table des processus (**ps *`axf`***). Vous devriez obtenir quelque chose comme :

```
 2745 ?        S      0:00 /usr/lib/postfix/master
 2748 ?        S      0:00  \_ pickup -l -t fifo -c
 2749 ?        S      0:00  \_ qmgr -l -t fifo -u -c
 2750 ?        S      0:00  \_ tlsmgr -l -t fifo -u -c
```

Vérifiez également les traces dans le fichier de journalisation.

32.3.2. Test de la configuration du serveur SMTP

Créez sur la machine locale deux comptes systèmes pour les tests. `cpt1` et `cpt2` par exemple.

Ouvrez une session sous le compte `cpt1` afin de réaliser un envoi de mail pour `cpt2`.

Lancez une transaction **`telnet VotreServeur 25`**, et réalisez un dialogue similaire à celui décrit en TD. Le message doit être délivré dans la boîte de `cpt2`. Utilisez la commande **`ps axf`** pour voir le chargement des différents démons.

Rélisez l'opération à l'aide du programme mail. Vérifiez que le message est bien délivré.

Avant de terminer la transaction, identifiez la session avec la commande **netstat** :

```
netstat -atup | grep ESTABLISHED
```

32.3.3. Installation du serveur PostOFFICE Pop3

La configuration du service pop est des plus simple. Il est même possible qu'il soit déjà actif. Décommentez la ligne dans le fichier `/etc/inetd.conf` ou utilisez **dpkg-reconfigure**.

Vérifier le fichier `/etc/inetd.conf`, relancez au besoin le service inetd.

Pop and imap mail services

Avec xinet, la configuration est dans `/etc/xinetd.d`. Éditez les fichiers correspondant aux différents services. Par exemple le fichier `/etc/xinetd.d/pop3s`.

```
# default: off
# The POP3S service allows remote users to access their mail \
# using an POP3 client with SSL support such as fetchmail.

service pop3s
{
        socket_type             = stream
        wait                    = no
        user                    = root
        server                  = /usr/sbin/ipop3d
        log_on_success          += USERID
        log_on_failure          += USERID
        disable                 = no
}
```

Dans `inetd.conf`, décommentez la ligne. Dans xinetd, mettez la variable `disable` à *no*.

Relancer le service inetd ou xinetd, vérifiez l'ouverture des ports avec la commande **netstat**.

Identifiez les numéros de ports des services dans le fichier `/etc/services`.

32.3.4. Test du serveur Pop3

Vous allez réaliser l'opération à partir de la machine locale et d'une machine distante. La résolution de nom doit fonctionner, sinon utilisez les adresses IP. Vous utiliserez kmail ou le client de messagerie de Mozilla.

1. Sur la machine locale qui est votre serveur SMTP et serveur POP3, configurez le client de messagerie avec les paramètres suivants :

   ```
   Serveur smtp : Nom de votre serveur
   Serveur POP     :       Nom de votre serveur POP
   Votre compte d'utilisateur
   Votre mot de passe
   ```

   Testez l'envoi et la réception de message.

   Renseignez bien le numéro de port. Dans kmail, l'onglet extras vous donne accès à un bouton tester ce que le serveur peut gérer, et va vous renseigner sur le support de ssl ou tls du serveur.

Avec un client pop, les messages sont, par défaut, téléchargés depuis le serveur sur le client. La procédure supprime les fichiers téléchargés sur le serveur. Cette option est configurable sur la majorité des clients.

Vérifiez que les fichiers sont bien supprimés sur le serveur.

Réitérez l'envoi de message en mettant un fichier attaché (par exemple un fichier `xls`). Vérifier et relevez la description MIME du message.

2. Sur un client configurez Nestcape Messenger avec les paramètres suivants :

```
Serveur smtp : Nom de votre serveur (Machine distante)
Serveur POP     :       Nom de votre serveur POP3 (Machine distante)
Votre compte d'utilisateur
Votre mot de passe
```

Testez l'envoi et la réception de message.

Identifiez les transactions dans le fichier de log.

32.3.5. Utilisation des alias

Créez un compte utilisateur `pn`,

Créez un alias `prenom.nom` pour ce compte système dans le fichier `/etc/aliases`.

Mettez à jour le fichier `/etc/aliases/db`.

Vérifiez que les messages envoyés à :

pn@votredomaine ou prenom.nom@votredomaine doivent tous être correctement délivrés.

32.3.6. Utilisation des listes

Ouvrez à l'aide d'un éditeur le fichier `/etc/aliases`

Créez une liste de la façon suivante :

```
maliste: cpt1, cpt2
```

Enregistrez et régénérez le fichier `aliases.db`

Envoyez un message à maliste@foo.org, vérifiez que tous les membres de la liste ont bien reçu le message.

32.3.7. La gestion des erreurs

Certaines erreurs << systèmes >> sont gérés par un compte particulier << MAILER_DAEMON >>. Ce compte est en général un alias vers `postmaster`, qui, lui même redirige sur le compte de l'administrateur en fonction. Il agit commem un << robot >> notamment quand un message ne peut être délivré.

Procédure de test

Envoyez un message à QuiNexistePas@foo.org

Relevez vos nouveau messages. Normalement, vous êtes averti que votre message n'a pas pu être délivré.

32.3.8. Mise en place du service IMAP sur le serveur

La mise en oeuvre est identique à celle du service Pop3. Configurer le fichier `inetd.conf` ou le fichier `/etc/xinetd.d/imap`. Relancer le service serveur (inetd ou xinetd).

Vous allez tester le bon fonctionnement de votre serveur : la commande **`ps aux | grep imap`** ne donne rien car le serveur imap est lancé << à la demande >> par le serveur inetd ou xinetd.

Par contre la commande **`netstat –a | grep LISTEN | grep imap`** montre bien qu'un port est bien ouvert en << écoute >>.

```
tcp        0      0 *:imap                  *:*               LISTEN
```

Saisissez la commande **`telnet DeVotreServeurImap 143`** pour activer le service imap. Le serveur doit répondre :

```
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
* OK [CAPABILITY IMAP4 IMAP4REV1 STARTTLS LOGIN-REFERRALS AUTH=LOGIN]\
   uranus.foo.org IMAP4rev1 2000.287rh at Sat, 17 Nov 2001 14:14:42 +0100 (CET)
```

Dans une autre session xterm, la commande **ps aux | grep *imap*** montre mainetant que le service est maintenant bien dans la liste des processus :

```
root     11551  0.0  1.1  3664 1448 ?        S    14:14   0:00 imapd
```

et la commande **ps axf**

```
  231 ?        S      0:00 /usr/sbin/inetd
  315 ?        S      0:00  \_ imapd
```

montre bien que le processus imapd dépend (est fils de) inetd.

32.3.9. Plus loin dans le décryptage

La commande **netstat a | grep *imap*** donne l'état d'une connexion établie entre un client et le serveur.Socket client TCP sur le port 1024

```
tcp        0      0 *:imaps                 *:*                     LISTEN
tcp        0      0 *:imap2                 *:*                     LISTEN
tcp        0      0 knoppix:imap2           knoppix:1025            ESTABLISHED
tcp        0      0 knoppix:1025            knoppix:imap2           ESTABLISHED
```

La commande **fuser *1025/tcp*** utilise le pseudo−système de fichiers d'informations sur les processus `/proc` pour identifier << QUI >> utilise la connexion tcp sur le port 1025.

```
root@knoppix:/home/knoppix# fuser 1025/tcp
1025/tcp:             364
```

La commande **ls −l */proc/364*** donne les indications sur le programme qui utilise cette connexion et montre que c'est une commande **telnet** qui a déclenché le processus.

```
root@knoppix:/home/knoppix# ls −al /proc/364
total 0
dr−xr−xr−x 3  knoppix  knoppix  0 2003−04−16 15:06 .
dr−xr−xr−x 49 root     root     0 2003−04−16 16:52 ..
−r−−r−−r−− 1  knoppix  knoppix  0 2003−04−16 15:06 cmdline
−r−−r−−r−− 1  knoppix  knoppix  0 2003−04−16 15:06 cpu
lrwxrwxrwx 1 knoppix  knoppix 0 2003−04−16 15:06 cwd −> /home/knoppix
−r−−−−−−−− 1 knoppix  knoppix 0 2003−04−16 15:06 environ
lrwxrwxrwx 1 knoppix  knoppix 0 2003−04−16 15:06 \
                                          exe −> /usr/bin/telnet−ssl
dr−x−−−−−− 2 knoppix  knoppix 0 2003−04−16 15:06 fd
−r−−r−−r−− 1 knoppix  knoppix 0 2003−04−16 15:06 maps
−rw−−−−−−− 1 knoppix  knoppix 0 2003−04−16 15:06 mem
−r−−r−−r−− 1 knoppix  knoppix 0 2003−04−16 15:06 mounts
lrwxrwxrwx 1 knoppix  knoppix 0 2003−04−16 15:06 root −> /
−r−−r−−r−− 1 knoppix  knoppix 0 2003−04−16 15:06 stat
−r−−r−−r−− 1 knoppix  knoppix 0 2003−04−16 15:06 statm
−r−−r−−r−− 1 knoppix  knoppix 0 2003−04−16 15:06 status
```

et voir la commande qui a activé cette connexion **more */proc/364/cmdline*** qui retourne `telnet localhost 143`.

32.3.10. Mise en place du client IMAP

Utilisez Mail & NewsGroup de Mozilla ou kmail par exemple. Dans Mail & News Group, allez dans le menu de configuration (Edit) et ajoutez un compte. Prenez un compte imap.

Complétez la configuration de votre client de messagerie

Ouvrez l'application Messenger, testez l'utilisation du client IMAP.

32.3.11. Le relayage

Utiliser le << relayage >> consiste pour un client A à utiliser le service serveur SMTP d'un domaine B pour inonder de messages (*spammer*) des boîtes aux lettres. Les serveurs sont généralement configurés pour empêcher le relayage. Dans Postifx, cette option est configurée par défaut.

Le relayage pose plusieurs problèmes. Remplissage des boîtes aux lettres sans l'accord des destinataires, utilisation des ressources disques et CPU à l'insu des sociétés qui relaient les courriers...

Afin de combattre un peu le phénomène, une société qui relai les messages peut se voir << black listée >>, c'est−à−dire inscrite dans une liste noire référencée. Il existe plusieurs sites référençant ces listes noires. Certains de ces messages ne seront plus distribués. Voir pour cela, http://mail−abuse.org/rbl/, page principale de MAPS (*Mail Abuse Prevention System LLC*) RBLSM (*Realtime Blackhole List*).

Il est possible d'utiliser ces bases de données pour empêcher le relayage, ou refuser de délivrer les messages d'un site << black listé >>.

```
maps_rbl_domains = rbl.maps.vix.com
maps_rbl_reject_code = 554
reject_maps_rbl
```

Vous allez activer la fonction de relayage sur votre serveur et tester son comportement. Modifiez la ligne :

```
relay_domains = $mydestination
```

par

```
relay_domains = $mydestination, domaine1.dom, domaine2.dom...
```

où domaine1.dom, domaine2.dom... représentent les différents domaines de votre salle de TP. Relancer les services serveurs.

Vous pouvez maintenant à partir d'un client, utiliser le serveur smtp d'un autre domaine pour vous en servir comme << agent de relai >> et envoyer des messages aux utilisateurs des autres domaines.

32.3.12. Autres techniques de filtrage et autres services de postfix

Le fichier de configuration `main.cf`, permet de filtrer sur les entêtes de messages (**grep**) sous sur le contenu (body). Ces outils permettent dans certains cas de limiter le spam.

Le serveur postfix.org tient à jour des produits complémentaires qui permettent de mettre en place des antivirus, des outils de filtrage de spam ou des outils de type web−mail.

Chapitre 33. Installation d'un serveur DDNS avec bind et DHCP

Le DNS dynamique

Le DNS dynamique

33.1. Résumé

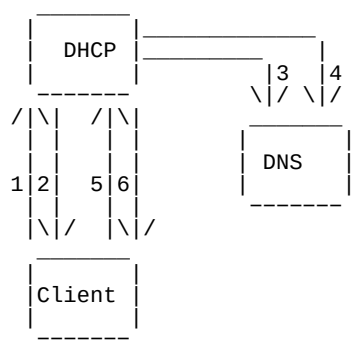
DHCP offre la possibilité de mettre à jour dynamiquement le système de résolution de nom.

Il s'agit, dans cette application, de faire cohabiter et faire fonctionner ensemble le service de résolution de nom bind et le service dhcp.

L'environnement a été testé sur une distribution debian, avec bind9 et dhcp3.

Vous devez savoir configurer un serveur DHCP, un serveur de nom, avoir compris le fonctionnement de rndc et des clés partagées, de dig.

Pour les amateurs d'ASCII−art, voici un schéma qui décrit les processus mis en oeuvre.




```
(1) DHCPDISCOVER from 00:08:c7:25:bf:5a (saturne) via eth0
(2) DHCPOFFER on 192.168.0.195 to 00:08:c7:25:bf:5a (saturne) via eth0
(3) Added new forward map from saturne.freeduc−sup.org to 192.168.0.195
    Ajout de l'enregistrement de type A
(4) added reverse map from 195.0.168.192.in−addr.arpa to saturne.freeduc−sup.org
    Ajout de l'enregistrement de type PTR
(5) DHCPREQUEST for 192.168.0.195 (192.168.0.1) from 00:08:c7:25:bf:5a (saturne) via eth0
(6) DHCPACK on 192.168.0.195 to 00:08:c7:25:bf:5a (saturne) via eth0
```

Les opérations 1, 2, 5, 6 ont déjà été vues lors de l'étude du service DHCP. On voit en étudiant le << log >> ci−dessus, que l'inscription dans le DNS d'un client se fait avant l'acceptation du bail et l'inscription finale de ce client (DHCPACK).

Dans cette application, vous installerez successivement le serveur de nom, le serveur dhcp, puis vous ferez les manipulations qui permettent l'intégration.

33.2. Éléments sur le service DDNS

Tout est décrit dans las pages de man de `dhcpd.conf`.

Deux façons de faire sont décrites (ad−hoc et interim) et une troisième est en cours d'élaboration. La méthode ad−hoc n'est semble t−il plus supportée par les paquets, du moins elle ne l'est pas avec le paquet dhcp3 de debian que j'utilise car considérée comme obsolète.

Le processus utilisé est défini par la variable `ddns−updates−style`. Si la mise à jour n'est pas dynamique, la variable prend la valeur `none`, nous, nous utiliserons `interim`.

La méthode << ad−hoc >> ne prend pas en charge le protocole `failover` des DHCP. C'est à dire qu'avec cette méthode vous ne pourrez pas avoir 2 serveurs DHCP assurant un système redondant et mettant à jour un même ensemble d'enregistrements DNS.

Le serveur détermine le nom du client en regardant d'abord dans les options de configuration des noms (ddns−hostname). Il est possible de générer dynamiquement un nom pour le client en concaténant des chaînes << dyn+N°+NomDeDomaine >>. S'il ne trouve rien, il regarde si le client lui a fait parvenir un nom d'hôte. Si aucun nom n'est obtenu, la mise à jour du DNS n'a pas lieu.

Pour déterminer le nom FQDN, le serveur concatène le nom de domaine au nom d'hôte du client.

Le nom du domaine lui, est défini uniquement sur le serveur DHCP.

Actuellement le processus ne prend pas en charge les clients ayant plusieurs interfaces réseau mais cela est prévu. Le serveur met à jour le DNS avec un enregistrement de type A et un enregistrement de type PTR pour la zone reverse. Nous verrons qu'un enregistrement de type TXT est également généré.

Quand un nouveau bail est alloué, le serveur crée un enregistrement de type TXT qui est une clé MD5 pour le client DHCP (DHCID).

La méthode `interim` est le standard. Le client peut demander au serveur DHCP de mettre à jour le serveur DNS en lui passant ses propres paramètres (nom FQDN). Dans ce cas le serveur est configuré pour honorer ou pas la demande du client. Ceci se fait avec le paramètre `ignore client−updates` ou `allow client−updates`.

Par exemple, si un client `jschmoe.radish.org` demande à être inscrit dans le domaine `exemple.org` et que le serveur DHCP est configuré pour, le serveur ajoutera un enregistrement PTR pour l'adresse IP mais pas d'enregistrement A. Si l'option `ignore client−updates` est configurée, il y aura un enregistrement de type A pour `jschmoe.exemple.org`.

33.3. Les aspects sur la sécurité

Le serveur DNS doit être configuré pour pouvoir être mis à jour par le serveur DHCP. La méthode la plus sûre utilise les signatures TSIG, basées sur une clé partagée comme pour le programme d'administration des serveurs de nom rndc.

Vous devrez en créer une. Pour cela utiliser les éléments fournis dans la partie traitant de bind. Ces aspects y ont déjà été abordés.

Par exemple dans le fichier `named.conf`, le serveur DHCP disposant de la clé `DHCP_UPDATER`, pourra mettre à jour la zone directe et la zone reverse pour lesquelles la déclaration `allow−update` existe.

Description du fichiers `named.conf` :

```
key DHCP_UPDATER {
         algorithm HMAC−MD5.SIG−ALG.REG.INT;
         secret pRP5FapFoJ95JEL06sv4PQ==;
       };

       zone "example.org" {
            type master;
            file "example.org.db";
            allow−update { key DHCP_UPDATER; };
       };

       zone "17.10.10.in−addr.arpa" {
            type master;
            file "10.10.17.db";
            allow−update { key DHCP_UPDATER; };
       };
```

Dans le fichier de configuration du serveur DHCP vous pourrez mettre :

```
 key DHCP_UPDATER {
```

```
        algorithm HMAC-MD5.SIG-ALG.REG.INT;
        secret pRP5FapFoJ95JEL06sv4PQ==;
      };

      zone EXAMPLE.ORG. {
        primary 127.0.0.1; # Adresse du serveur de noms primaire
        key DHCP_UPDATER;
      }

      zone 17.127.10.in-addr.arpa. {
        primary 127.0.0.1; # Adresse du serveur de noms primaire
        key DHCP_UPDATER;
      }
```

La clé `DHCP_UPDATER` déclarée pour une zone dans le fichier `dhcpd.conf` est utilisée pour modifier la zone si la clé correspond dans le fichier `named.conf`.

Les déclarations de zone doivent correspondre aux enregistrement SOA des fichiers de ressources des zones.

Normalement il n'est pas obligatoire d'indiquer l'adresse du serveur de nom primaire, mais cela peut ralentir le processus d'inscription des enregistrements, voire même ne pas fonctionner, si le serveur de nom n'a pas répondu assez vite.

Chapitre 34. Travaux pratiques : DDNS

34.1. Réalisation

Vous allez réaliser l'opération avec un client windows 2000 serveur et un client Linux. Le serveur Linux sera également serveur de nom.

Vous pouvez utiliser les exemples de fichiers fournis. Vosu aurez bien sûr à les adapter à votre configuration. Voici comment vont se dérouler les étapes :

1. Installation du serveur de nom et test
2. Installation du serveur DHCP et test
3. Intégration des deux services

Nous verrons à la fin comment générer des noms dynamiquement pour les clients.

34.2. Les fichiers de configuration

Dans les fichiers il y a des lignes qui sont en commentaires avec ###, elles seront décommentées pour la phase d'intégration des services

34.2.1. Le fichier `named.conf`

```
// Pour journaliser, les fichiers doivent créés
logging {
            channel update_debug {
                 file "/var/log/log-update-debug.log";
                 severity  debug 3;
                 print-category yes;
                 print-severity yes;
                 print-time     yes;
            };
            channel security_info    {
                 file "/var/log/log-named-auth.info";
                 severity  info;
                 print-category yes;
                 print-severity yes;
                 print-time     yes;
            };

            category update { update_debug; };
            category security { security_info; };
      };


// clé partagée entre bind, rndc et dhcp
include "/etc/bind/mykey";

options {
        directory "/var/cache/bind";
        query-source address * port 53;
        auth-nxdomain yes;    # conform to RFC1035
        forwarders { 127.0.0.1; 192.168.0.1;};
};

// Autorisations rndc sur la machine.
controls {
   inet  127.0.0.1   allow {any;} keys {mykey;};
```

```
   inet  192.168.0.0 allow {any;} keys {mykey;};
};


zone "." {
        type hint;
        file "/etc/bind/db.root";
};


zone "localhost" {
        type master;
        file "/etc/bind/db.local";
};

zone "127.in-addr.arpa" {
        type master;
        file "/etc/bind/db.127";
};

zone "0.in-addr.arpa" {
        type master;
        file "/etc/bind/db.0";
};

zone "255.in-addr.arpa" {
        type master;
        file "/etc/bind/db.255";
};

zone "freeduc-sup.org" {
        type master;
        file "/etc/bind/freeduc-sup.org.hosts";
// Sert à la mise à jour par DHCP
// Sera décommenté lors de l'intégration des services
###        allow-update { key mykey; };
        };

zone "0.168.192.in-addr.arpa" {
        type master;
        file "/etc/bind/freeduc-sup.org.hosts.rev";
// Sert à la mise à jour par DHCP
// Sera décommenté lors de l'intégration des services
###        allow-update { key mykey; };
        };
```

34.2.2. Le fichier de zone directe

```
$ORIGIN .
$TTL 86400      ; 1 day
freeduc-sup.org         IN SOA  master.freeduc-sup.org. root.freeduc-sup.org. (
                                2004050103 ; serial
                                10800      ; refresh (3 hours)
                                3600       ; retry (1 hour)
                                604800     ; expire (1 week)
                                38400      ; minimum (10 hours 40 minutes)
                                )
                        NS      master.freeduc-sup.org.
                        MX      10 master.freeduc-sup.org.
$ORIGIN freeduc-sup.org.
master                  A       192.168.0.1
www                     CNAME   master
```

34.2.3. Le fichier de zone `in-addr.arpa`

```
$ORIGIN .
$TTL 86400      ; 1 day
0.168.192.in-addr.arpa  IN SOA  master.freeduc-sup.org. root.freeduc-sup.org. (
                                2004050103 ; serial
                                10800      ; refresh (3 hours)
                                3600       ; retry (1 hour)
                                604800     ; expire (1 week)
                                38400      ; minimum (10 hours 40 minutes)
                                )
                        NS      master.freeduc-sup.org.
$ORIGIN 0.168.192.in-addr.arpa.
1                       PTR     master.freeduc-sup.org.
```

34.2.4. Le fichier `rndc.conf`

```
include "/etc/bind/mykey";
options {
        default-server  localhost;
        default-key     "mykey";
};
```

```
server localhost {
        key     "mykey";
};
```

34.2.5. Le fichier de clé partagée

Ici il est nommé `mykey`.

```
key "mykey" {
        algorithm       hmac-md5;
        secret "X/ErbPNOiXuC8MIgTX6iRcaq/1OFCEDIlxrmnfPgdqYIOY3U6lsgDMq15jnxXEXmdGvv1g/ayYtAA73bUQvWBw==";
};
```

34.2.6. Le fichier dhcpd.conf

```
ddns-update-style none;
### ddns-update-style interim;
###           deny client-updates;
###           ddns-updates on;
###           ddns-domainname "freeduc-sup.org";
###           ddns-rev-domainname "in-addr.arpa";
authoritative;

subnet 192.168.0.0 netmask 255.255.255.0 {
option broadcast-address  192.168.0.255;
option routers 192.168.0.2;
option domain-name "freeduc-sup.org";
option domain-name-servers 192.168.0.1;
option broadcast-address  192.168.0.255;
option routers 192.168.0.2;
range 192.168.0.100 192.168.0.195;
default-lease-time 600;
max-lease-time 7200;

# Instructions pour la mise à jour des zones
### include "/etc/bind/mykey";

### zone freeduc-sup.org. {
###         primary 192.168.0.1;
###         key mykey;
###         }

### zone 0.168.192.in-addr.arpa. {
###         primary 192.168.0.1;
###         key mykey;
###         }

}
```

34.3. Procédure de tests des services

Vous allez pouvoir tester. À partir de maintenant vous devrez consulter les fichiers de logs si vous rencontrez des problèmes de fonctionnement, les tables de processus... bref tout ce qui pourra vous permettre de déterminer la ou les sources possibles des dysfonctionnements si vous en constatez.

Lancez le service bind et tester son fonctionnement avec rndc.

```
root@master:/home/knoppix# rndc status
number of zones: 8
debug level: 0
xfers running: 0
xfers deferred: 0
soa queries in progress: 0
query logging is OFF
server is up and running
root@master:/home/knoppix#
```

Ça permet de vérifier que la clé est bien reconnue.

Vérifiez le fonctionnement du serveur à l'aide de la commande **dig**.

```
root@master:/home/knoppix/tmp# dig @127.0.0.1 freeduc-sup.org axfr
; <<>> DiG 9.2.2 <<>> @127.0.0.1 freeduc-sup.org axfr
;; global options:  printcmd
freeduc-sup.org.        86400   IN      SOA     master.freeduc-sup.org. root.freeduc-sup.org. 2004050107 10800
freeduc-sup.org.        86400   IN      NS      master.freeduc-sup.org.
freeduc-sup.org.        86400   IN      MX      10 master.freeduc-sup.org.
argo.freeduc-sup.org.   86400   IN      A       192.168.0.253
master.freeduc-sup.org. 86400   IN      A       192.168.0.1
www.freeduc-sup.org.    86400   IN      CNAME   master.freeduc-sup.org.
freeduc-sup.org.        86400   IN      SOA     master.freeduc-sup.org. root.freeduc-sup.org. 2004050107 10800
;; Query time: 36 msec
```

```
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Tue May  6 19:15:38 2003
;; XFR size: 8 records
```

Vérifier de la même façon le fonctionnement de la zone reverse.

Vérifiez la strucure du fichier `dhcp`.

```
root@master:/home/knoppix# dhcpd3 −t
Internet Software Consortium DHCP Server V3.0.1rc9
Copyright 1995−2001 Internet Software Consortium.
All rights reserved.
For info, please visit http://www.isc.org/products/DHCP
```

Ça permet de vérifier qu'il n'y a pas d'erreur de syntaxe dans le fichier.

Testez le fonctionnement traditionnel de votre serveur DHCP à partir d'un client Linux et windows. Faites des renouvellement de baux.

34.4. Intégration des services

Par défaut les client Linux ne transmettent pas leur nom d'hôte comme c'est le cas pour les clients windows. Modifiez sur le client Linux le fichier `/etc/dhclient.conf` de la façon suivante, nous verrons plus loin comment générer un nom dynamiquement :

```
[root@bestof mlx]# more /etc/dhclient.conf
           send host−name "bestof";
```

Décommentez dans les fichiers `named.conf` et `dhcpd.conf` les lignes commentées par ###.

Supprimez dans le **dhcpd.conf** la ligne :

```
ddns−update−style none;
```

Relancez le service DNS et testez sont bon fonctionnement

Vérifiez le fichier `dhcpd.conf` avec la commande **`dhcpd3 −t`**.

Lancez dhcp en mode `foreground` **`dhcpd3 −d`**, voici ce que vous devriez obtenir :

```
root@master:/etc/dhcp3# dhcpd3 −d
Internet Software Consortium DHCP Server V3.0.1rc9
Copyright 1995−2001 Internet Software Consortium.
All rights reserved.
For info, please visit http://www.isc.org/products/DHCP
Wrote 1 leases to leases file.
Listening on LPF/eth0/00:d0:59:82:2b:86/192.168.0.0/24
Sending on   LPF/eth0/00:d0:59:82:2b:86/192.168.0.0/24
Sending on   Socket/fallback/fallback−net
```

Demandez un bail à partir du client windows (ici windows 2000 Server), voici ce qui devrait se passer :

```
DHCPDISCOVER from 00:08:c7:25:bf:5a (saturne) via eth0
DHCPOFFER on 192.168.0.195 to 00:08:c7:25:bf:5a (saturne) via eth0
Added new forward map from saturne.freeduc−sup.org to 192.168.0.195
added reverse map from 195.0.168.192.in−addr.arpa to saturne.freeduc−sup.org
DHCPREQUEST for 192.168.0.195 (192.168.0.1) from 00:08:c7:25:bf:5a (saturne) via eth0
DHCPACK on 192.168.0.195 to 00:08:c7:25:bf:5a (saturne) via eth0
```

Demandez un bail à partir du client Linux, voici ce qui devrait se passer :

```
DHCPDISCOVER from 00:08:c7:25:ca:7c via eth0
DHCPOFFER on 192.168.0.194 to 00:08:c7:25:ca:7c (bestof) via eth0
Added new forward map from bestof.freeduc−sup.org to 192.168.0.194
added reverse map from 194.0.168.192.in−addr.arpa to bestof.freeduc−sup.org
DHCPREQUEST for 192.168.0.194 (192.168.0.1) from 00:08:c7:25:ca:7c (bestof) via eth0
DHCPACK on 192.168.0.194 to 00:08:c7:25:ca:7c (bestof) via eth0
```

Voici le contenu du fichier de journalisation de bind :

```
Log de Bind log−update−debug.log
root@master:/var/log# more log−update−debug.log
May 06 07:49:50.457 update: info: client 192.168.0.1#32846: updating zone 'freeduc−sup.org/IN': adding an RR
May 06 07:49:50.458 update: info: client 192.168.0.1#32846: updating zone 'freeduc−sup.org/IN': adding an RR
May 06 07:49:50.512 update: info: client 192.168.0.1#32846: updating zone '0.168.192.in−addr.arpa/IN': deleting
t
May 06 07:49:50.512 update: info: client 192.168.0.1#32846: updating zone '0.168.192.in−addr.arpa/IN': adding a
May 06 07:50:47.011 update: info: client 192.168.0.1#32846: updating zone 'freeduc−sup.org/IN': adding an RR
May 06 07:50:47.011 update: info: client 192.168.0.1#32846: updating zone 'freeduc−sup.org/IN': adding an RR
May 06 07:50:47.017 update: info: client 192.168.0.1#32846: updating zone '0.168.192.in−addr.arpa/IN': deleting
t
```

```
May 06 07:50:47.017 update: info: client 192.168.0.1#32846: updating zone '0.168.192.in-addr.arpa/IN': adding a
root@master:/var/log#
```

Voici le contenu du fichier de déclaration de zone avec les nouveaux enregistrements

```
root@master:/var/log# dig @127.0.0.1 freeduc-sup.org axfr

; <<>> DiG 9.2.2 <<>> @127.0.0.1 freeduc-sup.org axfr
;; global options:  printcmd
freeduc-sup.org.        86400   IN      SOA     master.freeduc-sup.org. root.freeduc-sup.org. 2004050103 10800
freeduc-sup.org.        86400   IN      NS      master.freeduc-sup.org.
freeduc-sup.org.        86400   IN      MX      10 master.freeduc-sup.org.
argo.freeduc-sup.org.   86400   IN      A       192.168.0.253
bestof.freeduc-sup.org. 300     IN      TXT     "00e31b2921cd30bfad552ca434b61bda02"
bestof.freeduc-sup.org. 300     IN      A       192.168.0.194
master.freeduc-sup.org. 86400   IN      A       192.168.0.1
saturne.freeduc-sup.org. 300    IN      TXT     "310e43cfc20efbe1c96798d48672bc76aa"
saturne.freeduc-sup.org. 300    IN      A       192.168.0.195
www.freeduc-sup.org.    86400   IN      CNAME   master.freeduc-sup.org.
freeduc-sup.org.        86400   IN      SOA     master.freeduc-sup.org. root.freeduc-sup.org. 2004050103 10800
;; Query time: 381 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Tue May  6 07:56:43 2003
;; XFR size: 12 records
```

34.5. Générer un nom dynamiquement pour les clients DHCP

Cela est possible en modifiant le fichier de configuration de DHCP. Vous pourrez retrouver tous les éléments dans la page de manuel.

Par exemple rajoutez dans le fichier la ligne ci-dessous pour adapter le nom à partir de l'adresse MAC du client :

```
#ddns-hostname = binary-to-ascii (16, 8, "-",substring (hardware, 1, 12));
```

Ou celle-ci pour localiser le client :

```
ddns-hostname = concat ("dhcp-a-limoges","-",binary-to-ascii(10,8,"-",leased-address));
```

Avec cette dernière, voici les enregistrements ajoutés :

```
Added new forward map from dhcp-a-limoges-192-168-0-194.freeduc-sup.org to 192.168.0.194
added reverse map from 194.0.168.192.in-addr.arpa to dhcp-a-limoges-192-168-0-194.freeduc-sup.org
DHCPREQUEST for 192.168.0.194 from 00:08:c7:25:ca:7c via eth0
DHCPACK on 192.168.0.194 to 00:08:c7:25:ca:7c (bestof) via eth0
```

Le fichier des incriptions :

```
root@master:/home/knoppix# more /var/lib/dhcp3/dhcpd.leases
lease 192.168.0.194 {
  starts 2 2003/05/06 17:38:38;
  ends 2 2003/05/06 17:48:38;
  binding state active;
  next binding state free;
  hardware ethernet 00:08:c7:25:ca:7c;
  set ddns-rev-name = "194.0.168.192.in-addr.arpa";
  set ddns-txt = "00e31b2921cd30bfad552ca434b61bda02";
  set ddns-fwd-name = "dhcp-192-168-0-194.freeduc-sup.org";
  client-hostname "bestof";
}
```

Les transferts de zones directes et inverses :

```
root@master:/home/knoppix/tmp# dig @127.0.0.1 freeduc-sup.org axfr

; <<>> DiG 9.2.2 <<>> @127.0.0.1 freeduc-sup.org axfr
;; global options:  printcmd
freeduc-sup.org.        86400   IN      SOA     master.freeduc-sup.org. root.freeduc-sup.org. 2004050116 10800
freeduc-sup.org.        86400   IN      NS      master.freeduc-sup.org.
freeduc-sup.org.        86400   IN      MX      10 master.freeduc-sup.org.
0-8-c7-25-ca-7c.freeduc-sup.org. 300 IN TXT     "00e31b2921cd30bfad552ca434b61bda02"
0-8-c7-25-ca-7c.freeduc-sup.org. 300 IN A       192.168.0.194
argo.freeduc-sup.org.   86400   IN      A       192.168.0.253
dhcp-192-168-0-194.freeduc-sup.org. 300 IN TXT  "00e31b2921cd30bfad552ca434b61bda02"
dhcp-192-168-0-194.freeduc-sup.org. 300 IN A    192.168.0.194
dhcp-a-limoges-192-168-0-194.freeduc-sup.org. 300 IN TXT "00e31b2921cd30bfad552ca434b61bda02"
dhcp-a-limoges-192-168-0-194.freeduc-sup.org. 300 IN A 192.168.0.194
master.freeduc-sup.org. 86400   IN      A       192.168.0.1
www.freeduc-sup.org.    86400   IN      CNAME   master.freeduc-sup.org.
freeduc-sup.org.        86400   IN      SOA     master.freeduc-sup.org. root.freeduc-sup.org. 2004050116 10800
;; Query time: 3 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Tue May  6 19:39:08 2003
;; XFR size: 14 records
```

La zone reverse :

```
root@master:/home/knoppix/tmp# dig @127.0.0.1 0.168.92.in-addr.arpa axfr

; <<>> DiG 9.2.2 <<>> @127.0.0.1 0.168.92.in-addr.arpa axfr
;; global options:  printcmd
; Transfer failed.
root@master:/home/knoppix/tmp# dig @127.0.0.1 0.168.192.in-addr.arpa axfr

; <<>> DiG 9.2.2 <<>> @127.0.0.1 0.168.192.in-addr.arpa axfr
;; global options:  printcmd
0.168.192.in-addr.arpa. 86400   IN      SOA     master.freeduc-sup.org. root.freeduc-sup.org. 2004050113 10800
0.168.192.in-addr.arpa. 86400   IN      NS      master.freeduc-sup.org.
1.0.168.192.in-addr.arpa. 86400 IN      PTR     master.freeduc-sup.org.
194.0.168.192.in-addr.arpa. 300 IN      PTR     dhcp-192-168-0-194.freeduc-sup.org.
3.0.168.192.in-addr.arpa. 86400 IN      PTR     argo.freeduc-sup.org.
0.168.192.in-addr.arpa. 86400   IN      SOA     master.freeduc-sup.org. root.freeduc-sup.org. 2004050113 10800
;; Query time: 3 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Tue May  6 19:40:08 2003
;; XFR size: 7 records
```

Chapitre 35. Installation d'un service Web-mail

Agent Utilisateurs de Messagerie basés sur HTTP

Agent Utilisateurs de Messagerie basés sur HTTP

35.1. Présentation

Il est préférable d'avoir réalisé les ateliers sur les serveurs HTTP, SMTP et DNS avant de commencer celui-ci.

Le service Web-mail permet l'utilisation d'un service de messagerie à partir d'un client Web comme mozilla. Cette interface est intéressante, car contrairement à un client pop3 standard qui serait configuré pour rapatrier les courriers sur la machine locale, ceux-ci, vont rester sur le serveur. Vous pouvez les consulter à partir de n'importe quel poste pourvu qu'il dispose d'un navigateur. Vous aurez ensuite tout le loisir de les récupérer avec votre client de messagerie préféré si vous en utilisez un.

Il existe un très grand nombre de serveur Web-mail, et écris dans des langages très différents. Une étude a été réalisée par le CRU http://www.cru.fr/http-mail/, mais parmi les principaux on peut citer IMP écris en PHP et qui s'appuie sur la librairie horde. Il existe aussi OpenWebmail qui lui est écris en Perl. Ces deux produits existent en paquets Debian, on utilisera pour le TP OpenWebmail, mais ces deux outils comportent chacun de nombreuses qualités, le choix devra se faire en fonction du degré d'intégration que vous souhaiterez obtenir avec vos autres applications.

35.2. Architecture générale du service

Figure 35-1. Architecture globale d'un service Web-mail

1. En 1 et 2, le client passe par une phase préalable d'authentification, il faut donc un service correspondant sur le serveur. Cela peut être pris directement en charge par le service Web-mail ou par un service extérieur (pam, ldap par exemple). (Nous utiliserons l'authentification pam).
2. En 3 et 4, le dialogue s'effectue en un navigateur et un serveur HTTP. Le dialogue peut s'effectuer dans un canal SSL ou TLS. Le suivi de session peut être réalisé à l'aide de cookie par exemple. (Nous utiliserons Apache comme serveur HTTP).
3. En A et B, le service Web-mail utilise une base de données pour les boîtes aux lettres des utilisateurs, pour les dossiers (`inbox`, `outbox`, `trash`...) et la conservation des courriers. Certains Web-mail s'appuient sur des bases de type MySQL, PostgreSQL... ou simplement sur une arborescence de répertoires dans le `HOME_DIRECTORY` de l'utilisateur. (Nous n'utiliserons pas de SGBD/R).
4. Sur le serveur, il faut activer un protocole de traitement du courrier (pop3, pop3s, imap, imaps...). Nous utiliserons imap et pop3.
5. Vous aurez également besoin d'un service SMTP pour le traitement des courriers sortants (nous utiliserons postfix) et d'un service de livraison (DUA) (nous utiliserons procmail) pour délivrer les courriers entrants.

35.3. Installation et configuration OpenWebmail

Vous allez installer OpenWebmail mais auparavant il est nécessaire de s'assurer du bon fonctionnement de certains services. (smtp, mail, procmail, apache...)

35.3.1. Préparation de la machine

Suivez la procédure ci−dessous pour préparer la machine.

35.3.1.1. Configuration générale

On considère la configuration suivante, vous adapterez les noms, adresses IP et autres paramètres à votre configuration. Il n'y a pas de DNS.

```
Nom d'hôte : freeduc−sup ($NAME)
Nom FQDN de la machine : freeduc−sup.foo.org ($FQDN)
Adresse de réseau : 192.168.0.0
Adresse de la machine : 192.168.0.2
Adresse de la passerelle par défaut : 192.168.0.254
```

35.3.1.2. Test de la résolution de nom

Vérifier que la résolution de nom fonctionne parfaitement. Les commandes : **ping $NAME** et **ping $FQDN** doivent répondre correctement.

```
# Exemple de fichier /etc/hosts
127.0.0.1       freeduc−sup freeduc−sup.foo.org localhost localhost.localdomain
192.168.0.2    freeduc−sup  freeduc−sup.foo.or
```

35.3.1.3. Le service Apache

Activez le service apache. Il ne doit pas y avoir de message d'erreur au lancement, notamment sur la résolution de nom. Vérifiez également le bon fonctionnement avec une requête sur : http://localhost

35.3.1.4. Le service SMTP (Postfix)

Pour configurer postfix, utilisez la commande **dpkg−reconfigure *postfix***, vous prendrez site internet. Les valeurs par défaut doivent normalement fonctionner.

Ouvrez le fichier `/etc/postfix/main.cf`, vérifiez qu'il correspond à celui−ci, au besoin modifiez le :

```
# Fichier de configuration de Postfix
# Adaptez vos noms d'hôtes et vos noms de machines

# see /usr/share/postfix/main.cf.dist for a commented, fuller
# version of this file.

# Do not change these directory settings − they are critical to Postfix
# operation.
command_directory = /usr/sbin
daemon_directory = /usr/lib/postfix
program_directory = /usr/lib/postfix

smtpd_banner = $myhostname ESMTP $mail_name (Debian/GNU)
setgid_group = postdrop
biff = no

myhostname = freeduc−sup.foo.org
mydomain = foo.org
myorigin = $myhostname
inet_interfaces = all
alias_maps = hash:/etc/aliases
alias_database = hash:/etc/aliases
mydestination = $myhostname, localhost.$mydomain
relayhost =
mailbox_command = procmail −a "$EXTENSION"
mailbox_size_limit = 0
recipient_delimiter = +
```

Une fois cela réalisé, activez ou relancez le service.

```
/etc/init.d/postfix start
/etc/init.d/postfix reload
```

35.3.1.5. Activation des services imap

Cela s'effectue dans le fichier `inet.conf`. Il faudra adapter si vous utilisez xinetd. Cela dépend de la distribution de GNU/Linux que vous utilisez. Vous pouvez également utiliser la commande **dpkg−reconfigure *uw−imapd***. Dans ce cas, prenez imap2 (qui correspond à imap4 (? ;−))) et imaps.

Extrait d'un exemple de configuration de `inetd.conf` :

```
#:MAIL: Mail, news and uucp services.
imap2   stream  tcp     nowait  root    /usr/sbin/tcpd /usr/sbin/imapd
```

Adaptez votre fichier de configuration, puis relancer inetd avec la commande :

```
/etc/init.d/inetd restart
```

Vérifiez que les ports sont bien ouverts avec la commande **netstat**. Vous devez avoir les ports 25 (pop3)si vous l'avez activé et 143 (imap) ouverts.

```
netstat -atup | grep LISTEN
tcp     0      0 *:netbios-ssn       *:*      LISTEN      269/smbd
tcp     0      0 *:imap2             *:*      LISTEN      263/inetd
tcp     0      0 *:sunrpc            *:*      LISTEN      151/portmap
tcp     0      0 *:ssh               *:*      LISTEN      278/sshd
tcp     0      0 *:ipp               *:*      LISTEN      290/cupsd
tcp     0      0 *:smtp              *:*      LISTEN      899/master
```

Nous voyons imap2, ligne 2, pris en charge par inetd.

```
root@freeduc-sup:/home/mlx# netstat -natup | grep LISTEN
tcp   0      0 0.0.0.0:139   0.0.0.0:*       LISTEN      269/smbd
tcp   0      0 0.0.0.0:143   0.0.0.0:*       LISTEN      263/inetd
tcp   0      0 0.0.0.0:111   0.0.0.0:*       LISTEN      151/portmap
tcp   0      0 0.0.0.0:22    0.0.0.0:*       LISTEN      278/sshd
tcp   0      0 0.0.0.0:631   0.0.0.0:*       LISTEN      290/cupsd
tcp   0      0 0.0.0.0:25    0.0.0.0:*       LISTEN      899/master
```

Ici l'option `-n` de **netstat** nous indique les numéros de ports utilisés.

Remarque : si vous souhaitez utiliser un client pop, vous devrez activer également le protocole pop.

35.3.1.6. Test des services

À ce stade, il nous est possible de tester complètement le service de messagerie. Vous pouvez indifféremment utiliser un client comme kmail ou Mozilla. Le plus simple est d'utiliser le client mail.

Suivez la procédure ci-dessous :

1. Créez deux comptes utilisateurs alpha et beta qui serviront pour les tests avec la commande **adduser**.
2. Testez avec la commande **mail** que l'envoie de courrier se déroule correctement. Les commandes :

   ```
   mail alpha
   mail alpha@freeeduc-sup
   mail alpha@freeduc-sup.foo.org
   ```

   doivent fonctionner correctement.

35.3.2. Installation d'OpenWebmail

Si vous utilisez la freeduc-sup, OpenWebmail n'est pas installé. Utilisez la commande :

```
apt-get install openwebmail
```

La commande installera également 3 paquets supplémentaires qui correspondent aux dépendances puis lancera la procédure de configuration.

35.3.3. Configuration de l'application OpenWebmail

Vous pouvez à tout moment reconfigurer l'application avec **`dpkg-reconfigure openwebmail`**. Prenez comme option :

```
authentification -> auth_pam.pl
langage -> fr
```

35.3.4. Test de l'environnement

Pour tester l'environnement vous avez deux liens :

```
http://localhost/openwebmail
```

qui vous place sur un espace documentaire

```
http://localhost/cgi-bin/openwebmail/openwebmail.pl
```

qui lance l'application proprement dite et vous amène sur la première fenêtre de login

Figure 35-2. Ouverture de session sur un Web-mail

Il vous sera possible de créer un serveur Web virtuel pour avoir par exemple : << openwebmail.freeduc−sup.org >>.

35.3.5. Configuration de l'environnement utilisateur

À la première session, la personne reçoit une invite lui permettant de configurer son environnement et ses paramètres particuliers, comme son adresse de réponse, modifier son mot de passe... Ces paramètres sont modifiables à tout moment.

Figure 35−3. Configuration de l'environnement utilisateur

35.3.6. Test et environnement OpenWebmail

À partir de ce moment, l'environnement complet est disponible. L'utilisateur dispose également d'un calendrier et d'une documentation en ligne.

Figure 35−4. Voir ses messages

Figure 35−5. Le calendrier

Figure 35−6. L'aide en ligne

35.4. Application

1. Configurez et vérifiez le bon fonctionnement des services de résolution de nom, apache, postfix.
2. Installez et configurez OpenWebmail.
3. Testez le fonctionnement OpenWebmail.

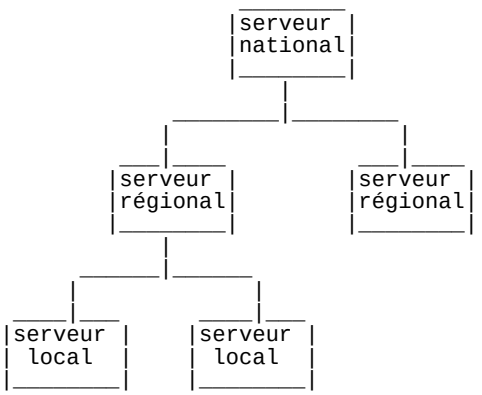
Chapitre 36. Installation d'un service mandataire (Proxy SQUID)

Serveur mandataire et serveur de cache − Fiche de cours

Serveur mandataire et serveur de cache − Fiche de cours

Squid est un service serveur proxy−cache sous linux. Les objets consultés par les clients sur internet, sont stockés en cache disque par le serveur. À partir du deuxième accès, la lecture se fera en cache, au lieu d'être réalisée sur le serveur d'origine. De ce fait il permet << d'accélérer >> vos connexions à l'internet en plaçant en cache les documents les plus consultés. On peut aussi utiliser la technique du service serveur mandataire pour effectuer des contrôles d'accès aux sites.

Les services proxy peuvent être organisés de façon hiérarchique :

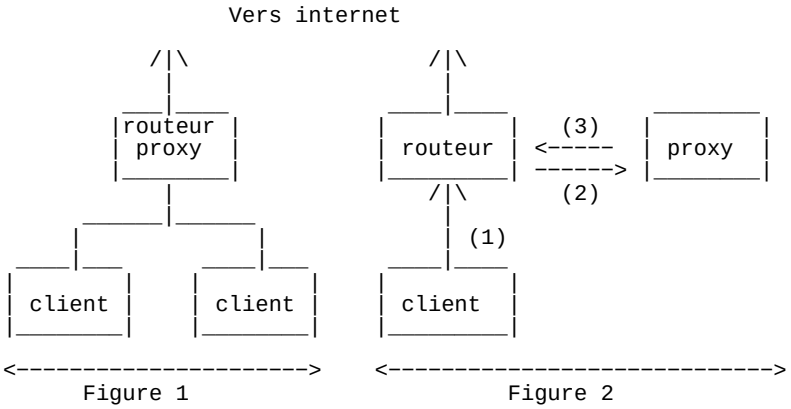


Ls serveurs peuvent être paramétrés pour les autorisations d'accès et les synchronisations.

Les postes clients sont souvent configurés pour utiliser un serveur proxy. Le client s'adresse au serveur proxy, et c'est ce dernier qui traite la requête sur internet. Un fois la réponse reçue, le serveur met en cache la réponse et la retourne au client interne. Le service proxy est fréquemment configuré sur un routeur qui remplit aussi le service de translation d'adresse ou translation de port, mais toutes ces fonctions sont bien différentes.

Dans certains cas, on peut ne pas souhaiter que la configuration soit réalisée au niveau du client. On souhaite que celle−ci soit faite au niveau du serveur. Cela peut arriver par exemple si vous avez plusieurs centaines de postes à configurer ou bien si vous ne souhaitez pas que les utilisateurs puissent modifier ou avoir accès à cette partie de la configuration. On parlera de << service proxy transparent >>>. Le service serveur proxy peut être sur le routeur d'accès à l'internet ou sur une autre machine.

```
Service proxy tranparent :
La configuration des navigateurs, sur les postes clients, n'est pas concernée.
```



Sur la Figure 1, le service proxy est installé sur le routeur.

Sur la figure 2, les requêtes du client (1), sont redirigées vers le proxy par le routeur (2), qui retourne au client la réponse ou redirige vers le routeur (3) pour un envoie sur l'extérieur.

36.1. Installer Squid

Sur debian **apt−get `install` *`squid`***.

Squid comporte de très nombreux paramètres. L'optimisation n'en est pas toujours simple. Nous allons voir uniquement quelques options permettant un fonctionnement du service. Il sera nécessaire, pour un site en production, de se référer à la documentation officielle.

Pour démarrer une configuration simple, il est possible d'utiliser le fichier de configuration `/etc/squid.conf`, dont chaque paramètre est documenté.

36.2. Configuration de squid

Toute la configuration de Squid se trouve dans le fichier `squid.conf`. La plupart des options par défaut du fichier ne sont pas à changer (vous pouvez alors laisser le # pour conserver les options en commentaire.)

`http_port` : le port que vous souhaitez utiliser. Le plus fréquent est 8080. Il faut donc changer cette valeur car par défaut Squid utilise 3128.

`icp_port` : conserver le port 3130. Ceci vous permet de communiquer avec des proxy−cache parents ou voisins.

`cache_mem` : correspond au cache mémoire, la valeur dépend de votre système. Par défaut squid utilise 8 Mo. Cette taille doit être la plus grande possible afin d'améliorer les performances (Considérez 1/3 de la mémoire que vous réservez à Squid). Il faut avec cache_mem régler cache_mem_low et cache_mem_high qui sont les valeurs limites de remplissage du cache mémoire. Par défaut les valeurs sont 75 % et 90 %. Lorsque la valeur de 90 % est atteinte le cache mémoire se vide jusqu'à 75 %. Les valeurs par défaut sont correctes dans la plupart des cas.

`cache_swap` : correspond à la taille de votre cache disque. Si la taille du disque le permet, et en fonction de la taille de votre établissement (nombre de client qui utilise le cache), mais aussi de la durée de rafraîchissement de votre cache et du débit de votre ligne, vous devez mettre la valeur qui vous semble correspondre à votre situation.

`acl QUERY urlpath_regex cgi−bin \? \.cgi \.pl \.php3 \.asp` : Type de page à ne pas garder dans le cache afin de pas avoir les données d'un formulaire par exemple.

`maximum_object_size` : taille maximale de l'objet qui sera sauvegardé sur le disque. On peut garder la valeur par défaut.

`cache_dir` : Vous indiquez ici le volume de votre cache. Si vous avez plusieurs disques utilisez plusieurs fois cette ligne. Si squid ne fonctionne pas bien, où s'arrête parfois sans raison apparente, vérifiez que vous avez un cache assez important ou bien configuré.

```
cache_dir ufs /cache1 100 16 256         (cache de 100 Mb)
cache_dir ufs /cache2 200 16 256         (cache de 200 Mb)
```

Les valeurs 16 et 256, indiquent le nombre de sous−répertoires créés respectivement dans le premier niveau et suivants pour le stockage des données du cache.

`cache_access_log` ; `cache_log` ; `cache_store_log` : Indique l'endroit ou se trouve les logs (fichiers de journalisation. Si vous ne souhaitez pas avoir de log (par exemple des objets `cache_store_log`) indiquer `cache_store_log none`.

`debug_options ALL,1` : niveau de debug. Indiquer 9 pour avoir toutes les traces à la place de 1. Attention cela donne de gros fichiers.

`dns_childrens` : par défaut le nombre de processus simultanés dns est de 5. Il peut être nécessaire d'augmenter ce nombre afin que Squid ne se trouve pas bloqué. Attention de ne pas trop l'augmenter cela pouvant poser des problèmes de performance à votre machine (indiquer 10 ou 15).

`request_size` : taille maximale des requêtes. Conserver le défaut, concerne les requêtes de type GET, POST...

`refresh_pattern` : permet de configurer la durée de mise à jour du cache. Utiliser `−i` pour ne pas tenir compte des minuscules ou des majuscules. (voir le fichier `squid.conf`). Les valeurs Min et Max sont indiquées en minutes. Exemple :

```
# refresh_pattern ^ftp:         1440    20%     10080
```

`visible_hostname` : indiquer ici le nom de votre serveur proxy.

`logfile_rotate` : pour faire tourner vos logs et garder un nombre de copies. par défaut 10. attention si votre cache est trés utilisé il peut générer un grand volume de logs, pensez donc à réduire ce nombre.

`error_directory` : Pour avoir les messages d'erreurs en français (indiquer le répertoire ou ils se trouvent). Exemple :

```
#error_directory /etc/squid/errors
#Créer un lien vers le répertoire où sont logés les messages en Français.
```

36.3. Initialisation de Squid

Cela n'est réalisé que la première fois afin de générer le cache.

```
squid −z
```

36.4. Les options de démarrage de squid

On peut aussi démarrer squid en lui passant des commandes sur la ligne de commande. Différents paramètres peuvent être passés sur la ligne de commande. Les options passées de cette façon remplacent les paramètres du fichier de configuration de Squid `squid.conf`.

```
−h : Pour obtenir les options possibles
−a : Pour indiquer un port particulier
−f : pour utiliser un autre fichier de conf  au lieu de squid.conf
−u : spécifie un port pour les requêtes ICP. (3110 par défaut)
−v : pour indiquer la version de Squid
−z : Pour initialiser le disque cache.
−k : Pour envoyer des instructions à Squid pendant son fonctionnement.
     Il faut faire suivre −k d'une instruction
     (rotate|reconfigure|shutdown|interrupt|kill|debug|check).
−D : pour démarrer squid lorsque vous n'êtes pas connecté en
     permanence à internet (évite de vérifier si le serveur DNS répond).
```

36.5. Contrôler les accès

Pour contrôler tout ce qui passe par votre serveur proxy, vous pouvez utiliser ce que l'on appelle les ACL (*Access Control List*). Les ACL sont des règles que le serveur applique. Cela permet par exemple d'autoriser ou d'interdire certaines transactions.

On peut autoriser ou interdire en fonction du domaine, du protocole, de l'adresse IP, du numéro de port, d'un mot, on peut aussi limiter sur des plages horaires.

La syntaxe d'une ACL est la suivante :

```
acl             aclname          acltype            string[string2]
http_access     allow|deny       [!]aclname
```

`acltype` peut prendre comme valeur :

```
src (pour la source) : indication de l'adresse IP du client sous la
    forme adresse/masque. On peut aussi donner une plage d'adresse
    sous la forme adresse_IP_debut-adresse_IP_fin
dst (pour la destination) : idem que pour src, mais on vise
    l'adresse IP de l'ordinateur cible.
srcdomain : Le domaine du client
dstdomain : Le domaine de destination.
url_regex : Une chaîne contenu dans l'URL
   (on peut utiliser les jokers ou un fichier).
urlpath_regex : Une chaîne comparée avec le chemin de l'URL
   (on peut utiliser les jokers).
proto : Pour le protocole.
```

Exemple 1 : Interdire l'accés à un domaine : supposons que nous souhaitions interdire l'accés à un domaine (par exemple le domaine pas_beau.fr). On a donc

```
acl           veuxpas         dstdomain         pas_beau.fr
http_access   deny            veuxpas
http_access   allow           all     # On accepte tout
```

La dernière ligne ne doit exister qu'une fois dans le fichier `squid.conf`.

Exemple 2 : interdire l'accés aux pages contenant le mot jeu.

```
acl           jeu            url_regex            jeu
http_access   deny            jeu
http_access   allow           all
```

Attention `url_regex` est sensible aux majuscules/minuscules. Pour interdire `JEU`, il faut aussi ajouter `JEU` dans votre ACL. Il n'est pas besoin de réécrire toute l'ACL. On peut ajouter `JEU` derrière jeu en laissant un blanc comme séparation (cela correspondant à l'opérateur logique OU).

On peut placer un nom de fichier à la place d'une série de mots ou d'adresses, pour cela donner le nom de fichier entre guillemets. Chaque ligne de ce fichier doit contenir une entrée.

Exemple 3 : utilisation d'un fichier

```
# URL interdites
acl          url_interdites  url_regex "/etc/squid/denied_url"
http_access  deny            url_interdites
```

Des produits associés à Squid (redirecteurs) permettent un contrôle plus simple. SquidGuard, par exemple, permet d'interdire des milliers de sites. Le site d'information est référencé plus loin dans la rubrique << liens >>. Pensez, si vous utilisez SquidGuard, à configurer la ligne suivante dans le fichier `squid.conf` :

```
redirect_program /usr/local/squid/bin/SquidGuard
```

Exemple 4 : pour contrôler qui a le droit d'utiliser votre cache, créez une ACL du type :

```
acl             si_OK        src    192.168.0.0/255.255.0.0
http_access     allow        localhost
http_access     allow        site_OK
http_access     deny         all
```

36.6. Contrôler les accés par authentification

Parmi les demandes qui reviennent le plus souvent, la question de l'utilisation de Squid pour contrôler qui a le droit d'aller sur internet, est l'une des plus fréquente.

On peut imaginer deux solutions :

La première consiste à contrôler les accés par salle et par horaires, en fonction d'un plan d'adressage de votre établissement. Le travail de l'académie de Grenoble avec ls projet SLIS permet de faire cela. On l'administre avec une interface Web. Ce n'est alors pas Squid qui est utilisé pour cela mais les fonctions de filtrage du routeur (netfilter par exemple). Construire des ACL directement dans Squid est faisable, mais cela n'est pas toujours simple à mettre en oeuvre.

La deuxième solution est de contrôler en fonction des individus. Squid permet de faire cela, à partir de plusieurs façons (APM, LDAP, NCSA auth, SMB...). Les différentes techniques sont décrites dans la FAQ de Squid sur le site officiel. *Squid*

Si vous utilisez un annuaire LDAP, vous devez avoir dans le fichier `squid.conf` les lignes suivantes :

```
acl             identification   proxy_auth        REQUIRED
http_access     allow            identification
authentificate_program /usr/lib/squid/squid_ldap_auth \
```

```
                                        -b $LDAP_USER -u uid SERVEUR_LDAP
LDAP_USER est l'ou dans laquelle se trouve les clients
(par exemple ou=people, ou= ac-limoges, ou=education, ou=gouv, c=fr).
```

Si vous n'avez pas de serveur LDAP, une méthode simple à mettre en oeuvre, consiste à utiliser une méthode similaire ux fichier `.htaccess` d'Apache.

Exemple de configuration avec NCSA_auth

```
authenticate_program /usr/lib/ncsa_auth /etc/squid/passwd
acl                  foo                proxy_auth REQUIRED
acl                  all                src 0/0
http_access          allow              foo
http_access          deny               all
```

36.7. Interface web de Squid et produits complémentaires

squid dispose en standard de quelques outils, mais sinon vous pouvez utiliser webmin. Vous trouverez également, sur le site officiel de squid, une liste de produits supplémentaires pouvant être interfacés avec Squid.

36.8. La journalisation

Squid journalise les transactions dans un fichier `access.log`. Ce fichier donne les informations sur les requêtes qui ont transité par Squid. Le fichier `cache.log` informe sur l'état du serveur lors de son démarrage. Le fichier `store.log` informe sur les objets stockés dans le cache.

Les dates indiquées dans le fichiers `access.log` indique le temps en secondes depuis le 1 janvier 1970 (format epoch), ce qui n'est pas trés facile à lire. Un petit script en perl, permet de recoder les dates :

```
#! /usr/bin/perl -p
s/^\d+\.\d+/localtime $&/e;
```

36.9. Configurer les clients

Pour configurer les clients, on peut utiliser la configuration manuelle ou la configuration automatique avec des fichiers `.pac` ou des fichiers `.reg` que l'on place dans le script de connexion des clients.

Configuration manuelle des clients

Configuration automatique

36.10. Forcer le passage par Squid (Proxy transparent)

Il existe plusieurs solutions :

Configurer votre navigateur avec le bon proxy ou en utilisant le fichier de configuration automatique et le rendre impossible à changer. Mais cela nécessite que vous contrôliez les clients ce qui n'est pas toujours le cas.

Intercepter les requêtes sur le port 80 du routeur pour les rediriger sur Squid.

Vous devez alors avoir dans votre fichier `squid.conf` :

```
# Configuration de traitement des requêtes du client
    httpd_accel_host virtual
    httpd_accel_port 80
    httpd_accel_with_proxy on
    httpd_accel_uses_host_header on
    httpd_accel_single_host off
```

Puis ajouter la règle pour netfilter de redirection des requêtes sur le port 80

```
iptables -t nat -F PREROUTING

iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 80 \
                                              -j REDIRECT --to-port 8080
# Les clients peuvent envoyer leurs requêtes sur le port 80 du proxy
# Le service NAT du routeur les redirige sur le port 8080
```

36.11. Le redirecteur SquidGuard

Squid dispose d'une fonctionalité qui permet de passer une URL (requête entrante) à une applications externe. Cela présente l'avantage de pouvoir bénéficier des services d'applications spécialisées. C'est par exemple le cas pour le redirecteur SquidGuard, largement utilisé pour protéger les accès sur des sites déclarés comme << impropres >>. Une base de données de ces sites est tenue à jour. C'est cette dernière qui ets utilisée pour filtrer les accès.

36.12. Les applications non prises en charge par un service proxy

Certaines applications ne sont pas prises en charge par Squid (https, smtp, pop, ftp...). Les raisons peuvent être diverses. Soit le service n'est pas pris en charge (pop, smtp...), soit il n'est pas conseillé de stocker en cache certaines informations d'authentification par exemple (https).

Pour les applications ou services nons pris en charge par un service proxy, vous devrez utiliser l'ipmasquerade, un service de translation d'adresse ou utiliser une autre technologie.

Chapitre 37. Travaux pratiques : installation de SQUID

37.1. Application

Vous devez maîtriser les techniques de routage avec netfilter.

Vous allez installer un service proxy minimal, configurer les clients puis tester le fonctionnement de l'accès à internet à partir des clients.

Vous configurerez des ACLs permettant un contrôle d'accès aux données externes, vous ferez ensuite évoluer cette configuration vers un service mandataire transparent.

Le service proxy sera installé sur le routeur.

Utilisez les éléments de ce document, ainsi que les exemples de fichiers de configuration donnés en annexe. Vous pourrez également vous référer au document sur netfilter.

37.1.1. Préparation de la maquette

Vous avez un routeur qui vous relie au réseau de l'établissement et un client qui représente un segment de réseau privé. L'ensemble doit fonctionner (accès à internet, résolution de nom, masquage d'adresse.

```
iptables −t nat −A POSTROUTING −s 192.168.0.0/24 −j MASQUERADE
# si 192.168.0.0 est le réseau privé
```

Vérifier que le routeur fonctionne. Faites un test à partir du client. Supprimez au besoin toutes les règles iptables et activez l'ipmasquerade.

Mettez une règle qui interdise toute requête à destination d'une application HTTP (port 80). Vérifier que les clients ne peuvent plus sortir.

```
# Ici on bloque tout, c'est brutal, mais on va faire avec.
iptables −P FORWARD DROP
```

37.1.2. Installation et configuration du service proxy

Faites une sauvegarde de votre fichier de configuration original (`/etc/squid.conf`). Modifiez le fichier de configuration de squid en vous appuyant sur celui donné ci−dessous.

```
http_port 3128
#We recommend you to use the following two lines.
acl QUERY urlpath_regex cgi−bin \?
no_cache deny QUERY
cache_mem 8 MBM

maximum_object_size_in_memory 8 KB

cache_dir ufs /var/spool/squid 100 16 256

cache_access_log /var/log/squid/access.log
cache_log /var/log/squid/cache.log
cache_store_log /var/log/squid/store.log

# Put your FQDN here
visible_hostname freeduc−sup.foo.org

pid_filename /var/run/squid.pid

#Recommended minimum configuration:
acl all src 0.0.0.0/0.0.0.0
acl manager proto cache_object
acl localhost src 127.0.0.1/255.255.255.255

#Recommended minimum configuration:
http_access allow manager localhost
http_access allow all
```

Initialisez l'espace disque pour le cache.

```
# Vérifiez que le FQDN de votre serveur est renseigné
# et que la résolution de nom locale fonctionne (fichier hosts ou DNS).
# initialisation de la zone de cache
squid −z
# Lancement de squid
/etc/init.d/squid start | restart
```

Démarrer et vérifier le bon fonctionnement de Squid. Consultez également les journaux.

```
$>ps aux |  grep squid
root      2984  0.0  0.4  4048 1124 ?  S  15:22   0:00 \
                                          /usr/sbin/squid −D −sYC
proxy     2987  2.1  1.6  6148 4068 ?  S  15:22   0:00 (squid) −D −sYC
```

Vérifier que le port d'écoute est correct.

```
mlx@uranus:~$ netstat −atup | grep LISTEN
(Tous les processus ne peuvent être identifiés, les infos sur
les processus non possédés ne seront pas affichées, vous devez
être root pour les voir toutes.)

tcp        0      0 *:3128          *:*               LISTEN      −
```

Identifiez l'endroit de stockage du cache sur le disque.

37.1.3. Configuration du client

Configurer le client pour qu'il utilise le service proxy sur les requêtes HTTP, vérifier le bon fonctionnement.

Figure 37−1. Configuration du client

Identifiez les traces dans les journaux.

```
root@uranus:/home/mlx# more /var/log/squid/access.log
1053864320.437   1741 192.168.0.2 TCP_MISS/200 5552 \
   GET http://www.cru.fr/documents/ − DIRECT/195.220.94.166 text/html
1053864320.837   1096 192.168.0.2 TCP_MISS/304 331 \
   GET http://www.cru.fr/styles/default.css − DIRECT/195.220.94.166 −
1053864321.257    420 192.168.0.2 TCP_MISS/304 331 \
   GET http://www.cru.fr/logos/logo−cru−150x53.gif − DIRECT/195.220.94
1053864321.587    696 192.168.0.2 TCP_MISS/304 331 \
   GET http://www.cru.fr/icons−cru/mailto.gif − DIRECT/195.220.94.166
1053864550.537   1461 192.168.0.2 TCP_MISS/200 5552 \
   GET http://www.cru.fr/documents/ − DIRECT/195.220.94.166 text/htm
```

Interdisez tous les accès avec la règle :

```
http_access deny all
```

Vérifiez le fonctionnement.

37.1.4. Mise en place d'une ACL simple

Interdisez l'accès à un serveur (google.fr) par exemple. Vérifiez le fonctionnement.

```
acl google dstdomain .google.fr
http_access deny google
```

37.1.5. Utilisation de fichiers pour stocker les règles des ACL

Construisez deux fichiers, l'un qui permettra de stocker des adresses IP, l'autres des mots clés. Construisez une ACL qui interdit l'accès en sortie aux machines qui ont les adresses IP déterminées dans le premier fichier, et une ACL qui empêche l'accès aux URL qui contiennent les mots clés stockés dans le second fichier.

```
# Exemple de ce que le fichier "adresse_ip" contient :
# Mettez dans laliste des adresse celle de votre client pour tester
192.168.0.2
192.168.0.10

# Exemple de ce que le fichier "mot_cle" contient :
jeu
game

# Exemple d'ACL
acl porn url_regex "/etc/squid/mot_cle"
acl salleTP_PAS_OK src "/etc/squid/adresse_ip"
```

```
http_access deny porn
http_access deny salleTP_PAS_OK
```

Tester le fonctionnement de ces deux ACL. (Utiliser comme url de destination par exemple http://games.yahoo.com/)

37.1.6. Configuration des messages d'erreurs

Configurez Squid pour qu'il affiche des pages (messages d'erreur) en Français. Vérifiez le fonctionnement.

```
error_directory /usr/share/squid/errors/French
```

Identifiez la page qui est retournée lors d'un refus d'accès. Modifiez la page et le message retourné, puis vérifiez le fonctionnement.

37.1.7. Automatisation de la configuration des clients.

Créez un fichier `.pac` pour la configuration des clients Mozilla. Vous en avez un complet dans la FAQ de squid. Celui−ci fait le minimum.

```
function FindProxyForURL(url, host)
  {
   return "PROXY 192.168.0.1:3128; DIRECT";
  }
```

Mettez le fichier sur votre routeur dans `/var/www/mozilla.pac` et vérifiez que le serveur apache est bien démarré. Si la résolution de nom fonctionne, vous pouvez mettre le nom du serveur de configuration plutôt que l'adresse IP.

Configurez le client :

Figure 37−2. Configuration du client

Testez le bon fonctionnement du client.

Remettez la configuration du client dans sa situation initiale.

37.1.8. Installation et configuration du service proxy Squid transparent.

Modifier la configuration du client et du serveur, afin que la configuration globale devienne celle d'un proxy tranparent.

Vous allez modifier le fichier de configuration de squid et configurer votre routeur avec les règles suivantes si le service proxy est sur le routeur :

```
# A mettre dans le fichier de configuration de suid
# Relancer le service après
httpd_accel_host virtual
httpd_accel_port 80
httpd_accel_with_proxy  on
httpd_accel_uses_host_header on

# Les règles iptables
# On nettoie la table nat
# On utilise le port 3128, par utilisé par défaut sous Squid.
iptables −t nat −F PREROUTING

# ou toutes les tables nat si besoin
iptables −t nat −F

# On laisse passer (masque) les requêtes autres que sur le port 80
 iptables −t nat −A POSTROUTING −j MASQUERADE

# On redirige les requêtes sur le port 80
iptables −t nat −A PREROUTING −j DNAT −i eth1 −p TCP −−dport 80 \
                                  −−to−destination 192.168.0.1:3128
```

Supprimer toute configuration de proxy sur le client. Vérifier le bon fonctionnement du client.

Arrêtez les service proxy, vérifiez que les requêtes HTTP des clients ne sortent plus.

Il est possible de séparer les services du routage et proxy sur 2 machines différentes. Le principe est identique, seules les règles sur le routeur changent un peu. Vous trouverez la description d'une telle configuration dans :

```
# Transparent proxy with Linux and Squid mini HOWTO
http://www.tldp.org/HOWTO/mini/TransparentProxy.html

# Lire aussi sur netfilter
http://www.netfilter.org/documentation/HOWTO/fr/NAT−HOWTO.txt
http://www.cgsecurity.org/Articles/netfilter.html
```

37.1.9. Mise en place de l'authentification

Mettez en place une ACL pour déclarer l'authentification des personnes.

```
# Ici on utilise le module ncsa_auth
auth_param basic program /usr/lib/squid/ncsa_auth /etc/squid/users
auth_param basic realm Squid proxy-caching web serve
auth_param basic children 5
acl foo proxy_auth REQUIRED
http_access allow foo
```

Créez les fichiers de compte et de mots de passe avec un compte utilisateur.

```
 htpasswd -c /etc/squid/users unUTILISATEUR
# mettez ensuite son mot de passe.

# Testez le fonctionnement du fichier et du module
# Vous passez en paramètre le nom du fichier de comptes
# Vous mettez le compte et le mot de passe, le module retourne OK
# En cas d'erreur il retourne ERR
root@uranus:/etc# /usr/lib/squid/ncsa_auth /etc/squid/users
mlx password
OK
mlx mauvais
ERR
```

Figure 37-3. Authentification SQUID

Il y a pas mal de différences entre les paramètres des versions de Squid 1, squid 2 et Squid 2.5. Il est important de consulter les fichiers de documentation fournis avec le produit.

L'authentification ne fonctionne pas avec la configuration d'un proxy transparent.

37.2. Liens

1. *Squid*
2. *Le CRU*
3. *SquidGuard - Université de Toulouse*
4. *Les HOWTOs*

37.3. Annexes

37.3.1. Fichier `squid.conf` - testé avec Squid 2.5

Fichier minimal pour Squid

```
http_port 3128

#Ne pas "cacher" les données des formulaires
acl QUERY urlpath_regex cgi-bin \?
no_cache deny QUERY
cache_mem 8 MBM

maximum_object_size_in_memory 8 KB

cache_dir ufs /var/spool/squid 100 16 256

cache_access_log /var/log/squid/access.log
cache_log /var/log/squid/cache.log
cache_store_log /var/log/squid/store.log

# Ici mettez le nom de votre machine
visible_hostname uranus.freeduc-sup.org

pid_filename /var/run/squid.pid

#Recommended minimum configuration:
acl all src 0.0.0.0/0.0.0.0
acl manager proto cache_object
acl localhost src 127.0.0.1/255.255.255.255

# Test des fichiers @ip et mots clés
acl porn url_regex "/etc/squid/mot_cle"
acl salleTP_PAS_OK src "/etc/squid/adresse_ip"
http_access deny porn
http_access deny salleTP_PAS_OK

# Authentification
```

```
auth_param basic program /usr/lib/squid/ncsa_auth /etc/squid/users
auth_param basic realm Squid proxy-caching web serve
auth_param basic children 5
acl foo proxy_auth REQUIRED
http_access allow foo

#Default:
#http_access deny all

#Messages d'erreurs en FR
error_directory /usr/share/squid/errors/French

# Pour le proxy cache transparent
httpd_accel_host virtual
httpd_accel_port 80
httpd_accel_with_proxy  on
httpd_accel_uses_host_header on
```

37.3.2. Exemples d'ACLs Squid 2.2

Utiliser des fichiers externes pour la déclarations d'adresse ou de mots clés.

```
acl salleTP_OK src "/etc/squid/salleTP_OK.txt"
acl porn url_regex "/etc/squid/porn.txt"
acl salleTP_PAS_OK src "/etc/squid/salleTP_PAS_OK.txt"
http access salleTP_OK
http access porn
http deny    salleTP_PAS_OK
```

37.3.3. ACL par authentification Squid 2.2

Utilisation d'une authentification simple similaire à celle mise en oeuvre dans les `.htaccess`. Créer le fichier par script ou manuellement avec **htpasswd**.

```
authenticate_program /usr/bin/ncsa_auth /etc/squid/users
authenticate_children 5
acl_authenticate_users REQUIRED
http_access authenticate_users
```

37.3.4. ACL sur des plages horaires Squid 2.2

Combinaison par << ET >> logique des plage horaire et des salles. Mettre la machine à l'heure avec ntpdate par exemple.

```
# Interdire les accès en dehors des plages horaires 8h-12h et 14h-18h
S Sunday M Monday T Tuesday W Wednesday H Thrusday F Friday A Saturday
acl am time MTWHF 08:00-12:00
acl PM time MTWHF 14:00-18:00
http_access allow am salleTP_PAS_OK
http_access allow pm salleTP_PAS_OK
```

Chapitre 38. Installation d'un serveur PostgreSQL avec Apache

>Serveur WEB dynamique avec PostgreSQL et PHP

>Serveur WEB dynamique avec PostgreSQL et PHP

Création d'un site web dynamique avec PostgreSQL et Apache. Pour PostgreSQL vous pouvez aussi utiliser la ressource Linux-France.org qui détaille de façon assez complète un mode d'utilisation de ce serveur de bases de données.

38.1. Avant de démarrer

Si vous utilisez la Freeduc-Sup rc3, un bogue empêche PostgreSQL de se lancer. Vous devez avoir un message d'erreur dans "/var/log/postgres" qui vous indique qu'il n'arrive pas à trouver un fichier "pg_control".

Voici comment corriger cela : sous le compte root taper

```
mv /var/lib/postgres/data /var/lig/postgres/data.old
dpkg-reconfigure postgresql
OK
OK
```

```
Yes
OK
fr_FR@euro
OK
LATIN1
OK
ISO
European
OK
NO
#C'est terminé, vous pourrez lancer PostgreSQL normalement.
```

Cela sera corrigé sur la prochaine version.

38.2. Les ressources sur PostgreSQL

Vous avez un support de cours, TD et TP assez complet sur Linux−Francequi décrit bien le mode d'utilisation de PostgreSQL.

38.3. Accès aux archives

Vous pourrez récupérer les documents nécessaires sous forme d'archive sur le serveur de linux−france. Pour cela voir la page d'introduction du document.

38.4. Présentation

Accès à une base de données PostgreSQL à partir d'un client WEB (Mozilla ou autres)

On veut à partir d'un client "Web" comme Mozilla (ou autres) interroger une base de données PostgreSQL. Le client HTTP passe (via des formulaires) des requêtes SQL à un serveur Web sous Linux (Apache). Celui−ci dispose d'une interface "PHP" qui lui permet d'interroger la base de données. En fait Apache va "lancer" l'exécution de "scripts PHP" et éventuellement récupérer et retourner les résultats d'exécution au client.

Les processus mis en jeu côté serveur sont les suivants :

HTTPD qui va permettre les accès via le Web (Gestion des formulaires)

Postmaster qui est le daemon gérant tous les accès à la base.

Le serveur disposera également des documents HTML et des scripts PHP

- Le travail à réaliser en TP consistera donc à :
- − Créer une base de données Postgres
- − Démarrer le daemon postmaster permettant sa gestion
- − Démarrer le daemon HTTPD
- − Accéder à la base de données (via httpd) à l'aide de scripts PHP.

38.5. Présentation de PostgreSQL

PostgreSQL est un système de gestion de base de données, développé à l'origine par l'université de Berkeley. Il s'appuie sur les modèles relationnels mais apporte des extensions objet comme :

- les classes,
- l'héritage,
- les types de données utilisateurs (tableaux, structures, listes..),
- les fonctions,
- supporte complètement SQL,
- portable sur plus de 20 environnements depuis la version 6.4.

Cela permet de qualifier PostgreSQL de système de gestion de base de données "relationnel−objet" (ORDBMS), à ne pas confondre avec les bases de données orientées objets qui ne supportent pas SQL, mais OQL (Object Query Language).

PostgreSQL est diffusé avec ses sources (licence libre).

38.5.1. Mode de fonctionnement de PostgreSQL

Les trois composantes majeures sont :

- un processus de supervision (daemon) qui prend en charge les connexions des clients : postmaster,
- les applications clientes comme psql, qui permettent de passer des requêtes SQL,
- le ou les serveurs de bases de données (*agents*). Processus d'ouverture de session : (voir le schéma d'ouverture de session.)

38.5.1.1. Description du processus d'ouverture de session

1. Le client passe une requête au daemon *postmaster* via un socket. Par défaut sur le *port 5432*. La requête contient le nom de l'utilisateur, le nom de la base de données. Le daemon, peut à ce moment utiliser une procédure d'authentification de l'utilisateur. Pour cela il utilise le catalogue de la base de données, dans lequel sont définis les utilisateurs.
2. Le daemon crée un alors un *agent* pour le client. Le processus serveur répond favorablement ou non en cas d'échec du démarrage du processus. (exemple : nom de base de données invalide).
3. Le processus client se connecte sur le processus agent. Quand le client veut clore la session, il transmet un paquet approprié au processus agent et ferme la connexion sans attendre la réponse.
4. Plusieurs processus *agents* peuvent être initialisés pour un même client.

38.5.1.2. Le dictionnaire :

Comme pour la plupart des systèmes de gestion de données, toutes les informations système sont stockées dans des tables qui forment le dictionnaire (catalogue ou repository en Anglais). Utiliser le cataloque est essentiel pour les administrateurs et les développeurs. Vous pouvez voir la structure et le contenu de ces tables système.

38.5.1.3. PostgreSQL fournit :

un langage d'administration (création de base, d'utilisateurs)

un langage d'interrogation de données basé conforme à SQL

des extensions C, C++, perl, php, python...

38.5.1.4. Les comptes utilisateurs :

Le compte administrateur de la base est par défaut " *postgres* "

il faut créer les comptes utilisateurs

Voir le TP sur HTTP pour obtenir le compte système qui est utilisé par Apache pour les requêtes http. Sur la freeduc−sup c'est "www−data". Dans la suite du document on utilisera $COMPTE_HTTP pour parler de ce compte système.

38.5.2. Langage de commande pour PostgreSQL

Voici quelques commandes d'administration de base :

Création d'une base de données : createdb

createdb [dbname]

createdb [−h host] [−p port] [−D datadir] [−u] [dbname]

Exemple : createdb −h uranus −p 5432 −D PGDATA −u demo

ou encore *createdb demo*

Suppression d'une base de données

dropdb [dbname]

Exemple *dropdb demo*

Créer un utilisateur :

createuser [username]

createuser [−h host] [−p port] [−i userid] [−d | −D] [−u | −U] [username]

−d | −D permet ou interdit la création de base à l'utilisateur

−u | −U permet ou interdit la création d'autres comptes à l'utilisateur.

Crée un compte dans pg_user ou pg_shadow. (tables système)

Si la base est accessible par Internet (exemple avec PHP), l'accès est réalisé par le compte " $COMPTE_HTTP ".

Utiliser la commande "select * from pg_user;" pour avoir la liste des utilisateurs.

Supprimer un utilisateur

drop user [username]

Accéder à une base:

psql [dbname]

psql −A [−c query] [−d dbname] −e [−f filename] [−F separator] [−h hostname] [−o filename] [−p port] −qsSt]

[−T table_options] −ux [dbname]

```
mlx@mr:~$ psql template1
Welcome to psql, the PostgreSQL interactive terminal.

Type:  \copyright for distribution terms
       \h for help with SQL commands
       \? for help on internal slash commands
       \g or terminate with semicolon to execute query
       \q to quit

template1=#
```

38.6. Présentation de PHP

PHP, (Personal Home Page) est un langage de programmation complet, assez proche du C. Il fournit :

- des structures de données,
- des structures de contrôle,
- des instructions de gestion des entrées/sorties.

Il est diffusé également sous licence libre. Il permet la création de pages web dynamiques.

Il est considéré comme une alternative à CGI, Perl, ASP (Active Server Page de Microsoft);

Développé à l'origine pour Linux, il est maintenant portable sur plusieurs environnements (Windows 9.x, NT).

Il fournit des API pour les bases de données Oracle, PostgreSQL, MySQL, DB2 ;, et est conforme aux standards ODBC et ISAPI

Il fonctionne avec de nombreux serveurs HTTP comme Apache ou IIS (Internet Information Server) de MS.

PHP peut être utilisé seul ou combiné avec des bases de données et un serveur HTTP (Objet du TP).

Simple à mettre en oeuvre, documenté, sécurisé et fiable, de nombreux sites (FAI) comme libertysurf, free mettent cet outil à la disposition des clients.

38.6.1. Mode de fonctionnement de PHP

Sur Linux, PHP est compilé comme un module dynamique ou directement intégré à Apache, ce qui accroît les performances.

Le code PHP peut être intégré directement dans une page HTML comme vb−script ou à l'extérieur sous forme de fonctions (comme CGI).

Le code est logé entre deux balises < ? Ici le code ?>. Il est possible que pour assurer la compatibilité avec XML, les balises deviennent : <php et ?>

L'extension généralement utilisée pour les documents PHP est .php. Voir ci−dessous l'exemple " test.php " qui permet de vérifier le support de PHP par votre environnement.

Listing : test.php

```
<?
        echo (" Test du module PHP ");
        phpinfo();
?>
```

38.6.2. Le langage PHP

Le guide utilisateur et ses extensions comprennent plus de 300 pages (voir les sources de documentations plus bas). La description ci−dessous donne les principales instructions pour les accès à une base de données PostgreSQL.

pg_Connect : Connexion à une base de données :

int pg_connect(string host, string port, string options, string tty, string dbname);

Retourne faux si la connexion échoue, un index dans l'autre cas. Il peut y avoir plusieurs connexions.

Exemple : $conn = $conn = pg_Connect("localhost", "5432","","", "template1");

Ou : $conn = pg_connect("dbname=marliese port=5432");

pg_Close : Fermer une connexion

bool pg_close(int connection);

pg_cmdTuples : Donne le nombre de tuples affectés par une commande insert,

update ou delete. Renvoie 0 sinon.

int pg_cmdtuples(int result_id);

Exemple :

```
<?php
$result = pg_exec($conn, "INSERT INTO verlag VALUES ('Autor')");
$cmdtuples = pg_cmdtuples($result);
echo $cmdtuples . " affectés.";
?>
```

string pg_dbname(int connection);

Donne le nom de la base de données.

Exemple $NomBase = pg_Dbaname ($conn);

pg_ErrorMessage :

string pg_errormessage(int connection);

Message d'erreur renvoyé par le serveur

pg_Exec : int pg_exec(int connection, string query);

Exécute une requête.

$UneChaineSQL = "Select * from UneTable");

Exemple : $result = pg_exec($conn, $UneChaineSQL);

pg_FieldName : string pg_fieldname(int result_id, int field_number);

Renvoie le nom du champ d'indice field_number ;

Exemple :

```
indice = 0
While (indice [lt ] NombreDeChamp)
{
$NomChamp = pg_fieldname($result, indice)
echo $NomChamp
indice ++;
}
```

pg_FieldNum : int pg_fieldnum(int result_id, string field_name);

Donne l'indice pour un nom de champ.

pg_Host : string pg_host(int connection_id);

Donne le nom du Host

pg_NumFields : int pg_numfields(int result_id);

Renvoie le nombre de champs de la requête.

Exemple : $numF = pg_Numfields($result);

pg_NumRows : int pg_numrows(int result_id);

Renvoie le nombre de tuples (enregistrements) de la requête.

Exemple : $numR = pg_NumRows ($result);

if ($numR == 0)

{

echo "Aucun enregistrement retourné. ";

exit;

}

pg_Result : mixed pg_result(int result_id, int row_number, mixed fieldname);

Renvoie la valeur d'un champ, pour un n° d'enregistrement donné et un résultat de requête. Les numéros d'enregistrement et de champ commencent à 0.

Exemple avec $i − indice d'enregistrement et $j − indice de champ :

$Valeur = pg_result ($conn, $i, $j)

pg_Options : pg_Options (int connection_id);

Renvoie une chaîne contenant les options de connexion à la base.

pg_FreeResult : int pg_freeresult(int result_id);

Libérer la mémoire.

Autres fonctions de base :

pg_Fetch_Array, pg_Fetch_Object, pg_Fetch_Row, pg_FieldsNull, pg_PrtLen,

pg_FieldSize, pg_FieldType, pg_GetLastOid, pg_port, pg_tty.

Vous trouverez la documentation de ces commandes dans celle de PHP.

38.7. Dialogue client et serveurs PHP, Apache et PostgreSQL

- Une requête SQL est passée par un formulaire HTML ou autre et via le protocole HTTP
- Le serveur Apache reçoit la requête HTTP
- Le module PHP exécute la requête sur la base PostgreSQL en utilisant les API
- Le code PHP met en forme le résultat de la requête
- La page est remise au serveur Apache
- Le serveur Apache retourne le résultat au client.

Vous avez deux méthodes pour passer les paramètres au serveur : la méthode "GET" et al méthode "POST".

38.8. Exemple de code

Voici un exemple de formulaire html et le script PHP associé.

Le formulaire : formsql.html

```
<!doctype html public "−//w3c//dtd html 4.0 transitional//en">
<html>
<head>
</head>
<body>
Lancement d'un formulaire de requête SQL via un serveur HTTP
```

```
 Utilise une base Demo
 <br>
 Entrez une chaîne sql valide − Exemple :
 <form action="resultsql.php" METHOD=post> // Ici le script qui sera exécuté
 <textarea cols="50" rows="5"
 name="c_SQL">Select * from phonebook ;</textarea></p>
 <br>
 <INPUT TYPE="submit" VALUE="Search!">
 </form>
 </body>
 </html>
```

Figure 38−1. Formulaire de saisie

Le script associé : Page resultsql.php

```
 'Solution qui permet de s'affranchir du nombre de champs.
 <?
/* Test de la connexion à la base */
 if($c_SQL != "")
 {
 echo $c_SQL ;
 $conn = pg_Connect("localhost","5432","","","demo");
  if (! $conn)
 {
  echo "Erreur de connection à la base. \n";
  exit;
  }

 /* teste le résultat de la requête */
 $result = pg_Exec($conn, $c_SQL);
  if (! $result)
 {
  echo "Erreur d'accès aux tables. \n";
  exit;
  }

/* teste le nombre de tuples retournées */
  $numR = pg_NumRows ($result);
  if ($numR == 0)
 {
  echo "Aucun enregistrement retourné. \n";
  exit;
  }

 /* Compte le nombre de champs */
 $numF = pg_Numfields($result);

 /* mise en forme du résultat sous forme tabulaire */
 /* lignes (tuples), colonnes (champ) */
 echo "<table border = 1>";
 $i = 0;
 while ($i < $numR)    {
   echo "<tr>";
  $j = 0;
    while ($j < $numF)  {
   $nc=pg_result($result,$i,$j);
   echo "<td>"; echo $nc;  echo "</td>";  $j++;

   }
  echo "</tr> \n";
  $i++;
  }
 echo "</table> \n";
/* Libère la mémoire */
 pg_FreeResult;
/* Ferme la connection */
 pg_Close($conn);
 }
 ?>
```

Figure 38−2. Résultat de la requête

Chapitre 39. Travaux pratiques : PostgreSQL

39.1. Présentation

Accès à une base de données PostgreSQL à partir d'un serveur Apache. Utilisation du langage PHP.

La maquette terminée devrait permettre, à partir d'un client HTTP comme Netscape de passer des requêtes SQL à un serveur Apache. Le serveur Apache dispose d'une interface PHP, qui lui permet d'échanger avec une base de données PostgreSQL .

Vous devrez récupérer pour le TP les documents suivants :

1. Le script de création de la base de démo "formdemo.sql"
2. le document HTML "formsql.html"
3. le document php "resultsql.php"

39.2. PostgreSQL

Connectez−vous en tant que *root.*

1 Préparation de la configuration

Installez les packages correspondant à PostgreSQL et à PHP s'ils ne sont pas déjà installés.

2 Configuration de Postgres

2.1 Postgres est installé. Le script de lancement est dans *"/etc/init.d"*

Editez ce script et relevez :

l'emplacement où sont stockées les bases de données.

Recherchez le port et les protocoles de transports utilisés par PostgreSQL avec la commande *"grep postgres /etc/services"*

Vérifiez que le fichier "/etc/postgresql/postmaster.conf" comporte bien la ligne "POSTMASTER_OPTIONS="−i −p 5432". Cela permet de définir le numéro de port, et d'inquer à PostgreSQL de supporter les sessions sockets (−i).

Vous allez configurer les options de sécurité permettant au serveur de recevoir des requêtes. Ouverz le fichier "/var/lib/postgres/data/pg_hba.conf". Recherchez les lignes ci−dessous :

```
# Put your actual configuration here
# ----------------------------------
host         all         127.0.0.1     255.0.0.0           ident sameuser
host         all         0.0.0.0       0.0.0.0             reject
```

Modifiez ces lignes après avoir fait une copie de sauvegarde de ce fichier, de la façon suivante :

```
host         all         127.0.0.1     255.0.0.0           trust
host         all         x.y.z.t       x'.y'.z'.t'         trust
```

Vous adapterez "x.y.z.t" à l'adresse de votre réseau et "x'.y'.z'.t'" au masque de votre réseau.

2.2 Il s'agit maintenant d'activer le service. Utilisez les commandes :

/etc/init.d/postgresql stop

/etc/init.d/postgresql start

Vérifiez le chargement de postgres dans la table des processus : *"ps aux | grep post"*

Vérifiez dans les journaux les messages d'erreurs si le serveur ne démarrer pas.

Vérifiez également :

− qu' un service n'est pas déjà actif,

− que les variables sont bien déclarées. En général les messages de Postgres sont assez clairs et donnent la marche à suivre pour corriger. N'allez pas plus loin tant que tout cela ne fonctionne pas parfaitement.

3 Tester la configuration

La procédure précédente à créé un modèle de base de données "template1", qui sert de modèle pour la création d'autres bases, et a créé un compte d'administrateur de base de données "*Postgres*". Toujours en mode commande et en tant qu'utilisateur postgres (*su postgres*), vous allez utiliser la commande suivante :

psql template1

Attention, il n'y a qu'un seul compte de base de données, celui de l'administrateur "postgres". Vous allez ouvir uen session sous le comtpe "root" puis passer sous le compte "postgres" avec la commande "su postgres".

Vous devriez obtenir ceci :

```
# su postgres
sh-2.05b$ psql template1
Welcome to psql, the PostgreSQL interactive terminal.

Type:  \copyright for distribution terms
       \h for help with SQL commands
       \? for help on internal slash commands
       \g or terminate with semicolon to execute query
       \q to quit

template1=#
```

Le caractère "=>" est le prompt du mode commande. Vous pouvez désormais taper des commandes.

Retenez "\q" pour quitter.

Pour avoir de l'aide sur l'interpréteur de postgres : *template1=> \?*

Pour avoir de l'aide sur les commandes SQL : *template1=> \h*

Si l'aide s'affiche, c'est que tout fonctionne. Par contre vous ne pouvez pas faire grand chose, la base et vide. Vous pouvez le vérifier avec la commande "\dt".

template1=>\dt

Couldn't find any tables!

template1=\q

Tout autre message, signifie qu'il y a un problème de configuration. Si c'est le cas vérifiez soigneusement tous les paramètres.

4 Conclusion

Votre environnement fonctionne et est bien configuré. La prochaine étape consiste à se familiariser avec les premières commandes d'administration et d'utilisation.

39.3. Test de la base

1 Créer une base de données

1 − Création de la base :

Vous devez avoir récupéré le script de création de la base "formdemo.sql"

su postgres (Vous devez être administrateur de la base)

createdb demo (création d'une base de données s'appelant *demo*)

2 − Création des tables de la base *demo* :

psql demo < formdemo.sql

2 Test de la base de données

Vous allez, au préalable, tester le fonctionnement de tout cela à partir du compte Administrateur *"postgres"*. Pour cela utilisez les commandes suivantes:

psql demo

Pour afficher les tables

=> \dt

consultez la table *phonebook*. Vous devriez avoir le résultat.

=>*select * from phonebook;* (Ne pas oublier *;*)

#quitter

=> *\q*

3 Créer un compte d'utilisateur de base de données

Vous allez créer et utiliser deux comptes utilisateurs de bases de données. "*$COMPTE_HTTP*" qui est utilisé pour les accès HTTP, "*TP1* " que vous utiliserez comme compte local. Vous leur affecterez pour l'instant les droits minimums.

3.1 Normalement le compte système*$COMPTE_HTTP* existe déjà, vous pouvez vérifier avec *"grep $COMPTE_HTTP /etc/passwd"*. Si ce n'est pas le cas, vous devrez créer un compte système pour $COMPTE_HTTP.

3.2 Création du compte système "TP1"

Création du compte

adduser TP1

affectation d'un mot de passe.

passwd TP1

3.3 Vous allez créer les comptes de base de données pour $COMPTE_HTTP et TP1. Attention aux réponses que vous mettrez car $COMPTE_HTTP ne doit pas avoir la possibilité de créer des tables, ni créer d'autres comptes de bases de données.

3.3.1 Création du compte anonyme $COMPTE_HTTP

#passer en DBA (Data Base Administrator)

su postgres

$ *createuser $COMPTE_HTTP*

Enter user's postgres ID or RETURN to use unix user ID: 99 −>

Is user "$COMPTE_HTTP" allowed to create databases (y/n) *n*

Is user "$COMPTE_HTTP" allowed to add users? (y/n) *n*

createuser: $COMPTE_HTTP was successfully added

#c'est terminé, voilà le résultat :

```
demo=# \q
sh−2.05b$ createuser www−data
Shall the new user be allowed to create databases? (y/n) n
Shall the new user be allowed to create more new users? (y/n) n
CREATE USER
sh−2.05b$
```

3.3.2 Création d'un compte DBA TP1

#passer en DBA (Data Base Administrator)

su postgres

$*createuser TP1*

Enter user's postgres ID or RETURN to use unix user ID: 501 −>

Is user "TP1" allowed to create databases (y/n) *y*

Is user "TP1" allowed to add users? (y/n) *y*

createuser: TP1 was successfully added

#c'est terminé

3.3.3 $COMPTE_HTTP n'a aucune permission sur les bases de données. Vous allez lui donner la permission de faire des "select".

Utilisez les commandes suivantes :

psql demo

grant select on phonebook to $COMPTE_HTTP ;

```
demo=# grant select on phonebook to "www-data";
GRANT
demo=#
```

\q

4 Tester l'accès des comptes

Ouvrez une session avec le compte *TP1* que vous avez créé.

psql demo

Pour afficher les tables

=> *dt*

consultez la table *phonebook*. Vous devriez avoir le résultat.

=>*select * from phonebook;* (Ne pas oublier le *;*)

#quitter

=> *q*

39.4. Serveur Apache et PHP

Démarrez le serveur Apache : */etc/init.d/apache restart*

Vérifiez que le serveur est bien actif et opérationnel.

Cherchez et relevez l'emplacement de stockage (Home Directory) des pages html d'Apache.

Vérification de la prise en charge de php par apache :

Normalement il n'y a plus rien à faire. Il s'agit de vérifier que le module PHP est bien pris en charge par Apache. Voici comment procéder:

1 − Créer dans Home Directory d'Apache le document *testphp.php* suivant:

```
<? echo ("Test du module PHP");
phpinfo();
?>
```

2 − Lancez un navigateur (à partir de votre poste ou d'une autre machine) et tapez l'url "*http://@IP de votre PC/testphp.php* » .

Deux solutions :

- soit le résultat est bon, la fonction "phpinfo()" vous retourne des informations sur le module et sur Apache. Dans ce cas vous pourrez continuer,

  Figure 39−1. Interrogation de PHP

- soit ce n'est pas le cas, il faut revoir la configuration.

39.5. Serveur PostgreSQL/Apache et PHP

Le serveur HTTP et le client fonctionnent, php est pris en charge par le serveur Apache. Maintenant, nous allons créer un formulaire qui permet de passer des requêtes SQL sur la base et un script qui exécute ces requêtes.

1 Test de la demo

1.1 En fait il s'agit de deux documents (*formsql.html* et *resultsql.php*) :

- le premier est une page HTML qui permet de saisir et "passer" des requêtes SQL,
- le deuxième au format PHP, met en forme le résultat de la requête.

1.2 Installez les pages fournies dans le répertoire d'Apache.

Lancez ensuite un navigateur. Tapez l'url http://@IP du PC/formsql.html. Vous pouvez saisir une chaîne sql et "envoyer" le formulaire.

2 Modifications

a) Copier *insert.html* dans le répertoire d'Apache, modifiez les permissions du compte $COMPTE_HTTP afin de lui donner la possibilité d'insérer des tuples dans la base de données. Modifiez le document resultsql.php afin de pouvoir réaliser des insertions dans la base. Les enregistrements à insérer seront saisies dans insert.html.

Figure 39−2. Formulaire insert.html

39.6. TP de synthèse

```
Durée de réalisation 20h en binômes
```

Vous allez créer une base de données conforme au schéma relationnel suivant :

```
cours (cou_no, cou_lib)
etudiant (etu_no, etunom)
incrit(cou_no, etu_no)
```

On demande de développer l'interface web qui permet de :

1. ajouter nouveau cours
2. supprimer un cours
3. modifier le libellé d'un cours
4. ajouter un étudiant à un cours

Chapitre 40. Surveillance, continuité de service

> Heartbeat, logiciel de gestion de cluster pour la haute disponibilité (d'après *Linux Magazine Hors Série 18 − Haute Disponibilité*)
>
> La continuité de service consiste à garantir la disponibilité de votre serveur. Si celui−ci tombe en panne, il faut être capable de basculer rapidement sur une machine de secours. De même, il faudra gérer la remise en service du serveur principal. C'est le rôle de *HEARTBEAT* que je vous propose d'installer ici...

40.1. Principe de fonctionnement

Afin de garantir la continuité de service, nous allons utiliser des machines strictement identiques au niveau du contenu et des services offerts. Le rôle de *hearbeat*, c'est la surveillance de la machine principale par la (ou les) machine(s) de secours, et son activation en cas de défaillance du serveur principal. La synchronisation des contenus et des réglages n'est pas assurée par *Heartbeat*. Ces machines fonctionneront en *cluster*, c'est à dire qu'elles formeront à elles toutes une pseudo machine. Ainsi, définissons notre pseudo machine : 192.168.1.100. Nos deux machines réelles auront pour adresses respectives 192.168.1.1, et 192.168.1.2.

Exemple 40−1. le cluster

le cluster (virtuel) *www* : 192.168.1.100

la première machine *srv−principal* : 192.168.1.1

la deuxième machine *srv−secours* : 192.168.1.2

C'est le cluster qui sera visible pour les clients. Ainsi, vous mettez a disposition la machine *www* (192.168.1.100). On décidera par exemple que c'est réellement la machine *srv−principal* (192.168.1.1) qui assure ce service, secondée par *srv−secours* (192.168.1.2). Si d'aventure *srv−principal* tombait, alors très rapidement *srv−secours* le détectera, se donnera l'adresse IP de *www* (192.168.1.100), lancera des services selon sa configuration. Au retour de *srv−principal* , on pourra, selon le fichier de configuration, continuer le service sur le serveur de secours, ou redonner la charge au serveur principal. Là encore, si il y a transfert d'identité, les scripts seront exécutés à nouveau (arrêt sur *srv−secours*, et lancement sur *srv−principal*).

40.2. Le matériel

40.2.1. Assurer la surveillance entre machines du cluster

Plusieurs solutions :

- Soit utiliser le réseau existant entre les machines, mais alors l'ensemble du réseau risque d'être pollué par des messages UDP,
- Soit utiliser de nouvelles cartes réseaux pour relier les différents noeuds du cluster (leur propre réseau de surveillance)
- Soit utiliser un câble NULL MODEM pour relier les deux noeuds du cluster par l'interface série (pour seulement deux machines)

Il est preferable de privilégier une communication spécifique entre les noeuds du cluster, afin de ne pas polluer le réseau avec les diffusions UDP associées au service *Heartbeat*. Ce faisant, on rencontre le problème suivant : On ne teste pas la rupture du câble réseau, ou un problème de carte réseau. Seul l'arrêt complet de la machine est testé (plus de signal sur le port série).

40.2.2. La surveillance sur le réseau de production

Cette méthode a l'inconvénient de polluer le réseau, et de modifier les éventuelles règles du firewall, car Heartbeat utilise des trames UDP sur le port 694 (configurable). Cet inconvénient est compensé par la surveillance de la rupture du câble, ou d'une défaillance quelconque du réseau en lui−même.

40.2.3. Le NULL−MODEM sur le port série

Le câble NULL MODEM permet d'échanger des données sur les prises séries de deux machines. Sous Linux, le port série est accessible (*/dev/ttyS0*) comme tout périphérique. Pour le tester, envoyez des données par une redirection :

Exemple 40−2. Après avoir connecté le NULL MODEM

Sur la première machine, lire le port série :

```
cat /dev/ttyS0
```

Sur l'autre machine, envoyer des données dans le port série :

```
echo "test" > /dev/ttyS0
```

Sur la première machine, le mot test devrait apparaître...

40.2.4. Le réseau de surveillance

Vous pouvez choisir de monter un réseau spécifique chargé de faire circuler les informations de contrôle des différents noeuds. Grâce à des cartes réseaux et un hub ou un switch (voir un simple câble croisé si il n'y a que deux noeuds). Dans ce cas, vous pouvez choisir par exemple d'adresser ce réseau en 10.0.0.0, et de construire la structure suivante :

Exemple 40−3. le cluster en réseau doublé

le cluster (virtuel) *www* : 192.168.1.100

la première machine *srv−principal* :

- eth0 : 192.168.1.1
- eth1 : 10.0.0.1

la deuxième machine *srv−secours*

- eth0 : 192.168.1.2
- eth1 : 10.0.0.2

Ici, le contrôle du fonctionnement de votre réseau de surveillance se fera par les outils habituels tel que ping, comme pour le réseau de production.

40.3. Le logiciel

40.3.1. L'installation

Sur une debian stable, l'installation se fait par la méthode standard :

```
apt−get install heartbeat
```

Debconf va ensuite vous aider à configurer ce service. J'ai preferé construire mes fichiers de configuration manuellement.

40.3.2. les fichiers de configuration

Il y a trois fichiers de configuration principaux :

- *ha.cf* contient les réglages de la machine réelle.
- *haresources* contient les réglages du cluster (la machine à simuler).
- *authkeys* contient les modes d'authentification (utile si on utilise le réseau de production.

Il est important de noter qu'*haresources* décrit le cluster tel qu'il doit être vu (notamment la 'fausse' adresse IP), tandis qu'*ha.cf* décrit le fonctionnement du noeud réel. Ainsi, *ha.cf* pourra varier selon les machines (nom de la carte réseau différent, logs réglés différemment) alors que *haresources* est lui PARTOUT IDENTIQUE (tous les noeuds doivent être d'accord sur le cluster à simuler).

40.3.2.1. Le fichier ha.cf

petit tour du propriétaire :

- choix de médium de surveillance

  la carte réseau

  ```
  bcast eth0
  ```

  le câble NULL MODEM

  ```
  baud 19200
  ```

  ```
  serial /dev/ttyS0
  ```
- logs

  Les pépins

  ```
  debugfile /var/log/hda-debug
  ```

  Les évènements

  ```
  logfile /var/log/hda-log
  ```

  la facility (la sémantique des messages, voir le cours sur syslog)

  ```
  logfacility  local0
  ```
- délais de réaction (en secondes)

  délai entre deux 'pouls'

  ```
  keepalive 2
  ```

  durée du silence avant de déclarer un noeud 'mort'

  ```
  deadtime 10
  ```

  durée du silence avant de déclencher une alerte

  ```
  warntime 8
  ```

  temporisation lors du lancement du service (si un noeud est lent)

  ```
  initdead 30
  ```
- réglages generaux

  port UDP pour le 'pouls'

  ```
  udpport 694
  ```

  liste des noeuds participants au cluster

  ```
  node srv-principal
  ```

  ```
  node srv-secours
  ```

  le serveur principal reprend la main lors de son retour

  ```
  nice_failback on
  ```

La première partie permet de définir le mode d'écoute du logiciel de surveillance. Il semble que l'emploi du port série implique de bien définir les bauds avant. Vous pouvez combiner plusieurs modes d'écoute, c'est alors le silence sur tous ces médias qui entraînera les actions d'*heartbeat* .

La partie sur les logs permet de bien contrôler le fonctionnement du logiciel et valider les échanges entre les différents noeuds. Vous pouvez choisir les noms de fichiers que vous désirez, ainsi que la facility (pour le traitement par le démon syslogd).

La gestion de temps de réaction peut être donnée en millisecondes (par défaut, elle est en seconde) en ajoutant le suffixe ms (keepalive 200ms). Attention au initdead qui permet de gérer un noeud un peu lent a démarrer, et ainsi éviter de l'exclure immédiatement du cluster. La documentation préconise de doubler au minimum ce temps par rapport au deadtime.

Enfin, partie TRES IMPORTANT, la liste des noeuds qui participent au cluster. Lorsque tous les noeuds seront actifs, alors *heartbeat* activera le cluster et les services qui lui sont associés.

Ce fichier peut legerement différer selon les noeuds, dans les deux premières parties uniquement.

40.3.2.2. Le fichier haresources

Fichier de description du cluster a simuler. IL DOIT ETRE IDENTIQUE SUR TOUS LES NOEUDS !. Ce fichier décrit la machine maître parmi tous les noeuds du cluster, la ou les adresses IP à simuler, et les services à assurer sur ce cluster.

Dans notre exemple, si on veut que notre cluster *www* ait l'adresse IP 192.168.1.100, que *srv−principal* soit la machine qui assure prioritairement ce rôle, et que *www* soit un serveur Web et mysql, alors tous les noeuds auront le fichier *haresources* suivant :

Exemple 40−4. haresources pour tous

```
srv−principal   192.168.1.100   apache  mysql
```

Avec un tel fichier, automatiquement, dès la mise en route du cluster, le service Web et le serveur de base de données seront automatiquement activés. Si le serveur principal tombe, alors le serveur de secours prend le relais, lance le Web et Mysql, se donne la bonne adresse IP, fait un broadcast ARP pour avertir le réseau...

40.3.2.3. Le fichier authkeys

Ce fichier détermine le niveau de securite des échanges entre les différents noeuds du cluster. Si vous êtes sur un médium fiable, vous pouvez utiliser le niveau le plus bas de sécurité (*crc* = simple contrôle de contenu, par l'utilisation d'une somme de contrôle, aucun cryptage). Il est possible d'utiliser des systèmes plus puissant (crypté avec *md5* ou mieux encore (mais plus gourmand en temps processeur) *sha*). Dans les deux derniers cas, il faut fournir une clé...

Exemple 40−5. authkeys

```
auth 1
1 md5 "Y'a pas d'os dans les bananes"
2 crc
3 sha "Mais y'en a dans les SE"
```

Dans cet exemple, les trois modes sont prêts à fonctionner, avec les clés associées aux services, et c'est le md5 qui est choisi (auth est à 1). Ces clés doivent bien sur être les mêmes sur l'ensemble des noeuds participant au cluster.

40.3.3. Mise en route

Tous les fichiers de configuration mis en place sur tous les noeuds, vous pouvez alors lancer le service (pour les recopies, vous pouvez éventuellement vous aider de *scp*). le lancement du service se fait de la façon habituelle.

```
/etc/init.d/heartbeat start
```

Des que le cluster est constitue (tous les noeuds sont actifs), alors *srv−principal* héritera d'une nouvelle adresse IP (en ip−aliasing) 192.168.1.100, et les services apache et mysql seront alors lancés (si ce n'était déjà fait). Les scripts de lancement apache et mysql sont ceux trouvés à l'endroit habituel (/etc/init.d). Le cluster fonctionne, et les clients peuvent s'y connecter.

Vous pouvez surveiller les fichiers de logs (grâce à *tail −f /var/log/ha−log*), et faire joujou en coupant brutalement le courant sur le serveur principal. Testez le *nice_failback*, la mise en route de différents services, ainsi que le réseau (*ifconfig* sur les noeuds, et sniff du réseau pour voir les broadcast ARP lors des changements de serveurs...

40.4. Exercices

1. Montez deux machines qui formeront le cluster, connectées à un réseau qui comporte aussi un client. Le client aura l'adresse IP 192.168.1.1, les deux serveurs 192.168.1.100 et 101. Le cluster aura l'adresse 192.168.1.200.
2. Connectez le câble NULL MODEM. Vérifiez son bon fonctionnement.

3. installez *Heartbeat*. Configurez le simplement, avec le 'pouls' sur le port série. Pour le moment, on ignore les différents services, seule l'adresse IP doit être gérée par le cluster.
4. Sur le premier serveur, lancer heartbeat. Visualisez en continu le fichier de logs. Contrôlez les informations réseaux.
5. Sur le second serveur, lancer heartbeat. Visualisez en continu le fichier de logs.
6. Contrôlez le premier serveur : fichier de logs, et paramètres réseaux. Que remarquez vous ?. Faites des pings du cluster (192.168.1.200), et des deux serveurs. Contrôlez le contenu du cache arp (arp −a). Que remarquez vous ?
7. Éteignez violemment le serveur principal. Consultez les logs du serveur secondaire. Que s'est il passé ? Affichez les informations réseaux. Que remarquez vous ? Affichez le cache arp du client. Que remarquez vous (bien que vous n'ayez lancé aucune demande de résolution arp) ?
8. *Heartbeat* émet un 'flood' de réponses ARP (bien que personne n'ait posé de questions ARP) afin de forcer la mise à jour des caches ARP de tous les clients, le plus rapidement possible. Essayez de visualiser ces réponses ARP, en sniffant le réseau à partir du client, et en relançant le serveur principal. Vérifiez grâce aux logs, à ifconfig, et à votre sniff réseau l'ensemble des caractéristiques du logiciel.
9. Modifiez la configuration sur l'ensemble des noeuds afin de créer un cluster dont l'adresse est 192.168.1.210, et qui assure le fonctionnement d'un serveur Web et d'une base de données MySql. Le pouls sera émis sur le port série et sur la carte réseau. Enfin, la remise en route du serveur principal ne déclenchera pas sa promotion, le serveur de secours restant le support du service jusqu'à sa défaillance. Vérifiez bien que les deux services à assurer ne sont pas lancés lors de l'init de la machine.
10. Lancez le service *heartbeat* sur le premier serveur. Apache et MySql se lancent ils ?
11. Lancez le service *heartbeat* sur le second serveur. Apache et MySql se lancent ils ?
12. Testez le service à partir du client. Sniffez le réseau.
13. Éteignez le premier serveur. Que se passe t'il sur le second ? Vérifiez si tous les services sont bien lancés. Arrêtez le sniff réseau du client. Retrouvez le 'pouls', l'arrêt de celui−ci, puis le flood de réponse ARP.

Chapitre 41. Lilo : Linux Loader

Desciption des mécanismes de boot avec lilo

Le démarrage de votre système GNU/Linux ou d'autres systèmes, c'est le rôle de *LILO*

41.1. Objectifs

- Comprendre le fonctionnement de Lilo
- Adopter une stratégie d'installation de Linux

41.2. Présentation de Lilo

- En particulier : Lilo est un chargeur de Linux, LInux LOader
- En général : Lilo est un chargeur de systèmes d'exploitation

Lilo permet de charger sur une machine:

- plusieurs systémes d'exploitation différents (multi boot)
- plusieurs noyaux différents de Linux (ex: pour des tests)

41.2.1. Lilo

Prend en charge :

- les différentes partitions d'amorçage
- le chargement de Linux (disquette ou disque dur)
- la cohabitation d'autres systèmes (ex: Windows, *BSD, NT, ...)

D'autres systèmes ont des chargeurs de systèmes. Si Lilo est le principal chargeur des systèmes il est :

- le chargeur principal, sinon il est
- le chargeur secondaire

41.2.1.1. « Lilo est limité à 1024 cylindres. » FAUX !

Les versions récentes de lilo utilisées depuis Debian Potato supportent lba32. Si le BIOS de la carte mère est assez récent pour supporter lba32, lilo devrait être capable de charger au−delà de la vieille limite des 1024 cylindres. Assurez−vous simplement d'ajouter la ligne « lba32 » vers le début de votre fichier `/etc/lilo.conf` si vous avez gardé une ancienne version de ce fichier.

41.3. Documentation

- la documentation de lilo : en général dans `/usr/share/doc/kernel−doc−(version)` ou `/usr/doc/kernel−doc−(version)`
- les pages de manuels : **man *lilo lilo.conf***

- la documentation de votre distribution
- le guide du ROOTard

41.4. Avant de commencer

Lilo a besoin d'informations pour accéder au répertoire /boot Ce répertoire est normalement sur la partition principale appelée / ou racine (root). quelques rappels et précisions sur :

- le système de gestion de fichiers (sgf) de Linux,
- les systèmes de partitions des disques.

41.4.1. Linux SGF

Exemple de partitionnement qu'il faudra adapter à votre environnement.

```
/ (root)    Fichiers de démarrage (/ + /boot + /bin + /sbin) (min 50MB)
/boot       Partition contenant les images du noyau (Kernel) (Min 16MB)
/tmp        Fichiers temporaires du système et des utilisateurs (min 100MB)
/var        Journaux, spool (/var/log, /var/spool) (min 100MB)
/home       Fichiers utilisateurs (min 100MB)
/usr        Applications (min 300MB à 700MB avec X)
/usr/local  Applications (en général hors distribution) (100MB)
/usr/src    Sources des packages et du noyau
swap        Fichier d'échange, à adapter en fonction de la RAM (min 160 MB)
```

Lignes directrices pour la mémoire DRAM

```
Ce qui suit sont des indications grossières pour la DRAM.

    4 Mo :  Minimum suffisant pour faire fonctionner le noyau Linux.
   16 Mo :  Minimum pour un usage du système en mode console.
   32 Mo :  Minimum pour un système X simple.
   64 Mo :  Minimum pour un système X avec GNOME/KDE.
  128 Mo :  Confortable pour le système X avec GNOME/KDE.
  256+Mo :  Pourquoi pas si vous le pouvez. La DRAM est bon marché.
```

L'option de boot mem=4m (ou lilo append="mem=4m") montrera comment le système se comporterait en ayant 4Mo de mémoire installée. Un paramètre de démarrage pour lilo est requis pour un système ayant plus de 64Mo de mémoire avec un vieux BIOS.

En fonction de la destination du serveur (impression, mail, news, ...), il faudra prévoir une partition ou un disque pour les sous−systèmes.

41.4.2. Les partitions

- Un système de disque comporte un disque système ou une partition système, c'est le disque ou la partition de BOOT
- Un disque peut comporter plusieurs partitions primaires dites d'amorçage. Une partition d'amorçage est une partition sur laquelle il est possible d'amorcer un système d'exploitation. Ces partitions sont appelées partitions primaires.
- Sous Linux chaque disque peut comporter 4 partitions. 3 partitions primaires et une partition étendue, ou bien 4 partitions primaires par exemple.
- Un disque peut supporter jusqu'à 16 partitions. Exemple : 3 partitions primaires plus 12 partitions logiques sur une partition étendue.

41.4.3. Disque IDE ou EIDE

Si vous avez 2 bus IDE1 et IDE2 avec 2 disques chacun vous pouvez:

- installer Linux (/boot) sur n'importe quelle partition primaire d'un des disques,
- installer Lilo sur le MBR du disque système si Lilo est le chargeur primaire,
- installer Lilo sur le PBR de la partition de Linux s'il y a un autre chargeur.

41.4.4. Disques E(i)DE et CDROM

Si vous avez un lecteur CDROM en deuxième lecteur sur le bus IDE1 vous devrez:

- installer Linux (/boot) sur n'importe quelle partition primaire du premier disque du bus IDE1,
- installer Lilo sur le MBR du disque système si Lilo est le chargeur primaire,
- installer Lilo sur le PBR de la partition de Linux s'il y a un autre chargeur.

41.4.5. Disques E(i)DE et SCSI

Si vous avez un bus SCSI et un bus IDE vous pourrez:

- installer Linux (/boot) sur n'importe quelle partition primaire du premier disque du bus IDE1,
- installer Linux (/boot) sur n'importe quelle partition primaire du disque d'id0 sur le bus SCSI, (les autres id ne fonctionnent pas),
- installer Lilo sur le MBR du disque système su Lilo est le chargeur primaire,
- installer Lilo sur le PBR de la partition de Linux s'il y a un autre chargeur.

41.4.6. Disques SCSI

Si vous avez 2 disques SCSI, vous pourrez:

- installer Linux (/boot) sur le disque d'id0 ou id1 (les autres id ne fonctionneront pas),
- installer Lilo sur le MBR du disque système si Lilo est le chargeur primaire,
- installer Lilo sur le PBR de la partition de Linux s'il y a un autre chargeur.

Attention avec les Bus SCSI

- connectez le lecteur de CDROM sur le connecteur le plus proche du processeur,
- affecter lui un id supérieur à celui du dernier disque,
- vérifiez la terminaison physique et logique du Bus.

41.4.7. Restriction du BIOS

- Certains BIOS ne supportent pas de partitions d'amorçage au dessus du cylindre 1023,
- Attention si vous utilisez des outils comme fips.
- Ne créez pas de partition au−dessus de cette étendue avec un BIOS incompatible.

41.5. Installation

41.5.1. MBR et PBR

Installez Lilo

- sur le MBR (Master Boot Record), si vous n'avez pas d'autres chargeurs de système. Le MBR contient un enregistrement qui est chargé à chaque démarrage de la machine.
- sur le PBR (Partition boot Record) si vous utilisez un autre chargeur de système comme celui d'OS/2 ou Windows NT.

41.5.2. Installer Lilo

Quand ?

- Pendant l'installation de Linux.
- Après l'installation de Linux.

Où ?

- Sur une partition du disque dur. *Dans ce cas il faudra bien choisir cette partition*

- Sur le MBR si Lilo est le chargeur primaire
- Sur un PBR si Lilo est le chargeur secondaire (dans ce cas le chargement des systèmes est pris en charge par un autre chargeur)
- Linux doit être installé sur une partition primaire (ne pas utiliser de partition étendue)
- On peut modifier la partition de chargement en modifiant le fichier de configuration de Lilo (/etc/lilo.conf)

Attention: Si vous installez Lilo sur un MBR, alors qu'il y a déjà un chargeur, vous supprimerez le chargeur installé.

41.5.3. Dos ou Windows 9.x

- Créez une partition pour Windows, installez Windows,
- Créez les partitions pour Linux, installez Linux,
- Installez Lilo sur le MBR

Pour lancer ou relancer lilo après une modification de configuration : */sbin/lilo*

Attention : Linux ne lit pas naturellement les Fat 32 bits.

41.5.4. Windows NT

Windows NT propose son chargeur.

- Créez les partitions pour Windows NT et installez NT.
- Créez les partitions pour Linux et installez Linux.
- Installez Lilo sur le PBR de sa partition d'amorçage.
- Ajoutez Linux dans la table des partitions d'amorçage de NT (avec un utilitaire comme bootpart par exemple).

Le chargeur de NT permettra le chargement de Linux

41.5.5. Exemple avec 3 systèmes

Avec Windows 9.x et NT:

- créez les partitions appropriées,
- installez Windows 9.x,
- installez Windows NT et son chargeur,
- installez Linux sans mettre Lilo sur le MBR.

Attention aux systèmes de fichiers Fat, Fat 32, NTFS, ext2fs.

41.5.6. Avec d'autres systèmes

- voyez la documentation du système,
- les FAQs et HOWTOs,
- les forums de discussions.

41.6. Lilo

- Lilo crée une copie de sauvegarde du secteur de Boot (MBR) dans le répertoire /boot.
  - avec un disque IDE `/boot/boot.0300`
  - avec un disque SCSI `/boot/boot.0800`
- il est possible de restaurer ce secteur de boot:
  - ```
    dd if=/boot/boot.0300 of=/dev/hda bs=446 count=1
    ```

Explication On restaure le secteur de boot d'une partition primaire sur /dev/hda (disque IDE). Seuls les 446 premiers octets des secteurs sont nécessaires, les autres contiennent des informations sur les tables des partitions.

Attention il y a un risque, faire au préalable une disquette de démarrage.

41.6.1. Exécution de Lilo

Pour s'exécuter correctement, Lilo à besoin:

- d'information sur le système (type de disques, contrôleurs, ...)
- d'un fichier de configuration (option d'exécution, partition d'amorçage, ...).

41.6.2. Options de configuration

- *boot* partition qui contient le secteur de boot
- *delay* durée en 1/10 de seconde pendant laquelle le chargeur attend
- *map* emplacement du fichier de carte du noyau ou sinon `/boot/map`
- *prompt* affiche une invite au démarrage afin que l'utilisateur entre un choix
- *image* indique, pour la section, quelle image charger
- *label* alias permet de choisir entre plusieurs systèmes.
- *append* append permet de passer des paramètres au noyau, par exemple pour activer un peripherique specifique (graveur ide en SCSI).
- *disk et bios* disk et bios remplace le mapping entre nom de disque et l'ordre des disques dans le BIOS. A utiliser avec precaution.

41.6.3. Outils de configuration

- manuellement pour la création des fichiers et l'installation de Lilo,
- utilisation de l'environnement graphique (linuxconf ou webmin par exemple).

    ```

41.6.4. Exemple de fichier de configuration `/etc/lilo.conf`

```
# Support des disques de grande taille
lba32
#
#disk=/dev/hde
#    bios=0x81
#disk=/dev/sda
#    bios=0x80

boot=/dev/hda
map=/boot/map
install=/boot/boot.b
vga=normal
default="linux"
keytable=/boot/fr_CH-latin1.klt
prompt
nowarn
timeout=100
message=/boot/message
menu-scheme=wb:bw:wb:bw
root=/dev/hda1

image=/boot/vmlinuz
        label="linux"
        initrd=/boot/initrd.img
        append="devfs=mount hdc=ide-scsi acpi=ht resume=/dev/hda1 splash=silent"
        vga=788
        read-only

image=/boot/vmlinuz-old
        label="old"
        initrd=/boot/initrd.img-old
        append="devfs=mount hdc=ide-scsi acpi=ht resume=/dev/hda1 splash=silent"
        vga=788
        read-only

image=/boot/vmlinuz-suze
        label="suse"
        root=/dev/hda2
        initrd=/boot/initrd.img
        read-only

other=/dev/hda3
        label="win2k"

image=/boot/memtest-1.11.bin
        label="memtest-1.11"
```

Ici, il y a 4 images, et 4 systèmes différents

41.6.5. Désinstaller Lilo

- si on veut installer un autre système sur la machine
- si Lilo est installé sur le MBR et doit être déplacé sur un PBR

*Utilisez la commande "**fdisk /mbr**" sous Windows ou "**/sbin/lilo -u** sous Linux"*

41.7. Choix du système

- Au démarrage du système utilisez les touches CTRL ou SHIFT. Les touches CTRL et SHIFT vont temporiser le système, pour permettre à l'utilisateur d'entrer une commande
- Au prompt "boot:" utilisez la touche TAB pour voir la liste des différents systèmes. La touche TAB affichera la liste des systèmes (champ "label" de `/etc/lilo.conf`).

41.8. Autres solutions sans Lilo

- Démarrer Linux avec la disquette de démarrage (si on ne désire pas modifier la configuration d'une machine)
- utiliser d'autres chargeurs commerciaux,
- utiliser loadlin, (pour charger Linux à partir de Windows)
- syslinux, chboot

41.8.1. Loadlin

loadlin permet de charger Linux à partir d'un prompt MS-DOS

- Exemple : Linux est installé sur la partition `/dev/sda2`
- Commande : loadlin NomDuNoyau root=/dev/sda2

41.9. rdev

- si Lilo n'est pas installé, les valeurs codées dans le noyau sont prises en compte.
- ces valeurs sont codées avec la commande *rdev*.

La commande *rdev* permet de voir quelle est la partition d'amorçage.

41.10. initrd

initrd est un fichier spécial (disque RAM), initialisé par Lilo, avant de charger le noyau,

Il permet de "pré−charger" les modules contenu dans `/etc/modules.conf`,

L'appel à initrd est configuré dans `/etc/lilo.conf`

Pour en savoir plus : *man initrd mkinitrd mknod*

41.10.1. Modules

Les modules pour les pilotes de périphériques sont configurés lors de l'installation initiale. modconf permet de configurer les modules ensuite au travers d'une interface utilisant des menus. Ce programme est utile lorsque des modules ont été oubliés lors de l'installation ou lorsqu'un nouveau noyau est installé.

Le nom des modules à précharger est listé dans `/etc/modules`. Utilisez **lsmod** et **depmod** pour les contrôler manuellement.

`/etc/modules.conf`

```
above snd−via82xx snd−pcm−oss
probeall scsi_hostadapter aic7xxx ide−scsi
alias ieee1394−controller ohci1394
probeall usb−interface usb−uhci ehci−hcd
alias sound−slot−0 snd−via82xx
alias eth0 tulip
alias autofs autofs4
alias eth1 via−rhine
```

41.10.2. initrd (suite)

Exemple de mise à jour, suite à la modification de `/etc/modules.conf`:

- Modification de `/etc/modules.conf`

  ```
  alias scsi_hostadapter buslogic
  ```
- Création de la nouvelle image **/sbin/mkinitrd −o /boot/NouveauNomInitrd.img numero_version_kernel**

41.11. Conclusion

Vous avez vu :

- comment s'effectue le chargement de Linux,
- le rôle de lilo,
- comment installer et configurer Lilo,
- comment désinstaller lilo,
- adopter une stratégie pour installer plusieurs systèmes d'exploitation sur une machine Linux.

Chapitre 42. Travaux pratiques : Kernel et Noyau

Mise en oeuvre des mécanismes de boot et optimisation du noyau.

Le but de ce TP est de savoir gérer le chargement du système, de la phase de boot à celle du chargement du noyau et de ses modules, init et les niveaux d'exécutions seront abordes dans un autres TP.

42.1. Objectifs

- Utiliser Lilo
- Utiliser Grub
- Dé/Chargement de modules
- Librairies
- Installation d'un noyau standard

- Installation d'un noyau a partir des sources

42.2. Quelques remarques

- Durant l'installation, on sera interrogé sur le matériel ou les puces. Parfois, ces informations ne sont pas toujours faciles à trouver. Voici une méthode :
    - Ouvrez le PC et inspectez l'intérieur.
    - Notez les codes produit qui sont sur les grandes puces de la carte graphique, de la carte réseau, sur la puce à côté des ports série et la puce à côté des ports IDE.
    - Notez les noms des cartes imprimés au dos des cartes PCI et ISA.
- Relevez dans /etc/lilo.conf, la configuration initiale et effectuez une copie de ce fichier avant toute modifications.
- Relevez a l'aide de fdisk ou cfdisk le mapping de votre disque dur.
- Relevez a l'aide des commandes ci dessous, les caracteristiques materiel de votre poste de travail.

```
$ lspci -v |less
$ pager /proc/pci
$ pager /proc/interrupts
$ pager /proc/ioports
$ pager /proc/bus/usb/devices
```

42.3. Compilation

A partir d'une distribution Debian GNU/Linux

- Lisez attentivement ce document une premère fois sans lancer les commandes.
- Lisez egalement les pages de manuel des commandes avec man et info

Notez la version actuelle de votre noyau :

```
uname -a
ls -l /boot/
ls -l /lib/modules/
ls -l /usr/src/
```

ou

```
cat /proc/version
```

Affichez les modules chargés et repérez celui chargé du support de la carte réseau

```
lsmod
```

lancez la commande ci-dessous, choisissez la dernière version du noyau en relançant à nouveau la commande avec le nom + version

```
apt-get install debhelper modutils kernel-package libncurses5-dev
apt-get install kernel-source-[2.4.26]   # utilisez la dernière version
apt-get install fakeroot
```

entrez votre nom et adresse électronique

```
vi /etc/kernel-pkg.conf
```

déplacez vous dans le répertoire où se trouve les sources

```
cd /usr/src
```

décompactez l'archive

```
tar xjvf kernel-source-(nouvelle-version).tar.bz2
```

supprimez le lien symbolique linux qui pointe sur la dernière version du noyau, s'il existe.

```
rm /usr/src/linux
```

créez un lien symbolique vers votre nouvelle version

```
ln -s /usr/src/kernel-source-(nouvelle-version) /usr/src/linux
```

déplacez vous dans les sources

```
cd /usr/src/linux
```

consultez les fichiers avec par exemple **mc**

```
Documentation/*
README
README.debian
Makefile
```

lancez l'interface de configuration du noyau afin de constater que par defaut la selection des modules du noyau ne correspond pas à celle que vous avez, c'est normal car à ce stade vous n'avez pas récupéré votre ancienne configuration (fichier .config dans les sources).

! pour le TP quitter sans sauver ! Si vous avez lu trop tard, lancer les commandes suivantes afin de purger vos sources

```
make mrproper
```

pour l'interface sous X

```
make xconfig
```

en mode console texte et ultra−basique

```
make menuconfig
make config
```

vérifiez la présence des modules réseau, video, ... de votre machine

```
less /boot/config−(version−actuelle)
```

verifiez également leur présence dans `/lib/modules/kernel−(version−actuelle)/`...

Vous pouvez egalement constater leur chargement dans le système et dans le fichier de chargement

```
lsmod
cat /etc/modules
```

copiez le fichier de config par defaut

```
cp /boot/config−(version−actuelle) /usr/src/linux/.config
```

alternative sous Suse : **make cloneconfig**

demarrez la configuration de votre nouveau noyau

```
make menuconfig
```

enlevez le module de votre carte réseau

au cours d'une autre manipulation vous pourrez refaire ce TP en retirant tout support qui ne concerne pas votre architecture materielle, afin d'obtenir un noyau optimisé à vos besoins, attention toutefois à ne pas retirer les modules necessaires au fonctionnement de votre environnement.

help sur un module vous donne son label que vous retouvez dans .config

quittez et sauvez les modifications

regardez vos modification par rapport au fichier d'origine

pour voir vos modifications

```
diff /boot/.config−(version−actuelle) .config
```

éditez le fichier Makefile et mettez votre EXTRAVERSION personelle

profitez−en pour parcourir ce fichier et voir ce que font les différentes cibles (dep, clean, bzImage, ...)

copiez votre `/usr/src/linux/.config` dans /boot

```
cp .config /boot/config−(nouvelle−version)
```

lancez les commandes de compilation suivantes:

```
make dep
make clean
make bzImage
make modules
```

Vous pourriez également enchaîner les traitements en une seule commande : make clean bzImage modules

Attention ! A ce point si vous recompiliez la même version de kernel, vous devez déplacer `/lib/modules/kernel-(version-actuelle)`, afin de ne pas génerer, dans ce dernier, les modules issue de votre nouvelle compilation

```
mv /lib/modules/kernel-(version-actuelle) /lib/modules/kernel-version-actuelle-old
```

continuez ensuite par installer les modules

```
make modules_install
ls -l /lib/modules/ # Vous permet de constater le créations du repertoire contenant les modules de kernel-(vers
```

copiez les fichiers (noyau et System.map) créés par la compilation dans /boot

```
cp arch/i386/boot/bzImage /boot/vmlinuz-(nouvelle-version)
cp System.map /boot/Sytem.map-(nouvelle-version)
```

Si `/boot/System.map` existe et est un lien symbolique, vous devez le supprimer pour établir un lien avec le nouveau fichier.

```
rm  /boot/System.map
ln -s /boot/Sytem.map-(nouvelle-version)  /boot/Sytem.map
```

Supprimer `/vmlinuz.old` s'il existe, ou renommer le

```
rm /vmlinuz.old
```

permuttez le lien symbolique `/vmlinuz`

```
mv /vmlinuz /vmlinuz.old
```

créez un lien symbolique pointant sur la nouvelle image du noyau

```
ln -s /boot/vmlinuz-(nouvelle-version) /vmlinuz
```

Editer lilo.conf et activer la section vmlinuz.old (si ce n'est pas fait) et valider vos changements

```
lilo -v
```

rebooter

```
shutdown -r now ou reboot
```

Extras, à faire plus tard

il est possible de rendre totalement automatique l'installation du noyau en lançant la commande (ne l'utilisez pas pour le TP)

```
cd /usr/src/linux
make modules_install install
```

Fabriquer un noyau « debianisé » en simple utilisateur: recuperer les sources du kernel et de pcmcia dans votre repertoire

```
$ cd ~/kernel-source-(nouvelle-version)
$ make menuconfig
$ make-kpkg clean
# N'utilisez pas --initrd,  initrd n'est pas utilisé dans notre cas.
$ fakeroot make-kpkg --append_to_version -486 --initrd \
             --revision=rev.01 kernel_image \
             modules_image # modules_image pour pcmcia-cs* etc.
$ cd ..

# dpkg -i kernel-image*.deb pcmcia-cs*.deb # installation en tant que root
```

En réalité, make-kpkg kernel_image lance make oldconfig et make dep.

Si vous voulez les modules de pcmcia-cs, ou pas de support pcmcia, sélectionnez dans make menuconfig « General setup » dans « PCMCIA/CardBus support » et le déselectionner (càd décocher la case).

Sur une machine SMP, configurez CONCURRENCY_LEVEL selon kernel-pkg.conf(5).

Autre possibilité

```
cd /usr/src/linux
make-kpkg -revision debidon.2.4.26 kernel_image
```

après cette commande un noyau debianisé est créé dans /usr/src/

un utilisateur peut faire un package de kernel en utilisant fakeroot pour l'installer ensuite sur une autre machine, ou encore le stocker dans un repertoire dont le sources.list d'apt pointerait dessus.

```
fakeroot make-kpkg -revision debidon.2.4.26 kernel_image
```

root peut ensuite l'installer en lançant la commande

```
dpkg -i kernel_image-2.4.26_debidon.2.4.26_i386.deb
```

regardez les commandes **mkboot** et **installkernel**

42.4. Installation et activation de module

Affichez la liste des modules chargés et constatez que le module réseau n'est plus la, ni l'interface

```
lsmod
ifconfig -a
```

déplacez-vous dans les sources

```
cd /usr/src/linux
make menuconfig # (sélection du module de la carte réseau comme module)
```

quittez et sauvez

```
make dep
make modules modules_install
```

contrôlez les dépendences des modules

```
depmod -a
```

charger à chaud le module dans le systéme

```
modprobe nom_module
```

ou (ie remplacer alias par dans notre cas eth0) (voir /etc/modules.conf)

```
modprobe alias
```

contrôlez la présence du module

```
lsmod
```

réeactivez le réseau

```
/etc/init.d/networking restart
```

NB: pour charger/décharger les modules du noyau.

```
modconf
```

si modules.dep ne se trouve pas dans /lib/modules/(version-encours)

```
depmod -a
```

pour prendre en compte un module au chargement, ajouter le dans /etc/modules

```
man modules.conf
```

42.4.1. make-kpkg pour les modules

exemple pcmcia-cs + pcmcia-sources

```
cd /usr/src
```

récupérer sur http://pcmcia-cs.sourceforge.net/ et décompresser les sources pcmcia dans le repertoire modules après install du noyau

```
cd /usr/src/modules/pcmcia-cs
make clean
make config
cd /usr/src/linux
make-kpkg --revision=custom.1.0 modules_image
cd /usr/src/
dpkg -i pcmcia-modules*.*.deb
```

Pour plus d'information sur la compilation http://www.debian.org/doc/manuals/reference/ch-kernel.fr.html

42.5. Utilisation de Grub

Le nouveau gestionnaire de démarrage grub du projet GNU Hurd peut être installé sur un système Debian Woody

```
# apt-get update
# apt-get install grub-doc
# mc /usr/share/doc/grub-doc/html/
     ... lisez le contenu
# apt-get install grub
# pager /usr/share/doc/grub/README.Debian
     ... à lire
```

Comment configurer les paramètres de démarrage de GRUB GRUB est un nouveau gestionnaire de démarrage issu du projet Hurd et est beaucoup plus flexible que Lilo mais a une manière différente de gérer les paramètres de démarrage.

```
# grub
     grub> find /vmlinuz
     grub> root (hd0,2)
     grub> kernel /vmlinuz root=/dev/hda3
     grub> initrd /initrd # ne pas utiliser dans notre cas
     grub> boot
```

Là, vous devez connaître les noms de périphériques de Hurd :

```
     the Hurd/GRUB          Linux              MSDOS/Windows
      (fd0)              /dev/fd0           A:
      (hd0,0)            /dev/hda1          C: (habituellement)
      (hd0,3)            /dev/hda4          F: (habituellement)
      (hd1,3)            /dev/hdb4          ?
```

Voir `/usr/share/doc/grub/README.Debian` et `/usr/share/doc/grub-doc/html/` pour les détails.

Pour modifier le menu de GRUB, éditez `/boot/grub/menu.lst`. Voir Comment configurer les paramètres de démarrage de GRUB, pour la configuration des paramètres de démarrage car la syntaxe est différente de celle de lilo.

42.6. Librairies

Il faut distinguer les librairies statiques des librairies dynamiques.

Les libriaires statiques possèdent l'extension ".a". Ces librairies sont liées statiquement avec le binaire (programme). C'est à dire que durant la phase de linkage (édition des liens lors de la compilation) le compilateur va prendre le code des fonctions nécessaires de la librairie et les mettres en dur dans le binaire.

Les librairies dynamiques possèdent l'extension ".so". Ces libriaires sont liées dynamiquement avec le binaire. Dans ce cas, le code des fonctions utilisées par le programme ne se trouve pas dans le binaire. Mais lorsque ce bout de code est requis par le programme, il va charger ce code dynamiquement (durant l'exécution du programme).

Un programme lié statiquement avec une librairie sera plus gros en taille que pour une libriaire dynamique.

Pour les libriaires dynamiques, voici le fichier à configurer qui donne les chemins où se trouvent les librairies dynamiques `/etc/ld.so.conf`

La commande ldconfig parcours les chemins spécifiés dans le fichier de configuration et construit un cache. Ce cache est utilisé par le "run-time linker" (chargeur de librairies durant l'éxecution d'un programme).

Consulter les man page de la commande **ldconfig** pour plus de précision sur les arguments... elle s'utilise en général de manière très simple sans argument (doit être executée en root). Cette commande doit être exécutée après l'installation de nouvelle librairies (généralement cela est effectué automatiquement lors de l'installation des packages).

Consulter egalement les man page de la commande **ldd** et **ld**

Chapitre 43. Init : Initialisation du système sous Linux

Initialisation du système sous Linux

Une fois le processus de boot terminé, votre système a besoin de lancer les daemons (ftp, nfs, gettys, ...) , c'est le rôle d'*init*

43.1. Documentation

- Guide de l'administrateur système (Traduction E. Jacoboni)
- Guide du RooTard
- BootDisk HowTo
- BootPrompt HowTo

43.2. 5 phases:

- chargement du BIOS,
- initialisation du chargeur (LInux LOader LILO),
- chargement du noyau de Linux,
- exécution du programme "init" et initialisation des périphériques,
- chargement des extensions et des services.

Nous appelerons, à l'avenir le processus d'initialisation: processus de "BOOT" (c'est plus court...)

43.3. Premières explications:

- Le BIOS détermine le premier secteur à lire (disque dur, CDROM, disquette). On dit que le BIOS détermine le disque système. Démarrer sur une disquette est utile si :
    - on désire démarrer une machine équipée avec Linux sans toucher à ce qui existe déjà sur le disque, (voir xtermkit),
    - pour une opération de maintenance ou de réparation du système,
    - pour démarrer sur un deuxième disque mirroir quand le premier est tombé en panne.
- chargement du bootstrap. Le bootstrap tient sur un secteur et contient le chargeur. Voir le cours sur Lilo.
- le chargeur active le système d'exploitation (kernel) ou noyau de Linux qui peut être n'importe où sur le disque principal où sur un autre disque. Le noyau est un "micro système" normalement compressé sur le disque. Il se décompresse automatiquement au chargement.
- Linux initialise le matériel et les périphériques puis lance le programme "init". Ce programme est dans /sbin, init utilise le fichier `/etc/inittab`
- "init" a en charge l'exécution d'autres scripts et programmes.

43.4. Le processus de BOOT

Le processus de BOOT d'un peu plus près.

- sur une disquette de Boot, il n'y a pas nécessairement de système de fichier,
- le programme du secteur de Boot lit les secteurs séquentiellement pour charger le noyau,
- si la disquette contient un SGF, on utilise un autre procédé comme LILO.
- avec des disques durs, le disque principal contient un Master Boot Record, car un disque peu avoir plusieurs partitions, chacune ayant un Partition Boot Record (PBR),
- le MBR lit la table des partitions pour déterminer la partition bootable et lit le PBR,
- le PBR est un programme qui tient sur un seul secteur, ce programme initialise le système d'exploitation. Le rôle est similaire au processus de boot d'une disquette.
- comme un disque contient un SGF, le Boot record doit accéder à des secteurs. Pour cela on utilise Lilo,
- le noyau détecte le SGF et monte ensuite le système de fichier "root".

43.5. Lilo

- charge normalement le système d'exploitation par défaut,
- peut être configuré pour:
    - charger d'autres noyaux de Linux que celui par défaut,
    - charger d'autres systèmes d'exploitation,
    - attendre au démarrage une commande de l'utilisateur.
- utiliser la combinaison de touche ALT CTRL SHIFT quand Lilo est chargé,
- les messages de la phase de démarrage sont journalisés dans /var/log,
- les messages sont consultables avec la commande "dmesg"

Voir le cours sur Lilo pour en savoir plus

43.6. Init

Si tout s'est bien passé jusque là, le noyau lance le programme "init"

Le fichier de configuration d'init `/etc/inittab` spécifie que le premier script à exécuter **/etc/init.d/rcS**. Ce script lance tous les scripts de /etc/rcS.d/ en incluant le source ou en forkant un sous−processus, selon leur extension, pour exécuter des initialisations, comme la vérification et le montage des systèmes de fichiers, le chargement des modules, le démarrage des services réseau, le réglage de l'horloge, et l'exécution d'autres initialisations. Ensuite, pour compatibilité, il lance aussi les fichiers (sauf ceux ayant un « . » dans leur nom) de /etc/rc.boot/. Les scripts de ce dernier répertoire sont habituellement réservés à l'administrateur système, et leur utilisation dans des paquets est obsolète.

- le programme init est dans /sbin
- init est chargé de lancer les daemons (ftp, nfs, gettys, ...)
- il y a plusieurs versions d'init (BSD, System V)

- BSD à ses fichiers de configuration dans /etc
- System V à ses fichiers dans un sous répertoire de /etc/rc.d
- init System V tend à devenir le standard sous Linux

43.6.1. Le répertoire /etc/rc.d

Contient le fichier `rc.sysinit` et les répertoires suivants:

- init.d
- rc0.d
- rc1.d
- rc2.d
- rc3.d
- rc4.d
- rc5.d
- rc6.d

- Le répertoire rc.d peut contenir également les fichiers rc.local, rc.serial, rc.news, ...
- le répertoire init.d contient un script par service lancé au démarrage (nfs, ftp, inet, ...)
- les répertoires rc0.d à rc6.d correspondent aux programmes qui seront chargés en fonction du niveau d'exécution de Linux que nous allons voir.

43.6.2. Séquences du programme init

- le noyau charge init,
- init exécute dans l'ordre,
    - /etc/rc.d/rc.sysinit,
    - les scripts nécessaires pour la configuration /etc/rc.d/rcx.d
    - /etc/rc.d/rc.local

Remarque

- les scripts exécutés dépendent du niveau d'exécution de Linux,
- sinon c'est une configuration par défaut qui est sélectionnée.

43.6.3. Le niveaux d'exécution (runlevels)

- 0 sert pour l'arrêt du système (ne pas mettre ce niveau par défaut)
- 1 sert pour le mode mono utilisateur
- 2 mode multi utilisateurs et réseau (sans NFS)
- 3 mode multi utilisateurs et réseau
- 4 ne sert pas
- 5 demarrer avec l'environnement graphique (proposer un environnement)
- 6 sert pour rebooter le système (ne pas mettre ce niveau par défaut)

43.6.4. Le niveau d'exécution par défaut

- par défaut le niveau 3 − mode multi utilisateurs
- défini dans /etc/inittab id:3:initdefault:

Suivant la distribution, ce niveau par défaut peut être différent

- Debian : 2
- Mandrake : 5

43.7. Le fichier `/etc/inittab`

Le fichier contient une suite d'instruction sous la forme:*code:niveau d'action:action:commande* Exemple: configurer le mode "single user" cl1:1:wait:/etc/rc.d/rc 1 Pour en savoir plus sur les niveaux d'actions respawn, wait, once, boot, ... de lancement des commandes, voir *man inittab*

Un exemple de `/etc/inittab`

```
# Le niveau d'exécution par défaut
id:2:initdefault:

# Initialisation du système
si::sysinit:/etc/rc.d/rc.sysinit

# Les différents niveaux d'exécution
l0:0:wait:/etc/rc.d/rc 0
l1:1:wait:/etc/rc.d/rc 1
l2:2:wait:/etc/rc.d/rc 2
```

```
l3:3:wait:/etc/rc.d/rc 3
l4:4:wait:/etc/rc.d/rc 4
l5:5:wait:/etc/rc.d/rc 5
l6:6:wait:/etc/rc.d/rc 6

# Intercepter les touches CTRL-ALT-DELETE
ca::ctrlaltdel:/sbin/shutdown -t3 -r now

# Démarrer en mode graphique sous xdm
x:5:respawn:/opt/kde/bin/kdm -nodaemon

# Arrêt de la machine 2 mn après le signal donné par l'UPS s'il y a une coupure d'electricité.
pf::powerfail:/sbin/shutdown -f -h +2 "Power Failure; System Shutting Down"

# Annulation de l'arrêt si l'électricité est rétablie.
pr:12345:powerokwait:/sbin/shutdown -c "Power Restored; Shutdown Cancelled"

# Création des différentes consoles (CTRL ALT F[1-6])
1:2345:respawn:/sbin/mingetty tty1
2:2345:respawn:/sbin/mingetty tty2
3:2345:respawn:/sbin/mingetty tty3
4:2345:respawn:/sbin/mingetty tty4
5:2345:respawn:/sbin/mingetty tty5
6:2345:respawn:/sbin/mingetty tty6
```

43.8. Contenu d'un répertoire rcx.d

Extrait:

```
[root@pastorius root]# ll /etc/rc2.d/
total 0
lrwxrwxrwx  1 root root 18 oct  2 15:44 S10sysklogd -> ../init.d/sysklogd*
lrwxrwxrwx  1 root root 15 oct  2 15:44 S11klogd -> ../init.d/klogd*
lrwxrwxrwx  1 root root 13 oct  2 15:44 S14ppp -> ../init.d/ppp*
lrwxrwxrwx  1 root root 15 oct  2 21:06 S15bind9 -> ../init.d/bind9*
lrwxrwxrwx  1 root root 20 oct  2 16:35 S19nfs-common -> ../init.d/nfs-common*
lrwxrwxrwx  1 root root 13 oct  2 21:06 S19nis -> ../init.d/nis*
lrwxrwxrwx  1 root root 14 oct  2 15:44 S20postfix -> ../init.d/postfix*
lrwxrwxrwx  1 root root 13 oct  6 22:28 S20gpm -> ../init.d/gpm*
lrwxrwxrwx  1 root root 15 oct  2 15:44 S20inetd -> ../init.d/inetd*
lrwxrwxrwx  1 root root 13 oct  2 16:35 S20lpd -> ../init.d/lpd*
lrwxrwxrwx  1 root root 17 oct  2 15:44 S20makedev -> ../init.d/makedev*
lrwxr-xr-x  1 root root 16 oct 11 00:21 S20pcmcia -> ../init.d/pcmcia*
lrwxrwxrwx  1 root root 13 oct  2 16:35 S20ssh -> ../init.d/ssh*
lrwxrwxrwx  1 root root 16 oct  2 21:07 S20webmin -> ../init.d/webmin*
lrwxrwxrwx  1 root root 13 oct  3 02:15 S20xfs -> ../init.d/xfs*
lrwxrwxrwx  1 root root 17 oct  5 01:37 S20xfs-xtt -> ../init.d/xfs-xtt*
lrwxrwxrwx  1 root root 25 oct  2 21:07 S25nfs-user-server -> ../init.d/nfs-user-server*
lrwxrwxrwx  1 root root 17 oct  2 20:41 S50proftpd -> ../init.d/proftpd*
lrwxrwxrwx  1 root root 13 oct  2 15:44 S89atd -> ../init.d/atd*
lrwxrwxrwx  1 root root 14 oct  2 15:44 S89cron -> ../init.d/cron*
lrwxrwxrwx  1 root root 16 oct  2 21:06 S91apache -> ../init.d/apache*
lrwxrwxrwx  1 root root 13 oct  2 16:36 S99gdm -> ../init.d/gdm*
lrwxrwxrwx  1 root root 13 oct  3 01:12 S99kdm -> ../init.d/kdm*
lrwxrwxrwx  1 root root 19 oct  2 15:44 S99rmnologin -> ../init.d/rmnologin*
lrwxrwxrwx  1 root root 13 oct  3 02:15 S99xdm -> ../init.d/xdm*
```

Ce répertoire contient des liens logiques vers des scripts qui sont dans /etc/rc.d/init.d/

43.9. Comment choisir un mode d'exécution

- Vous pouvez modifier le mode par défaut dans /etc/inittab,
- ou lors du démarrage de Linux à l'affichage du prompt "boot"
    - exemple "linux single" ou "linux 1"
    - cette méthode permet parfois le dépannage

43.10. Utilitaires de configuration

- configuration manuelle à l'aide d'un éditeur et/ou des commandes
- en environnement graphique:
    - sous KDE ou gnome
    - linuxconf,
    - le control-panel.
    - webmin https://localhost:10000/

43.11. Arrêter ou démarrer un service

En mode commande

- root:~# /etc/init.d/NomDuService stop
- root:~# /etc/init.d/NomDuService start
- root:~# /etc/init.d/NomDuService status (pour certains)
- root:~# /etc/init.d/NomDuService (vous donne l'usage)

En mode graphique

- ksysv
- contrôle center/système/services

43.12. Ajout ou suppression d'un service

- chkconfig −−level 3 smb off (sous Mandrake, RedHat)
- update−rc.d −f ntp remove (sous Debian)

43.13. Placer une commande au démarrage du système

- Si vous voulez qu'une commande soit exécutée à chaque lancement, vous pouvez utiliser le script rc.local,
- ce script est exécuté en dernier, il vous permettra d'adapter la configuration de votre système.

43.14. Arrêt du système

Procédure

- signaler à tous les processus de se terminer pour fermer proprement les fichiers,
- démonter tous les systèmes de fichiers et partitions de swap,
- message d'avertissement du système "system halted",
- couper l'alimentation

Attention

- ne pas utiliser cette procédure peut entraîner des pertes de données,
- ne pas utiliser "halt" ou "reboot" mais "shutdown",
- l'opération est réalisée sur la console système ou par planification "cron",
- limiter aux administrateurs.

43.15. La commande shutdown

Procédure

- Pour arrêter le système immédiatement
  - root:~# shutdown −h now (h pour halt)
- Pour arrêter et relancer le système
  - root:~# shutdown −r now (r pour reboot)
- Arrêter le système et prévenir les utilisateurs par un message
  - root:~# shutdown −h+10 "les services réseaux vont s'arrêter et reprendre dans 2 heures"
  - les utilisateurs auront un message sur leur console et auront le temps de quitter leur session.

43.16. La disquette de BOOT

La disquette de BOOT et Root

43.16.1. Création des disquettes

43.16.1.1. La disquette de Boot

Il existe plusieur solutions, dont l'utilitaire "mkbootdisk"

- relever la version de votre système "uname −r"
- supposons que la commande est retourné 2.4.28
- tapez root:~# mkbootdisk −−device /dev/fd0 2.4.18

43.16.1.2. La disquette de Root

Les distributions de Linux proposent souvent une disquette Root ou Rescue

- root:~# mount /mnt/cdrom /* monter le lecteur */
- root:~# cd /mnt/cdrom/images /* si les images de disquettes sont dans ce répertoire */
- tapez root:~# mkbootdisk −−device /dev/fd0 2.4.18
- tapez root:~# dd if=rescue.img of=/dev/fd0 bs 1440 /* si l'image s'appelle rescue.img */

Dans tous les cas testez les disquettes.

43.17. Dépannage

43.17.1. Mot de passe de root oublié

Le mot de passe de root est oublié ou on ne peut plus se logger

- booter avec les disquettes de Boot et de Root,
- ouvrez une session root (pas besoin de mot de passe),
- monter le disque dur exemple

```
root:~# mount −t ext2 /dev/sda1 /mnt
```

- modifiez le fichier /mnt/etc/passwd.

Cet exemple montre trois choses:

- ces disquettes peuvent être utiles,
- il ne faut pas laisser une machine critique en libre accès,
- il faut savoir utiliser vi.

43.17.2. Démarrer en "single user"

Au message boot: entrez

- boot: linux single root=/dev/hdxx initrd=initrd−2.4.18.img
- Charger le clavier suisse romand : loadkeys `/usr/lib/kbd/keymaps/i386/qwertz/fr_CH−latin1.kmap.gz`
- monter la racine :

```
root:~# mount −w −n −o remount /
```

Remarques:

- adapter /dev/hdxx au nom de votre disque
- pour le détail des options : *man mount*

43.18. Conclusion

Vous devez maintenant être en mesure:

- d'expliquer le processus de démarrage de Linux,
- modifier ce processus,
- créer une disquette de dépannage de Boot et de Root,
- dépanner un problème d'amorçage de Linux.

Chapitre 44. TP : Sytème de gestion de fichiers

Ce TP va vous guider dans la manipulation des systèmes de fichiers.

A partir d'une distribution Debian GNU/Linux

Lisez attentivement ce document une première fois sans lancer les commandes.

Lisez également les pages de manuel des commandes avec **man** et **info**.

44.1. Swap

Sur une machine en runlevel 2 ou 5, relevez les daemons qui tournent sur votre système

```
root:~# ps axf
```

relevez également les informations concernant la mémoire

```
root:~# free
```

passer en mode maintenance c'est à dire en runlevel 1

```
root:~# init 1 # ou
root:~# telinit 1
```

man : ps runlevel free init inittab

Ajout d'un fichier swap de 10MB

```
root:~# mkdir /usr/local/fs
root:~# cd /usr/local/fs
root:~# dd if=/dev/zero of=test_swap bs=512 count=208000
root:~# mkswap  test_swap
root:~# swapon test_swap -p 1
root:~# cat /proc/swaps
```

ajoutez dans /etc/fstab

```
/usr/local/fs/test_swap none swap sw,pri=1 0 0
```

au prochain reboot l'espace d'échange sera automatiquement chargé

man : dd mkswap swapon swapoff fstab

44.2. ext

ajoutez 2 partitions de 2GB chacune de type ext2 avec fdisk ou cfdisk et formatez les en type ext3

```
root:~# mkfs.ext3 /dev/hdax
root:~# mkfs.ext3 /dev/hday
root:~# mkdir /mnt/data1 /mnt/data2
```

ajoutez dans /etc/fstab

```
root:~# /dev/hdax /mnt/data1 ext3 defaults 0 2
root:~# /dev/hday /mnt/data2 ext3 defaults 0 2
```

montez les partitions

```
root:~# mount -a
```

faites une synchro vers la nouvelle home

```
root:~# rsync -av /home/ /mnt/data1/home
```

faire un backup de la home

```
root:~# mkdir /mnt/data2/backup
root:~# cd /mnt/data2/backup
root:~# tar czvf home-backup.tgz /home/
```

supprimer /home

```
root:~# rm -rf /home/
```

faire un lien symbolique vers la nouvelle home

```
root:~# ln -s /mnt/data1/home /home
```

une autre possibilite aurait été de monter /dev/hdax dans un point de montage temporaire /mnt/tmp par exemple, de faire la synchro et le backup, supprimer le contenu de /home et monter /dev/hdax sur /home que l'on aura recréé

man : mkfs fstab mount tar rsync

44.3. loop

Création d'un système de fichiers virtuel

```
root:~# mkdir /mnt/virtualfs
root:~# cd /usr/local/fs
```

Création du fichier destiné à la création du système de fichier

```
root:~# dd if=/dev/zero of=test_ext3fs bs=512 count=208000
```

Formatage

```
root:~# mkfs.ext3  test_ext3fs
```

Montage

```
root:~# mount −o loop −t ext3 −v test_ext3fs /mnt/virtualfs
```

écrire dans fstab

```
/usr/local/fs/test_ext3fs ext3 defaults 0 2
```

Copie de fichiers dans le nouveau système de fichiers

```
root:~# rsync −av /home/ /mnt/virtualfs/home
```

Affichage de la liste des fichiers

```
root:~# find /mnt/virtualfs
```

Démontage du système de fichiers virtuel

```
root:~# umount /mnt/virtualfs
```

44.3.1. Alternative permettant de choisir le device loop

```
root:~# losetup /dev/loop1  test_ext3fs
root:~# mount −v −t ext3 /dev/loop1 /mnt/virtualfs
root:~# find /mnt/virtualfs
root:~# umount /mnt/virtualfs
root:~# losetup −d /dev/loop1
```

44.3.2. loop encrypté

la même chose en crypté (n'est pas l'objet du cours)

```
root:~# losetup −e aes /dev/loop0  test_ext3fs
root:~# mkfs.ext3 /dev/loop0
root:~# mount −t ext3 −v /dev/loop0 /mnt/virtualfs
root:~# rsync −av /home/ /mnt/virtualfs/home
root:~# find /mnt/virtualfs
root:~# umount /mnt/virtualfs
root:~# losetup −d /dev/loop0
root:~# mount −t ext3 −v −o loop,encryption=aes test_ext3fs /mnt/virtualfs
```

44.3.3. loop iso9660

monter des images de CDROM et/ou DVD sans lecteur.

```
root:~# cd /usr/local/fs
```

copie de l'image du CDROM dans un fichier

```
root:~# cp /dev/cdrom image.iso
```

calcul de l'empreinte digitale de l'image du CDROM et de l'image

```
root:~# md5sum /dev/cdrom && md5sum image.iso
```

vérifiez que les deux empreintes sont identiques

```
root:~# mkdir /mnt/virtualcdrom
root:~# mount −o loop −t iso9660 −v image.iso /mnt/virtualcdrom
```

vous pouvez ensuite ajouter cette entrée dans la fstab pour avoir votre cdrom virtuel en permanence, idéal par exemple pour avoir plusieurs cd d'installation à disposition.

```
/usr/local/fs/image.iso iso9660 loop,noauto,ro,exec 0 0
```

44.3.4. Fin du TP

arrêtez la machine

```
root:~# init 0
```

redémarrez et a l'invite du boot saisissez :

```
boot: linux single ou linux 1
```

Modifier le runlevel de façon à demarrer en mode graphique en niveau 5 et non en niveau 2. Utiliser pour cela la commande**update−rc**

passer en mode multi−utilisateurs

```
root:~# init 2
```

vérifiez que vos points de montage soient correct

```
root:~# df −a
```

Chapitre 45. CVS : Concurrent Version System

CVS (Concurrent Version System) permet une simplification de la gestion de projets pour le travail en groupe, conserve l'historique de toutes les modifications effectuées sur un fichier permettant ainsi une traçabilité totale.

45.1. Présentation

CVS est un système de contrôle de versions de fichiers, il permet de conserver les modifications successives des fichiers placés sous son contrôle (généralement du code source) et de conserver l'historique des changements et de leur description. Il permet également de gérer l'édition de fichiers par plusieurs auteurs en parallèle et de gérer les conflits possibles, de déclencher des actions (mail, scripts, ...) à différents moments du cycle de vie des fichiers.

45.2. Horloge

L'horloge du serveur et toutes les machines clientes devront être synchronisées à l'aide de NTP (voir le cours sur NTP), en effet CVS se sert de l'heure et de la date pour effectuer ses opérations et cela est capital pour l'intégrité de la base CVS.

45.3. Le dépôt (repository)

Le dépôt est la base centralisée de CVS à savoir les fichiers d'administration se trouvant dans le sous dossier CVSROOT ainsi que les dossiers des différents projets (livres, développements, sites web, ...). Ce répertoire peut se trouver n'importe où sur le système de fichiers (ex: /usr/local/cvsroot) et le path pour l'atteindre doit être défini dans la variable d'environnement $CVSROOT. Afin que chaque utilisateur d'un groupe de travail puisse travailler sur un projet, le dossier ($CVSROOT/nom_projet) devra avoir les droits en lecture et écriture sur le groupe ainsi que le bit s positionné (garanti que chaque fichier/dossier cré appartient au groupe), il en sera de même pour /var/lock/cvs/nom_projet.

45.3.1. Initialisation du dépôt

Dans le cas ou vous installez CVS pour la première fois, ou que vous n'avez pas de dépôt. Une fois la variable d'environnement CVSROOT définie lancez la commande

```
mkdir /usr/local/cvsroot
cvs init
```

ou pour plus de contrôle sur la création du dépôt

```
cvs −d /usr/local/cvsroot init
chown −R cvs:cvs /usr/local/cvsroot
chmod g+rwxs /usr/local/cvsroot/CVSROOT
```

Pour les accès en mode connecté, vous devrez ensuite créer le fichier $CVSROOT/CVSROOT/passwd ayant la structure suivante:

```
login_CVS:[mot_de_passe_crypt][:login_systeme]
```

ce fichier étant particulièrement sensible il est préférable de ne pas mettre les mêmes mots de passe que ceux pour se connecter au serveur et de donner les droit suivants au fichier passwd de cvs.

```
chmod 400 $CVSROOT/CVSROOT/passwd
```

c'est un des rares cas où vous irez modifier un fichier dans $CVSROOT/CVSROOT. L'accès se fait par l'utilisation des commandes CVS.

45.3.2. Configuration

Accédez au serveur avec un utilisateur faisant partie du groupe cvs.

```
mkdir ~/Projets/
cd ~/Projets/
cvs -d /usr/local/cvsroot checkout CVSROOT
```

si la variable CVSROOT est définie l'option *-d /usr/local/cvsroot* est facultative

```
-rw-rw-r--    1 jmj      jmj           495 mai 17 01:49 checkoutlist
-rw-rw-r--    1 jmj      jmj           760 mai 17 01:49 commitinfo
-rw-rw-r--    1 jmj      jmj           986 mai 17 02:35 config
drwxr-xr-x    2 jmj      jmj          4096 mai 23 19:01 CVS/
-rw-rw-r--    1 jmj      jmj           602 mai 17 01:49 cvswrappers
-rw-rw-r--    1 jmj      jmj          1025 mai 17 01:49 editinfo
-rw-rw-r--    1 jmj      jmj          1141 mai 17 01:49 loginfo
-rw-rw-r--    1 jmj      jmj          1151 mai 17 01:49 modules
-rw-rw-r--    1 jmj      jmj           564 mai 17 01:49 notify
-rw-rw-r--    1 jmj      jmj           649 mai 17 01:49 rcsinfo
-rw-rw-r--    1 jmj      jmj           879 mai 17 01:49 taginfo
-rw-rw-r--    1 jmj      jmj          1026 mai 17 01:49 verifymsg
```

Modifiez le fichier config :

```
cd CVSROOT
vi config

# Exemple de fichier config
SystemAuth=no
LockDir=/var/lock/cvs
TopLevelAdmin=no
LogHistory=TOEFWUPCGMAR
RereadLogAfterVerify=always
```

Validez les modifications

```
 cvs commit -m "Configuration initiale de CVS" config
```

45.3.3. Accès au dépôt

L'accès à cette base CVS peut s'effectuer de 5 manières différentes:

45.3.3.1. Direct

Les fichiers dans ce cas doivent être accessible directement au travers du système de fichier ou d'un système de fichier réparti tel que NFS ou SMB. Dans ce cas nous utilisons CVS en mode non connecté.

```
CVSROOT=:local:/usr/local/cvsroot ou CVSROOT=/usr/local/cvsroot
```

45.3.3.2. Serveur

Le serveur CVS est en attente des requêtes clientes, sur le port TCP 2401. A ajouter dans /etc/services:

```
cvspserver      2401/tcp                        # CVS client/server operations
```

dans /etc/inetd.conf si vous utilisez inetd

```
cvspserver stream tcp nowait cvs /usr/bin/cvs cvs --allow-root=/usr/local/cvsroot pserver
```

pour qu'inetd prenne en compte les changements dans son fichier de configuration

```
killall -HUP inetd
```

si vous utilisez xinetd

```
# CVS configuration for xinetd don't forget to specify your CVSROOT in
#  /etc/cvs/cvs.conf.
service cvspserver
{
  disable = no
        socket_type         = stream
        protocol            = tcp
        wait                = no
        user                = root
        passenv             = PATH
        server              = /usr/sbin/cvspserver
        server_args         = -f --allow-root=/usr/local/cvsroot pserver
}
```

pour que xinetd prenne en compte les changements

```
killall −HUP xinetd
```

Sur la machine cliente définir la variable CVSROOT

```
CVSROOT=:pserver:user@server:/usr/local/cvsroot
```

L'authentification est réalisée grâce à la commande :

```
cvs login
```

qui enregistrera le mot de passe sous forme chiffrée dans le fichier .cvspass si la connexion est acceptée (pour changer le nom du fichier.cvspass, définissez le dans $CVS_PASSFILE). Pour que la connexion aboutisse, ce fichier devra également exister dans $CVROOT/passwd

L'algorithme ci−dessous explicite l'utilisation que fait pserver de ces fichier pour décider d'accorder un droit d'accès en lecture seule ou en lecture−écriture à l'utilisateur *user*.

SI *user* n'existe pas dans le fichier passwd OU son mot de passe est incorrect ALORS −> ACCES REFUSÉ

SINON SI le fichier readers existe ET *user* y figure ALORS −> ACCES LECTURE SEULE

SINON SI le fichier writers existe ET user n'y figure pas ALORS −> ACCES LECTURE SEULE SINON −> ACCES LECTURE−ÉCRITURE FINSI

45.3.3.3. Kerberos

Le serveur CVS est en attente sur le port TCP 1999 ajoutez dans /etc/services:

```
cvskserver  1999/tcp
```

dans inetd.conf

```
cvskserver stream tcp nowait cvs /usr/bin/cvs cvs −−allow−root=/usr/local/cvsroot kserver
```

Sur la machine cliente définir la variable CVSROOT

```
CVSROOT=:kserver:server:/usr/local/cvsroot
```

sur la machine cliente utilisez kinit pour obtenir un ticket kerberos vous permettant ensuite de vous connecter et d'utiliser les commandes cvs.

45.3.3.4. GSSAPI

Permet d'accéder à des systèmes sécurisés tel que kerberos 5. CVS et ses outils auront été compilés préalablement en incluant le support GSSAPI (option −−with−gssapi). Ce mode est équivalent au mode serveur et utilise également le fichier $CVSROOT/passwd. Par défaut les communications ne sont ni authentifiées ni chiffrées et il faudra utiliser des options spéciales de CVS (voir détail des commandes man et info). Sur la machine cliente définir la variable CVSROOT

```
CVSROOT=:gserver:server:/usr/local/cvsroot
```

45.3.3.5. rsh et ssh

Dans ce mode le client accède au serveur en utilisant rsh. Sur la machine cliente définir la variable CVSROOT

```
CVSROOT=:ext:user@server:/usr/local/cvsroot
```

Vérifier que la commande rsh fonctionne indépendamment de cvs.

```
rsh −l user server uname −a
```

Il faudrait configurer rsh pour ne pas demander à chaque fois le mot de passe (.rhosts ou encore host.equiv), mais cette méthode n'est pas du tout sécurisée, nous utiliserons ssh en remplacement de rsh. Définissez la variable d'environnement CVS_RSH

```
CVS_RSH=ssh
```

Vous devrez mettre votre clef publique dans ~/.ssh/authorized_keys sur le serveur pour ne plus entrer le mot de passe à chaque fois. Nous retiendrons ce mode ou shell sécurisé utilisant kerberos pour toutes communications distantes afin d'éviter d'exposer votre système à des attaques, en chiffrant les connexions et les transferts de données.

```
# depuis le poste client
$ ssh−keygen −t dsa   # PubkeyAuthentication : clé DSA pour SSH2
$ cat .ssh/id_dsa.pub | ssh user1@remote \
          "cat − >>.ssh/authorized_keys[2]"
```

45.3.4. Modules

Chaque projet que vous ajoutez dans CVS correspond à un module. Pour ajouter un module

```
mkdir /usr/local/cvsroot/nom_projet && mkdir /var/lock/cvs/nom_projet
chowm cvs:groupe_du_projet /usr/local/cvsroot/nom_projet /var/lock/cvs/nom_projet
chmod g+rwxs /usr/local/cvsroot/nom_projet /var/lock/cvs/nom_projet
```

45.4. Les commandes principales de CVS

- cvs login (pour se connecter en mode client server)
- cvs logout (pour se déconnecter en mode client server)
- cvs import −m "Liste nouveaux composants" nom_projet/HOWTO LFO V1 (ajoute les fichiers du répertoire courant avec le vendeur−tag LFO et le release−tag V1, sans avoir de copie de travail et évite de recourir aux sous−commandes add et commit pour tous les fichiers et répertoires ajoutés)
- cvs checkout nom_projet (récupère en local le module nom_projet)
- cvs update (met à jour les fichiers de la copie de travail en local)
- cvs status −v nom_fichier (visualise l'état et les noms de versions symboliques du fichier)
- cvs −n update (idem que *cvs status* mais les informations sont plus condensées)
- cvs add −m (ajout de la procédure d'installation" INSTALL, ajoute le fichier INSTALL)
- cvs remove −f nom_fichier (supprime physiquement et dans la base CVS nom_fichier), pour être validée cette commande devra être suivie d'un *cvs commit*
- cvs commit −m "première version" index.html (archive la version dans CVS)
- cvs export −k v −d version_prod −r STABLE−V1_1 nom_projet (extrait une copie du module nom_projet dans le répertoire projet_v1_1 sans les répertoires de gestion utilisés par CVS)
- cd nom_projet/srv_server & cvs tag SRV−V1_0 (défini le nom symbolique SRV−V1_0 pour tous les fichiers de nom_projet/srv_server)
- cvs co −r SRV−V1_0 nom_projet/srv_server (récupère les fichiers ayant le tag SRV−V1_0 de nom_projet/srv_server)
- cvs rtag SRV−V2_0 nom_projet (défini le nom symbolique SRV−V2_0 à la dernière version présente dans la base des fichiers du module nom_projet)
- cvs diff −−ifdef=V1_2 −r1.1 −r1.3 index.html (affiche les différences entre les versions 1.1 et 1.3 en séparant les modifications par le symbole préprocesseur V1_2)
- cvs rdiff −s −r STABLE−V1_0 nom_projet (résume les différences entre la version STABLE−V1_0 et la dernière version en base)
- cvs rdiff −u −r STABLE−V1_0 nom_projet (visualise les différences)
- cvs release −d nom_projet (vérifie que toutes vos modifications sont archivées et indique à CVS que vous n'utilisez plus votre copie de travail)

Chapitre 46. Travaux pratiques : Concurrent Version System

CVS (Concurrent Version System) permet une simplification de la gestion de projets pour le travail de groupe, conserve l'historique de toutes les modifications effectuées sur un fichier permettant ainsi une traçabilité totale.

46.1. Objectifs

- Installer et configurer CVS
- Gestion d'un projet en mode non connecté
- Gestion d'un projet en mode connecté

46.2. Installer et configurer CVS

Cette première partie vous indique comment installer et configurer un Serveur CVS

```
apt−get install cvs
```

dans /etc/profile et/ou ~/.bashrc ~/.bash_profile ajoutez

```
CVSROOT=/usr/local/cvsroot
export CVSROOT
```

création du répertoire d'archive cvs

```
mkdir /usr/local/cvsroot
```

Ajoutez un utilisateur cvs et groupe cvs sans mot de passe et shell, vous ajouterez les administrateurs cvs dans son groupe. Initialisation de l'archive (cette action ne doit s'effectuer qu'une seule fois)

```
cvs −d /usr/local/cvsroot init
chown −R cvs:cvs /usr/local/cvsroot
chmod g+rwxs /usr/local/cvsroot/CVSROOT
mkdir /var/lock/cvs/CVSROOT
chown −R cvs:cvs /var/lock/cvs/CVSROOT
```

```
chmod g+rwxs /var/lock/cvs/CVSROOT
```

Les principaux fichiers d'administration de cvs sont maintenant créés

```
ls -lR /usr/local/cvsroot/*
/usr/local/cvsroot/CVSROOT:
total 14
-r--r--r--    1 cvs     cvs          495 mai 17 01:49 checkoutlist
-r--r--r--    1 cvs     cvs          760 mai 17 01:49 commitinfo
-r--r--r--    1 cvs     cvs          991 mai 17 01:49 config
-r--r--r--    1 cvs     cvs          602 mai 17 01:49 cvswrappers
-r--r--r--    1 cvs     cvs         1025 mai 17 01:49 editinfo
drwxrwxr-x    2 cvs     cvs         4096 mai 17 01:49 Emptydir/
-rw-rw-rw-    1 cvs     cvs           68 mai 17 02:00 history
-r--r--r--    1 cvs     cvs         1141 mai 17 01:49 loginfo
-r--r--r--    1 cvs     cvs         1151 mai 17 01:49 modules
-r--r--r--    1 cvs     cvs          564 mai 17 01:49 notify
-r--r--r--    1 cvs     cvs          649 mai 17 01:49 rcsinfo
-r--r--r--    1 cvs     cvs          879 mai 17 01:49 taginfo
-rw-rw-rw-    1 cvs     cvs            0 mai 17 01:49 val-tags
-r--r--r--    1 cvs     cvs         1026 mai 17 01:49 verifymsg
```

Les fichiers d'administration ne doivent pas être édités directement dans l'archive mais en faisant un checkout du dossier CVSROOT (premier module de CVS)

```
mkdir cvs_admin && cd cvs_admin
cvs -d /usr/local/cvsroot checkout CVSROOT
cd CVSROOT
```

éditez le fichier config et prenez en connaissance.

```
# Pour faciliter les tests dans un premier temps nous mettrons l'option suivante à no
# afin que pserver ne contrôle pas users/passwords
SystemAuth=no
# Nous mettrons les fichiers de lock en dehors du repository de CVS.
LockDir=/var/lock/cvs
# Par de création au toplevel
TopLevelAdmin=no
# Toutes les transactions dans le fichier d'historique
LogHistory=TOEFWUPCGMAR
# On autorise le script verifymsg de changer le message de log.
RereadLogAfterVerify=always
```

envoyez vos modifications au serveur

```
cvs commit
```

libérez l'archive

```
cd ..
cvs release -d CVSROOT
```

46.3. Gestion d'un projet en mode non connecté

Vos projets sont connus sous le terme de module dans CVS. Création du module test

```
mkdir /usr/local/cvsroot/test
chowm user:groupe /usr/local/cvsroot/test
chmod g+rwxs /usr/local/cvsroot/test
mkdir /var/lock/cvs/test
chowm user:groupe /var/lock/cvs/test
chmod g+rwxs /var/lock/cvs/test
```

créez votre répertoire de travail

```
cd ~
mkdir -p Projets/test/
```

copiez quelques fichiers de configuration de /etc/ dans Projets/test/

```
cd Projets/test && cp /etc/host* .
```

importez votre projet

```
cvs -d /usr/local/cvsroot import -m "Création du module test" test LFO V1
cd .. && rm -rf test/ && ls -la
cvs -d /usr/local/cvsroot co test
cd test && ls -la
```

Editez le fichier hosts et apportez y quelques modifications, validez vos modifications

```
cvs commit
```

```
cvs status
```

dans un terminal connectez−vous avec un utilisateur différent

```
mkdir −p Projets
cvs −d /usr/local/cvsroot co test
```

ajoutez des fichiers au projet test

```
cp /etc/aliases . && cp /etc/fstab .
cvs add aliases fstab
cvs commit −m "ajout du fichier aliases et fstab" aliases fstab
```

liberez votre archive

```
cd ..
cvs release −d test
```

retournez sur le compte utilisateur initial

```
cd Projets/test
cvs −n update
cvs update
```

liberez votre archive

```
cd ..
cvs release −d test
```

46.4. Gestion d'un projet en mode connecté

Définissez maintenant les variables suivantes (server étant le nom ou l'adresse ip d'une machine distante et user votre compte sur cette dernière) votre clef publique ssh devra se trouver dans le fichier authorized_keys de la machine distante, afin de ne pas devoir ressaisir le mot de passe à chaque action de cvs. pour plus de détails voir le cours sur ssh, ainsi que les pages de manuel.

```
CVSROOT=:ext:user@server:/usr/local/cvsroot
CVS_RSH=ssh
```

ajouter dans /etc/services:

```
cvspserver      2401/tcp
```

dans /etc/inetd.conf

```
cvspserver stream tcp nowait cvs /usr/bin/cvs cvs −−allow−root=/usr/local/cvsroot pserver
```

forcez inetd à prendre en compte les modifications

```
killall −HUP inetd
```

récupérez le module test du serveur distant

```
cd Projets
cvs co test
```

Chapitre 47. L'annuaire LDAP

Un annuaire électronique est une solution permettant la création d'une collection d'objets. Dans les réseaux, les annuaires permettent à l'échelle d'une entreprise de déclarer tous les objets (utilisateurs, applications, équipements matériels...), et pour chaque objet de définir ses propriétés (attributs).

Cela permet d'avoir un recensement de tous les objets dans une base de données. Cette base étant, le plus souvent répartie. Novell, Microsoft utilisent des bases de données d'annuaires (respectivement les NDS Netware Directory Services et AD Active Directory) pour la manipulation des ressources sur leurs réseaux.

Les annuaires sont ensuites accessibles à partir de tous types d'applications (outlook, mozilla, konqueror...), mais aussi par les processus d'identification/authentification, les processus systèmes...

47.1. Introduction

LDAP (Light Directory Access Protocol) est un service d'annuaire dérivé de la norme X.500. La norme X.500 est très lourde, LDAP en est une version allégée ("light") dans un sens absolument pas péjoratif.

Vous trouverez de bien meilleures descriptions du principe, concept et du protocole LDAP en suivant les références indiquées à la fin de ce document.

Un serveur LDAP permet de centraliser des informations très diverses. Il offre de nombreux avantages :

- un serveur d'annuaire (recensement de tous les objets d'un système) : c'est la fonction la plus connue, on peut trouver des serveurs LDAP chez bigfoot, netscape (netcentrer), infoseek et bien d'autres ;
- Information sur les utilisateurs (nom, prénom...), et données d'authentification pour les utilisateurs : cela permet aussi la définition de droits.
- Information pour les applications clientes et fonctions de serveur d'accès itinérant : cela permet de stocker ses informations personnelles sur un serveur et de les récupérer lors de la connexion;
- bien d'autres choses...

LDAP supporte le chiffrement SSL et cohabite parfaitement avec les applications Samba, DNS, NFS... ce qui permet son utilisation pour des applications comme les serveurs de liste de diffusion(sympa par exemple).

L'objet de cette séquence sera de voir comment installer, configurer puis administrer un serveur LDAP. Nous utiliserons la distribution OpenLDAP disponible sur les distributions Linux

47.2. Présentation de LDAP

LDAP fournit un ensemble d'outils.

1. un protocole permettant d'accéder à l'information contenue dans l'annuaire,
2. un modèle d'information définissant l'organisation et le type des données contenues dans l'annuaire,
3. un modèle de nommage définissant comment l'information est organisée et référencée
4. un modèle fonctionnel qui définit comment accéder à l'information,
5. un modèle de sécurité qui définit comment accéder aux données et comment celles−ci sont protégées. OpenLDAP est souvent configuré avec SASL (Simple Authentication and Security Layer), qui permet les transactions cryptées avec les protocoles fonctionnant en mode connecté.
6. un modèle de duplication qui définit comment la base est répartie entre serveurs,
7. des APIs pour développer des applications clientes,
8. LDIF, (Ldap Data Interchange Format) un format d'échange de données.

47.2.1. Le protocole

Un protocole d'accès aux données, qui décrit comment ajouter, modifier, supprimer des données dans la base de donnée, quels protocoles de chiffrement (kerberos, ssl...), et quels mécanismes d'authentification sont utilisés. Ce protocole est utilisé dans la relation client/serveur, mais également entre serveurs (serveur/serveur) car une base de données LDAP peut être répartie.

47.2.2. Le modèle de données

47.2.2.1. Le Directory Information Tree

LDAP fournit un modèle d'organisation des données. Ces données sont organisées sous forme hiérarchique. L'arbre est nommé Directory Information Tree (DIT). Le sommet (racine), contient le "suffixe". Chaque noeud représente une "entrée" ou "Directory Entry Service" (DSE). Les données sont stockées sur un format de base de données hiérarchique de type "dbm". Ce format est différent des bases de données relationnelles, conçues pour supporter de multiples mises à jour. DBM est conçu pour supporter peu de mises à jour, mais de nombreuses consultations.

Figure 47−1. LDAP : le DIT Directory Information Tree

47.2.2.2. Classes d'objets, objets, attributs et schéma

Une entrée (DSE) dnas le DIT correspond à un objet abstrait (organisation, ressource) ou concret (personne, équipement...). Les objets possèdent une description dans une "classe d'objet". Une classe d'objet donne une représentation modélisée des objets qu'elle représente en caractérisant tous les attributs des objets.

Certaines classes dobjet ont fait l'objet d'une normalisation et sont réutilisables. Elles sont définies par un nom, un OID (Object Identifier), la liste des attributs (facultatifs ou obligatoires), et, pour chaque attribut, un type. Le type est lié à la nature (essentiellement texte ou binaire) des attributs utilisés.

Une classe d'objet est définie par un nom, un OID (Object IDentifier), la liste des attributs (facultatifs et obligatoires), un type. Le type est lié à la nature des attributs utilisés.

Chaque objet est composé d'attributs en fonction des types d'attributs décrits dans les classes d'objets. Un attribut est généralement un couple clé/valeur, mais peut être caractérisé par un nom, un OID, s'il est mono ou multi−évalué, un indicateur d'usage (facultatif/obligatoire), un format (voir par exemple pour les images).

Les OID sont normalisés par la RFC2256 et sont issus d'un schéma X500. Les OID sont tenus à jour par l'IANA Internet Assigned Numbers Authority. Un OID est une séquence de chiffres séparés par un "." point (Exemple 1.2.3.4) qui permet d'identifier de façon unique un élément du schéma LDAP.

Exemple de la la classe inetOrgPerson (dépend de organizationalPerson)

```
# The inetOrgPerson represents people who are associated with an
# organization in some way.  It is a structural class and is derived
# from the organizationalPerson which is defined in X.521 [X521].
Objectclass     ( 2.16.840.1.113730.3.2.2  <-------------- OID
  NAME 'inetOrgPerson'
     DESC 'RFC2798: Internet Organizational Person'
     SUP organizationalPerson
       STRUCTURAL
       MAY (
                audio $ businessCategory $ carLicense $ departmentNumber $
                displayName $ employeeNumber $ employeeType $ givenName $
                homePhone $ homePostalAddress $ initials $ jpegPhoto $
                labeledURI $ mail $ manager $ mobile $ o $ pager $
                photo $ roomNumber $ secretary $ uid $ userCertificate $
                x500uniqueIdentifier $ preferredLanguage $
                userSMIMECertificate $ userPKCS12 )
        )
```

```
et l'attibut employeeType
# employeeType
# Used to identify the employer to employee relationship.  Typical values
# used will be "Contractor", "Employee", "Intern", "Temp", "External", and
# "Unknown" but any value may be used.
Attributetype ( 2.16.840.1.113730.3.1.4
        NAME 'employeeType'
        DESC 'RFC2798: type of employment for a person'
        EQUALITY caseIgnoreMatch
        SUBSTR caseIgnoreSubstringsMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )
```

```
Signification des mots réservés :

DESC :  Description
SUP :           Objet parent
MUST :  Attributs requis
MAY :   Attributs possibles
```

Chaque objet de l'annuaire qui correspond à une classe d'objet est décrit par des valeurs en fonction des attributs qui décrivent la classe d'objets. Un attribut est généralement un couple clé/valeur, mais peut être caractérisé par un nom, un OID, indique s'il est mono ou multi−évalué, donne un indicateur d'usage (facultatif/obligatoire), impose un format (par exemple base64 pour les images, utf−8 pour les données).

Les objets sont rattachés obligatoirement à au moins une classe d'objet. Une classe d'objet caractérise ou modélise les objets qui lui seront rattachés. On en déduit qu'un objet ne pourra pas avoir d'attribut non déclaré dans la classe d'objet. Par contre dans la classe d'objet, un attribut pourra être soit optionnel, soit obligatoire. L'administrateur a déjà des classes d'objets prédéfinies, il a la possibilité d'en définir d'autres.

Chaque objet hérite des attributs des objets dont il hérite. Quand on décrit un objet "fils", on doit décrire tous les liens de parenté avec l'attribut "ObjetClass". Les objets forment une hiérarchie, avec, au sommet, l'objet "top". Exemple pour enrichir l'objet "person" des attributs "technicalPerson".

```
objectClass: top
objectClass: myOrganization
objectClass: person
objectClass: technicalPerson
```

Figure 47−2. LDAP : Héritage

L'ensemble de la description des classes d'objet et des types d'attributs définissent le "schéma".

Une entrée peut appartenir à plusieurs classes d'objets. Les attributs de l'objets sont la réunion des attributs de toutes les classes d'objets :

```
Entry Type        Required Attributes   Optional Attributes
person            commonName (cn)       mail
                  surName (sn)          mobile
                  objectClass           ...

OrganizationUnit  ou                    description
                  objectClass           localisation
                                        ...
Organization      o                     description
```

```
                onjectClass           ...
```

Les OID sont normalisés par la RFC2256 et sont issus d'un schéma X55. Les OID sont tenus à jour par l'IANA Internet Assigned Numbers Authority. Un OID est une séquence de chiffres séparés par un "." point. Exemple 1.2.3.4

Le "dn", ou Distinguished Name, est le chemin absolu de l'entrée dans le DIT à partir de la racine. Par exemple :

```
dc=org, dc=mydomaine, ou=person, uid=toor
```

On peut utiliser aussi un nommage "relatif" par rapport à une position courante. Dans ce cas on utilise le RDN (Relative Distinguished Name).

47.2.2.3. Le format d'échange de donnée LDIF

Le format d'échange permet l'import/export de données des bases, mais sert également pour l'ajout ou la modification. Les données sont en ASCII codées en UTF−8, sauf pour le binaire qui est codé en base64 (images par exemple).

Les fichiers au format LDIF respectent une structure de description des objets et des commandes :

```
Syntaxe générale :
    dn: <distinguished name
    objectClass: <object class
    objectClass: <object class
    ...
    <attribute type:<attribute value
    <attribute type:<attribute value
    ...

Exemple :
    dn: cn= Manon Des Sources, ou= compta, dc=mydomain, dc=org
    objectClass: person
    objectClass: organization
    cn: AN GROSSI
    sn: GROSSI
    givenName: AM
    userPassword: {sha}KDIE3AL9DK
    uid: amg
```

Les fichiers supportent également des commandes pour ajouter, modifier ou supprimer une entrée.

```
    dn: distinguished name
    changetype <identifier
    change operation identifier
    list of attributes...
    ...
    −
    change operation identifier
    list of attributes
    ...
    <identifier :
        add (ajouter une entrée,
        delete (suppression),
        modrdn (modification du RDN),
        modify (modification : add, replace, delete d'un attribut)
```

On utilise le caractère "−" pour séparer 2 instructions. Par exemple :

```
     dn: cn= Morina Fuentes, ou=admin, dc=mydomain, dc=org
     changetype: modify
     add: telephonenumber
     telephonenumber: 05 55 55 55 55
     −
     add: manager
     manager: cn= toor root, ou=admin, dc=mydomain, dc=org
```

47.2.3. Les méthodes d'accès

Les opérations de base sont résumées ici :

```
Opération
Search             recherche dans l'annuaire d'objets
Compare            comparaison du contenu de deux objets
Add                ajout d'une entrée
Modify             modification du contenu d'une entrée
Delete             suppression d'un objet
Rename (Modify DN) modification du DN d'une entrée
Bind               connexion au serveur
Unbind             deconnexion
```

Les requêtes de type "search" ou "compare" reçoivent des paramètres.

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

Le scope (Profondeur de recherche)

Le scope définit la profondeur de la recherche dans l'arbre des données. La figure montre la portée d'une recherche ou d'une comparaison en fonction du paramètre scope.

```
search scope=base recherche uniquement dans l'entrée définie
search scope=one recherche dans l'entrée définie et le premier sous−niveau
search scope = subtree, cherche dans toute la sous−arborescence.
```

Figure 47−3. LDAP : ls scope

search scope=base recherche uniquement dans l'entrée définie

search scope=onelever search, cherche dans tous les noeuds fils rattachés directement au noeud courant

search scope = subtree, cherche dans toute la sous−arborescence.

Les URL LDAP offrent aux client web un accès aux bases de données :

```
ldap[s]://<hostname>:<port>/<base_dn>?\
          <attributes>?<scope>?<filter>

          <base_dn>   : DN de l'entrée qui est le
                        point de départ de la recherche
          <attributes>: les attributs que l'on veut consulter
          <scope>     : la profondeur de recherche dans le
                        DIT à partir du
          <base_dn>   :"base" | "one" | "sub"
          <filter>    : filtre de recherche,
                        par défaut (objectClass=*)

Exemples :
    ldap://ldap.netscape.com/ou=Sales,o=Netscape,c=US?cn,\
            tel,mail?scope=sub?(objetclass=person)
    ldap://ldap.point−libre.org/cn=Manon,ou=Contact,o=point−libre.org
```

47.2.4. Le langage de commande

1. − slapadd, slapcat, slapindex, slappasswd, fournis avec les serveurs LDAP (utilisables sous le compte root).
2. − ldapadd, ldapdelete, ldapmodify, ldapmodrdn, ldappasswd, ldapsearch, fournis avec les utilitaires ldap (utilisable par les utilisateurs et les applications). Ils sont fournis par le paquet (ldap−utils)

Exemple :

```
ldapsearch  −x −h localhost −b "dc=mydomain,dc=fr" "objectclass=*"
```

Recherche de tous les objets sur l'annuaire de la machine locale, à partir de la racine. (−b indique à partir de quel niveau la recherche doit être exécutée). L'option « −x » indique de ne pas utiliser la méthode d'authentification SASL si elle n'est pas activée.

```
ldapdelete 'cn=Jean Colombani,cn=mydomain,cn=fr'
```

Suppression d'une entrée dans l'annuaire

```
ldapadd −f /tmp/unFichierAuFormatLDIF
```

Ajout dans l'annuaire à partir d'un fichier contenant des données au format LDIF.

Note: Pour éviter d'avoir à préciser à chaque fois certains paramètres (machine, port, annuaire...) il est possible de configurer le fichier de configuration « ldap.conf » qui sera utilisé par les applications clientes.

47.3. Concevoir un annuaire

47.3.1. Déterminer les besoins, les données, le schéma

La première opération est une phase d'analyse qui va permettre de déterminer l'organisation du système. Le résultat sera souvent représentable sous une forme arborescente. Cette étape permet de déterminer quel "schéma" sera nécessaire pour le contexte.

LDAP fournit par défaut une hiérarchie de classe déjà complète. Ces classes font l'objet d'une normalisation. Il sera nécessaire d'en prendre connaissance, puis, ensuite de procéder aux extensions nécessaires pour l'étude en cours.

47.4. Créer une base de données

Créer une base de données LDAP, va consister donc d'abord à créer

- une country (c)
- une ou plusieurs organisation (o) à 1 ou plusieurs niveaux
- définir les objets (feuilles de l'arbre)

Il est possible d'importer les informations à partir d'un fichier texte dans la base de données grâce à un format d'échange de données, le format LDIF (Lightweight Data Interchange Format).

47.5. Installer, configurer et Administrer LDAP

Vous allez avoir schématiquement 4 étapes :

1. Installer les packages nécessaires
2. configurer les fichiers de configuration
3. initialiser la base de données
4. configurer un client et tester le ou les services à partir d'un client
5. passer à l'administration

47.5.1. Installer les packages sur le serveur

Il va falloir installer les packages OpenLDAP. La procédure est standard à partir de paquets (debian ou rpm), sinon à partir des sources.

```
tar xzvf openldap-x.y.z.tgz
cd openldap.x.y.z
./configure
make
make depend
make test
make install
```

Installez également les packages perl, php et "dévelop" pour l'administration.

```
ii  slapd          2.1.30-2       OpenLDAP server (slapd)
ii  ldap-utils     2.1.30-2       OpenLDAP utilities
ii  libldap2       2.1.30-2       OpenLDAP libraries
ii  php4-ldap      4.3.8-1        LDAP module for php4
```

Il est intéressant de prévoir aussi les bibliothèques de développement.

47.5.2. Les fichiers de configuration du serveur

Il faut bien identifier les objets à référencer et les objectifs de l'annuaire. Les fichiers sont dans "/etc/openldap".

Le premier fichier est "sldapd.conf" qui décrit les principaux paramètres de votre annuaire :

```
# $OpenLDAP: pkg/ldap/servers/slapd/slapd.conf,v 1.8.8.6 2001/04/20 23:32:43 kurt Exp $
#
# See slapd.conf(5) for details on configuration options.
# This file should NOT be world readable.
#
# Inclusion des schémas nécessaires
include         /etc/openldap/schema/core.schema
include         /etc/openldap/schema/cosine.schema
include         /etc/openldap/schema/inetorgperson.schema
include         /etc/openldap/schema/nis.schema

# Options que vous pouvez modifier
#pidfile        //var/run/slapd.pid
#argsfile       //var/run/slapd.args

#######################################################################
```

```
# ldbm database definitions
########################################################################
# Choix du format de base de données pour le stockage des informations.
database        ldbm

# Configurer le suffixe (racine) de l'annuaire
# en fonction du domaine  DNS
suffix          "dc=my-domain,dc=com"
# ou d'une autre organisation
#suffix         "o=My Organization Name,c=US"

# L'administrateur de l'annuaire
rootdn          "cn=Manager,dc=my-domain,dc=com"

# Le mot de passe de l'administrateur, préférer une option cryptée
# La commande htpasswd peut très bien faire l'affaire pour encrypter

# rootpw                secret
# rootpw                {crypt}ijFYNcSNctBYg


# Emplacement de la base de données
directory       /var/lib/ldap

# Création des index.
# Comme pour une base de données, indexer les rubriques
# les plus utilisées.

index   objectClass,uid,uidNumber,gidNumber,memberUid   eq
index   cn,mail,surname,givenname                       eq,subinitial

# La réplication ne sera  pas utilisée ici
# Vous pouvez répliquer tout ou partie d'un arbre
# activée par le daemon slurpd

# Directives de replication
# sinon les mettre dans un fichier à part et utiliser
# replogfile /chemincomplet/du/fichier

# Indiquer quels sont les serveurs réplicats
# et la méthode d'authentification
# Ici le serveur local, se répliquera sur ldap1

#replica host:ldap-1.example.com:389
#       bindmethod=simple
#       binddn="cn=replicat_slave1, dc=mydomain, dc=fr"
#       credential=UnMotDePasse

# Accés par défaut sur la base
defaultaccess read
```

Protéger ensuite le fichier avec un "chmod 600 /etc/openldap/slapd.conf".

Les autorisations d'accès nécessitent une remarque.

Ici l'accès par défaut est "read", mais il est possible d'affiner. Par exemple avec des règles d'écriture comme:

```
access to <what> [ by <who> <none | compare | search | read | write>]
# Donne un accés en écriture pour le manager du domaine
access to * by dn="cn=Manager,dc=mydomain,dc=fr" write
# Donne un accés en lecture à tout le monde sur la base
access to * by * read
# Donne un accés en écriture sur un attribut pour le manager
#                en lecture pour les autres.
access to attr=uid
                 by dn="manager,dc=mydomain,dc=fr" write
                 by * none
```

Le nombre d'options est très important, utilisez la commande "**man slapd.conf**".

47.6. Ressources

LDAP HOWTO

Chapitre 48. TP 1− Installer, configurer et Administrer LDAP

Le but de ce TP est de découvrir une première mise en oeuvre du service d'annuaire LDAP.

Vous allez avoir schématiquement 4 étapes :

1. Installer les packages nécessaires
2. configurer les fichiers de configuration

3. initialiser la base de données
4. configurer un client et tester le ou les services à partir d'un client
5. passer à l'administration

48.1. Installer les packages sur le serveur

Il va falloir installer les packages OpenLDAP. La procédure est standard à partir de paquets (debian ou rpm), sinon à partir des sources.

```
tar xzvf openldap-x.y.z.tgz cd openldap.x.y.z
./configure
make
make depend
make test
make install
```

Installez également les packages perl, php et "dévelop" pour l'administration.

```
$> rpm -qa | grep ldap

mod_auth_ldap-1.6.0-7mdk
libldap2-devel-2.0.25-7mdk
libldap2-2.0.25-7mdk
openldap-2.0.25-7mdk
nss_ldap-194-3mdk
libldap2-devel-static-2.0.25-7mdk
openldap-back_ldap-2.0.25-7mdk
openldap-back_sql-2.0.25-7mdk
pam_ldap-148-3mdk
php-ldap-4.2.3-1mdk
perl-ldap-0.26-2mdk
openldap-clients-2.0.25-7mdk
openldap-servers-2.0.25-7mdk
openldap-migration-2.0.25-7mdk
openldap-back_passwd-2.0.25-7mdk
openldap-guide-2.0.25-7mdk
openldap-back_dnssrv-2.0.25-7mdk
```

Il est intéressant de prévoir aussi les bibliothèques de développement.

48.2. Les fichiers de configuration du serveur

Il faut bien identifier les objets à référencer et les objectifs de l'annuaire. Les fichiers sont dans "/etc/openldap".

Le premier fichier est "sldapd.conf" qui décrit les principaux paramètres de votre annuaire :

```
# $OpenLDAP: pkg/ldap/servers/slapd/slapd.conf,v 1.8.8.6 2001/04/20 23:32:43 kurt Exp $
#
# See slapd.conf(5) for details on configuration options.
# This file should NOT be world readable.
#
# Inclusion des schémas nécessaires
include         /etc/openldap/schema/core.schema
include         /etc/openldap/schema/cosine.schema
include         /etc/openldap/schema/inetorgperson.schema
include         /etc/openldap/schema/nis.schema

# Options que vous pouvez modifier
#pidfile        //var/run/slapd.pid
#argsfile       //var/run/slapd.args

########################################################################
# ldbm database definitions
########################################################################
# Choix du format de base de données pour le stockage des informations.
database        ldbm

# Configurer le suffixe (racine) de l'annuaire
# en fonction du domaine  DNS
suffix          "dc=my-domain,dc=com"
# ou d'une autre organisation
#suffix         "o=My Organization Name,c=US"

# L'administrateur de l'annuaire
rootdn          "cn=Manager,dc=my-domain,dc=com"

# Le mot de passe de l'administrateur, préférer une option cryptée
# La commande htpasswd peut très bien faire l'affaire pour encrypter

# rootpw                secret
```

```
# rootpw                {crypt}ijFYNcSNctBYg


# Emplacement de la base de données
directory       /var/lib/ldap

# Création des index.
# Comme pour une base de données, indexer les rubriques
# les plus utilisées.

index   objectClass,uid,uidNumber,gidNumber,memberUid   eq
index   cn,mail,surname,givenname                       eq,subinitial

# La réplication ne sera  pas utilisée ici
# Vous pouvez répliquer tout ou partie d'un arbre
# activée par le daemon slurpd

# Directives de replication
# sinon les mettre dans un fichier à part et utiliser
# replogfile /chemincomplet/du/fichier

# Indiquer quels sont les serveurs réplicats
# et la méthode d'authentification
# Ici le serveur local, se répliquera sur ldap1

#replica host:ldap-1.example.com:389
#       bindmethod=simple
#       binddn="cn=replicat_slave1, dc=mydomain, dc=fr"
#       credential=UnMotDePasse

# Accés par défaut sur la base
defaultaccess read
```

Protéger ensuite le fichier avec un "**chmod 600** `/etc/openldap/slapd.conf`".

Les autorisations d'accès nécessitent une remarque.

Ici l'accès par défaut est "read", mais il est possible d'affiner. Par exemple avec des règles d'écriture comme:

```
access to <what> [ by <who> <none | compare | search | read | write>]
# Donne un accés en écriture pour le manager du domaine
access to * by dn="cn=Manager,dc=mydomain,dc=fr" write
# Donne un accés en lecture à tout le monde sur la base
access to * by * read
# Donne un accés en écriture sur un attribut pour le manager
#                en lecture pour les autres.
access to attr=uid
                  by dn="manager,dc=mydomain,dc=fr" write
                  by * none
```

Pour configurer un réplicat, procédez comme suit :

```
# Configurez le fichier slrupd.conf du réplicat
# Ajoutez les options updatedn
# Le dn replicat_slave1 doit être identique à celui déclaré sur le maître.
rootdn ="cn=replicat_slave1, dc=mydomain, dc=fr"
rootpw=UnMotDePasse
updatedn = "cn=replicat_slave1, dc=mydomain, dc=fr"
```

Le nombre d'options est très important, utilisez la commande "man slapd.conf".

Chapitre 49. Installation d'un annuaire LDAP et utilisation du langage de commande

49.1. Environnement

Relevez l'emplacement des fichiers des configuration et des schémas sur votre machine.

Le fichier ldap.conf et le fichier de configuration des applications clientes.

Le fichier slapd.conf et le fichier de configuration du serveur.

Le service serveur se nomme slapd.

Vérifier l'installation des schémas prédéfinis (généralement dans "/etc/shema" ou dans "/usr/share/openldap/schema/")

Faites une sauvegarde de tous vos fichiers de configuration.

Dans le répertoire "schema", sont référencés des définitions normalisées utilisables avec slapd. Les définitions, et ordre de définitions doivent respecter des règles (héritage des objets).

Notez de quels fichiers héritent les définitions données dans le fichiers inetorgperson.schema

Notez de quelles classes d'objets hérite la classe inetorgperson Identifier les propriétés des classes dont dépend la classe inetorgperson. Notez celles qui sont obligatoires.

49.1.1. Configuration du fichier slapd.conf

Modifier le fichier slapd.conf en adaptant l'exemple ci−dessous :

```
Include /usr/share/openldap/schema/cosine.schema
include /usr/share/openldap/schema/corba.schema
include /usr/share/openldap/schema/inetorgperson.schema
include /usr/share/openldap/schema/java.schema
include /usr/share/openldap/schema/krb5−kdc.schema
include /usr/share/openldap/schema/kerberosobject.schema
include /usr/share/openldap/schema/misc.schema
include /usr/share/openldap/schema/nis.schema
include /usr/share/openldap/schema/openldap.schema
include /usr/share/openldap/schema/autofs.schema
include /usr/share/openldap/schema/samba.schema
include /usr/share/openldap/schema/kolab.schema

# Inclusion des schémas utilisateurs
include /etc/openldap/schema/local.schema

# Définition des ACLs
include         /etc/openldap/slapd.access.conf



pidfile         /var/run/ldap/slapd.pid

argsfile        /var/run/ldap/slapd.args

modulepath      /usr/lib/openldap

# Support de TLS, il faut créer un certificat dans /etc/ssl/openldap/ldap.pem
# et décommenter les lignes ci−dessous

#TLSCertificateFile      /etc/ssl/openldap/ldap.pem
#TLSCertificateKeyFile   /etc/ssl/openldap/ldap.pem
#TLSCACertificateFile    /etc/ssl/openldap/ldap.pem

# Journalisation
loglevel 256

#######################################################
# database definitions
#######################################################

database        bdb
suffix          "dc=foo,dc=org"
rootdn          "cn=Manager,dc=foo,dc=org"

# Mot de passe d'accès à la racine de l'arbre.
rootpw          secret
# rootpw        {crypt}ijFYNcSNctBYg

# Emplacement de la base de  la base de données
# The database directory MUST exist prior to running slapd AND
# should only be accessable by the slapd/tools. Mode 700 recommended.
directory       /var/lib/ldap

# Création des index sur la base
index   objectClass,uid,uidNumber,gidNumber     eq
index   cn,mail,surname,givenname               eq,subinitial

# Contrôle d'accès aux données
# Basic ACL (deprecated in favour of ACLs in /etc/openldap/slapd.access.conf)
access to attr=userPassword
        by self write
        by anonymous auth
        by dn="cn=Manager,dc=foo,dc=org" write
        by * none

access to *
        by dn="cn=manager,dc=foo,dc=org" write
        by * read
```

Lancement du serveur

```
# /etc/init.d/slapd start
Starting OpenLDAP: slapd.

# ps aux | grep slapd
root 17433 0.1 1.6 7780 2144 ?    S 12:54 0:00 /usr/sbin/slapd
root 17434 0.0 1.6 7780 2144 ?    S 12:54 0:00 /usr/sbin/slapd
root 17435 0.0 1.6 7780 2144 ?    S 12:54 0:00 /usr/sbin/slapd
root 17437 0.0 0.5 1716 716 pts/3 R 12:54 0:00 grep slapd

# netstat −natup | grep LISTEN
tcp 0  0 0.0.0.0:389  0.0.0.0:*  LISTEN 17433/slapd
tcp 0  0 0.0.0.0:22   0.0.0.0:*  LISTEN   290/sshd
```

Le serveur fonctionne bien.

Identifiez à l'aide de la commande netstat, le port sur lequel « écoute » par défaut le service ldap.

49.1.2. Création de l'annuaire

Vous allez utiliser un fichier au format ldif afin de créer l'arbre initial et déclarer l'administrateur. La base de données sera créée dans /var/lib/ldap. En cas d'erreur d'interprétation du fichier ldif, vous devrez relancer la commande. Pour cela :

```
− arrêtez le serveur ldap,
− supprimez la base de données
  (pas le répertoire, mais le contenu du répertoire),
− relancez le serveur, puis le script corrigé.
[ #] /etc/init.d/slapd stop
[ #] rm −rf /var/lib/ldap/*
[ #] /etc/init.d/slapd start
```

Pour le fichier LDIF, vous pouvez voir ou utiliser les annexes pour l'exemple de fichier LDIF utilisable.

49.1.3. Création de l'annuaire

Créer un fichier "base.ldiff" basé sur celui ci :

```
dn: dc=foo, dc=org
objectClass: organization
dc: foo
o: foo
telephoneNumber: 05 05 05 05 05
street: cours du General de Gaulle
postalCode: 87000
postalAddress: Limoges
l: Limoges
st: Haute−Vienne

dn: cn=manager, dc=foo, dc=org
objectClass: top
objectClass: person
userPassword: secret
cn: admin
sn: Administrateur
```

La création de l'annuaire et l'ajout peut se faire, sous le compte root, avec les commandes :

```
slapadd < leFICHIER.diff
```

Vérifiez que le serveur a bien enregistré les premières informations à l'aide la commande "slapcat".

Note : il faut parfois redémarrer le service serveur pour qu'il tienne compte des modifications (ajouts).

49.1.4. Enrichissement de l'annuaire

Récupérez le jeu d'essai et ajoutez les entrées dans l'annuaire.

```
slapadd < le_FICHIER_LDIFF
```

Vérifier le résultat à l'aide de la commande slapcat.

49.1.5. Le langage de commande

OpenLDAP fournit un langage et des API pour accéder à la base de données à partir de tous les langages (perl, php, python, c...) mais aussi du shell.

Vous allez utiliser les commandes ldapsearch, ldapadd, ldapmodify

Modifier le fichier ldap.conf de façon à ne pas avoir à préciser à chaque fois les paramètres sur la ligne de commande (machine, et base de l'annuaire)

```
BASE    dc=foo, dc=org
HOST    127.0.0.1
Ainsi une ligne de commande :
ldapsearch −x −h localhost −b 'dc=foo, dc=org'   'objectclass=person' sn cn
devient
ldapsearch  −x 'objectclass=person' sn cn
```

Exemple d'utilisation de la commande ldapsearch

```
ldapsearch  −x  'objectclass=*' # On veut tous les objets
ldapsearch  −x  'objectclass=person' sn cn # On récupère les cn et sn
ldapsearch −x  'uid=mlx' mail # On veut le mail de la personnes d'uid mlx
```

Trouver les numéros de téléphone de toutes les personnes.

Donner les caractéristiques de la personne faisant partie de l'ou "user" et ayant pour uid "mlx".

Ajout d'un objet (mode interactif) avec ldapadd.

Il faut créer un fichier au format LDIF pour chaque objet que l'on souhaite ajouter. Voici un exemple :

```
dn: cn=be good, ou=user, dc=foo,dc=org
objectclass: top
objectclass: person
objectclass: organizationalPerson
objectclass: inetOrgPerson
objectclass: posixAccount
cn: be good
sn: Bourg
givenName: good
uid: begood
uidNumber:1010
gidNumber:1010
homedirectory:/home/begood
loginshell:/bin/bash
userpassword:{crypt}2/yajBmqc3tYw
mail: begood@foo.org
telephoneNumber: 00−00−00−00−00
street: none
postalCode: 87400
postaladdress: Chabant
preferredLanguage: fr
```

Seul un compte ayant le droit d'écrire "write" peut ajouter des occurrences. Ici vous n'utiliserez ces commandes avec un compte système standard (pas le compte root)

```
$ ldapadd −x −D "cn=manager,dc=foo,dc=org" −w secret −f  leFICHIER.ldif
ou alors
$ ldapadd −x −D "cn=manager, dc=foo, dc=org" −W −f le_FICHIER_ldiff
```

mais vous devrez saisir le mot de passe.

Chapitre 50. L'annuaire LDAP

50.1. Authentification système LDAP sur un système GNU/Linux

Faites une sauvegarde de vos fichiers de configuration.

```
root@uranus:/etc/ldap# cp ldap.conf ldap.conf.orig
root@uranus:/etc/ldap# cp slapd.conf slapd.conf.orig
root@uranus:/etc# cp nsswitch.conf nsswitch.conf.orig
root@uranus:/etc# cp libnss−ldap.conf libnss−ldap.conf.orig
```

Vérifier que votre serveur LDAP fonctionne.

Vérifier que votre serveur Annuaire est opérationnel.

50.1.1. Configuration de l'environnement pour l'authentification système

Les fichiers /etc/libnss−ldap.conf et /etc/pam_ldap.conf sont normalement configurés. Sinon vous pouvez utiliser les commandes :

```
dpkg−reconfigure libnss−ldap
```

```
dpkg-reconfigure libpam-ldap
```

Prenez les options par défaut en prenant soin de tenir compte de la structure de votre annuaire. Pour nous :

```
dc=point-libre,dc=org
```

Vous devez également modifier le fichier /etc/nssiwtch.conf afin que les applications utilisent aussi bien les fichiers standards (passwd, shadow) que l'annuaire ldap. :

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

50.1.2. Premiers tests de l'annuaire

La commande **getent passwd** doit vous permettre de récupérer à la fois les comptes du fichier /etc/passwd, mais également les compte de l'annuaire ldap qui n'existent pas dans la base passwd.

```
# getent passwd
[...]
mlx:2/yajBmqc3tYw:1005:1005:BOURG Jean:/home/mlx:/bin/bash
mly:2/yajBmqc3tYw:1006:1006:BOURG Marine:/home/mly:/bin/bash
```

Ces deux comptes proviennent bien de l'annuaire ldap.

50.1.3. Vérification du fonctionnement de l'annuaire.

L'utilisateur mly n'a pas de compte système. Il n'existe que dans l'annuaire ldap. Créez un répertoire et affectez-le au compte.

```
# mkdir /home/mly
# chown 1006:100 /home/mly
# ls -al /home/mly
drwxr-sr-x    2 mly      users        4096 2003-06-09 13:50 .
# Ici le serveur fonctionne car il substitue bien l'UID à l'uidNumber

# ls -al /home/mly
# Ici le serveur LDAP n'a pas été trouvé ou il ne fonctionne pas.
drwxr-sr-x    2 1006     users        4096 2003-06-09 13:50 .
```

50.1.4. Mise en place de l'authentification

On va mettre en place maintenant l'authentification ldap. Il faut modifier les fichiers de configuration qui assurent l'authentification. Mettez en début de fichier :

```
#/etc/pam.d/login
auth    sufficient pam_ldap.so
account sufficient pam_ldap.so
password        required        pam_ldap.so

#/etc/pam.d/passwd
auth    sufficient pam_ldap.so
account sufficient pam_ldap.so
password        sufficient        pam_ldap.so
```

Vérification de l'authentification :

```
mlx@uranus:~$ su mly
Password: #ici taper toto
mly@uranus:~$ cd
mly@uranus:~$ pwd
/home/mly
```

Chapitre 51. L'annuaire LDAP avec PHP

51.1. Utilisation de PHP avec LDAP

Faites une sauvegarde de vos fichiers de configuration.

Vérifier que vos serveurs HTTP et LDAP fonctionnent.

Vérifier que votre serveur Annuaire est opérationnel.

51.1.1. Les principales fonctions

```
$conn=ldap_connect(hote[,port])
```

établir une connexion avec un serveur LDAP, retourne un entier positif en cas de succès, FALSE en cas d'erreur

```
$dn = "cn=admin, dc=foo, dc=org"; $mdp= "secret"
$cr=ldap_bind($conn, $dn, $mdp) //connexion authentifiée
$cr=ldap_bind($conn) //connexion anonyme
```

Retourne TRUE ou FALSE.

```
$result=ldap_search($conn,$dn,$filtre)
```

Recherche sur le serveur LDAP avec le filtre $filtre et retourne un identifiant de résultat, ou bien FALSE en cas d'erreur.

```
$n=ldap_count_entries($conn, $result)
```

Retourne le nombre d'entrées trouvées ou FALSE en cas d'erreur.

```
$info=ldap_get_entries($conn, $result)
```

Retourne un tableau associatif multi−dimensionnel ou FALSE en cas d'erreur. Structure du tableau associatif :

```
$info["count"] = nombre d'entrées dans le résultat
$info[0] : sous−tableau renfermant les infos de la première entrée
$info[n]["dn"] : dn de la n−ième entrée du résultat
$info[n]["count"] : nombre d'attributs de la n−ième entrée
$info[n][m] : m−ième attribut de la n−ième entrée
```

info[n]["attribut"]["count"] : nombre de valeur de cet attribut pour la n−ième entrée

$info[n]["attribut"][m] : m−ième valeur de l'attribut pour la n−ième entrée

```
$r=ldap_add($conn, $dn, $info)
```

$dn est l'identification complète de l'entrée à ajouter, et $info un tableau contenant les valeurs des attributs

```
ldap_modify($conn, $dn, $info)
```

modifie l'entrée identifiée par $dn, avec les valeurs fournies dans le tableau $info.

51.1.2. Accès anonyme pour une recherche

Le script ci−dessous permet d'exécuter une recherche. Testez ce script avec la commande "php4 ./LeNomDuScript", puis à partir d'une requête http (via mozilla ou konqueror par exemple)

```
<?
$baseDN = "dc=foo,dc=org";
$ldapServer = "localhost";
$ldapServerPort = 389;
$mdp="secret";
$dn = 'cn=manager,dc=foo,dc=org';

echo "Connexion au serveur <br />";
$conn=ldap_connect($ldapServer);

// on teste : le serveur LDAP est−il trouvé ?
if ($conn)
 echo "Le résultat de connexion est ".$conn ."<br />";
else
 die("connexion impossible au serveur LDAP");

/* 2ème étape : on effectue une liaison au serveur, ici de type "anonyme"
 * pour une recherche permise par un accès en lecture seule */

// On dit qu'on utilise LDAP V3, sinon la V2 par défaut est utilisé
// et le bind ne passe pas.
if (ldap_set_option($conn, LDAP_OPT_PROTOCOL_VERSION, 3)) {
    echo "Utilisation de LDAPv3 \n";
 } else {
    echo "Impossible d'utiliser LDAP V3\n";
    exit;  }

$bindServerLDAP=ldap_bind($conn);


print ("Liaison au serveur : ". ldap_error($conn)."\n");
```

```
// en cas de succès de la liaison, renvoie Vrai
if ($bindServerLDAP)
  echo "Le résultat de connexion est $bindServerLDAP <br />";
else
  die("Liaison impossible au serveur ldap ...");

/* 3ème étape : on effectue une recherche anonyme, avec le dn de base,
par exemple, sur tous les noms commençant par B */

echo "Recherche suivant le filtre (sn=B*) <br />";
$query = "sn=B*";
$result=ldap_search($conn, $baseDN, $query);
echo "Le résultat de la recherche est $result <br />";

echo "Le nombre d'entrées retournées est "
echo ldap_count_entries($conn,$result)."<p />";
echo "Lecture de ces entrées ....<p />";
$info = ldap_get_entries($conn, $result);
echo "Données pour ".$info["count"]." entrées:<p />";

for ($i=0; $i < $info["count"]; $i++) {
        echo "dn est : ". $info[$i]["cn"] ."<br />";
        echo "premiere entree cn : ". $info[$i]["cn"][0] ."<br />";
        echo "premier email : ". $info[$i]["mail"][0] ."<p />";
}
/* 4ème étape : clôture de la session  */
echo "Fermeture de la connexion";
ldap_close($conn);
?>
```

51.1.3. Accès authentifié pour une recherche

Modifiez le script de façon à obtenir une connexion authentifiée (manager/secret) et retournez la valeur du "sn" de l'admin.

Construisez un formulaire html qui doit vous permettre de vous connecter en saisissant un compte utilisateur et un mot de passe sur l'annuaire LDAP via HTTP.

Adapter le script afin qu'il réalise l'authentification sur l'annuaire.

Vous retournerez les informations en cas de succés, un message d'erreur en cas d'échec.

Chapitre 52. Annexes à la séquence sur LDAP

52.1. Exemple de fichier LDIF

Exemple de fichier utilisable

```
# Penser à recoder en UTF-8
# avec recode ou iconv
dn: dc=point-libre,dc=org
objectclass: organization
objectclass: dcObject
o: point-libre.org
dc: point-libre.org
description: Annuaire de test
postalCode: 87000


dn: ou=ressource,dc=point-libre,dc=org
objectclass: top
objectclass: organizationalUnit
ou: ressource
description: Branche ressource

dn: ou=user,dc=point-libre,dc=org
objectclass: top
objectclass: organizationalUnit
ou: user
description: Branche utilisateurs


dn: ou=agenda,dc=point-libre,dc=org
objectclass: top
objectclass: organizationalUnit
ou: agenda
description: Branche contact


dn: ou=people,dc=point-libre,dc=org
objectclass: top
objectclass: organizationalUnit
ou: people
```

```
description: Branche personne

dn: ou=service,dc=point-libre,dc=org
objectclass: top
objectclass: organizationalUnit
ou: service
description: Branche service

dn: ou=etudiant,ou=people,dc=point-libre,dc=org
objectclass: top
objectclass: organizationalUnit
ou:etudiant
description:Branche etudiant

dn: ou=personnel,ou=people,dc=point-libre,dc=org
objectclass: top
objectclass: organizationalUnit
ou: personnel
description: Branche personnel

dn: ou=nfs,ou=service,dc=point-libre,dc=org
objectclass: top
objectclass: organizationalUnit
ou:nfs
description:nfs

dn: ou=groupe,ou=service,dc=point-libre,dc=org
objectclass: top
objectclass: organizationalUnit
ou:groupe
description: Groupes

dn: cn=manager,ou=ressource,dc=point-libre,dc=org
objectclass: top
objectclass: organizationalrole
cn:manager

#Ajout d'un contact

dn: cn=Aline BOURG, ou=agenda, dc=point-libre,dc=org
objectclass: top
objectclass: person
objectclass: inetOrgPerson
cn: BOURG Aline
sn: Bourg
givenName: Aline
mail: bingo@point-libre.org
telephoneNumber: 00-00-00-00-00
street: none
postalCode: 87400
postaladdress: Chabant
preferredLanguage: fr

#Ajout d'un utilisateur

dn: cn=Aline BOURG, ou=user, dc=point-libre,dc=org
objectclass: top
objectclass: person
objectclass: inetOrgPerson
objectclass: posixAccount
cn: BOURG Aline
sn: Bourg
givenName: Aline
uid: mlx
uidNumber:1005
gidNumber:1005
homedirectory:/home/mlx
loginshell:/bin/bash
userpassword:{crypt}2/yajBmqc3tYw
mail: bingo@point-libre.org
telephoneNumber: 00-00-00-00-00
street: none
postalCode: 87400
postaladdress: Chabant
preferredLanguage: fr

dn: cn=Marine BOURG, ou=user, dc=point-libre,dc=org
objectclass: top
objectclass: person
objectclass: inetOrgPerson
objectclass: posixAccount
cn: BOURG Marine
sn: Bourg
givenName: Marine
uid: mly
uidNumber:1006
gidNumber:1006
homedirectory:/home/mly
loginshell:/bin/bash
```

```
userpassword:{crypt}2/yajBmqc3tYw
mail: bingo@point-libre.org
telephoneNumber: 00-00-00-00-00
street: none
postalCode: 87400
postaladdress: Chabant
preferredLanguage: fr
```

Création de l'annuaire, ajout de données et test.

```
# slapadd -l init.ldiff
# slapcat

dn: dc=point-libre,dc=org
objectClass: organization
objectClass: dcObject
o: point-libre.org
dc: point-libre.org
description: Annuaire de test
postalCode: 87000

dn: ou=ressource,dc=point-libre,dc=org
objectClass: top
objectClass: organizationalUnit
ou: ressource
description: Branche ressource

dn: ou=user,dc=point-libre,dc=org
objectClass: top
objectClass: organizationalUnit
ou: user
description: Branche utilisateurs

dn: ou=agenda,dc=point-libre,dc=org
objectClass: top
objectClass: organizationalUnit
ou: agenda
description: Branche contact

dn: ou=people,dc=point-libre,dc=org
objectClass: top
objectClass: organizationalUnit
ou: people
description: Branche personne

dn: ou=service,dc=point-libre,dc=org
objectClass: top
objectClass: organizationalUnit
ou: service
description: Branche service

dn: ou=etudiant,ou=people,dc=point-libre,dc=org
objectClass: top
objectClass: organizationalUnit
ou: etudiant
description: Branche etudiant

dn: ou=personnel,ou=people,dc=point-libre,dc=org
objectClass: top
objectClass: organizationalUnit
ou: personnel
description: Branche personnel

dn: ou=nfs,ou=service,dc=point-libre,dc=org
objectClass: top
objectClass: organizationalUnit
ou: nfs
description:: bmZzICA=

dn: ou=groupe,ou=service,dc=point-libre,dc=org
objectClass: top
objectClass: organizationalUnit
ou: groupe
description:: R3JvdXBlcyA=

dn: cn=manager,ou=ressource,dc=point-libre,dc=org
objectClass: top
objectClass: organizationalrole
cn: manager

dn: cn=Jean BOURG, ou=agenda, dc=point-libre,dc=org
objectClass: top
objectClass: person
objectClass: inetOrgPerson
cn: BOURG Jean
sn: Bourg
givenName: Jean
mail: bingo@point-libre.org
```

```
telephoneNumber: 00-00-00-00-00
street: none
postalCode: 87400
postalAddress: Chabant
preferredLanguage: fr

dn: cn=Jean BOURG, ou=user, dc=point-libre,dc=org
objectClass: top
objectClass: person
objectClass: inetOrgPerson
objectClass: posixAccount
cn: BOURG Jean
sn: Bourg
givenName: Jean
uid: mlx
uidNumber: 1005
gidNumber: 1005
homeDirectory: /home/mlx
loginShell: /bin/bash
userPassword:: e2NyeXB0fTIveWFqQm1xYzN0WXc=
mail: bingo@point-libre.org
telephoneNumber: 00-00-00-00-00
street: none
postalCode: 87400
postalAddress: Chabant
preferredLanguage: fr


dn: cn=Marine BOURG, ou=user, dc=point-libre,dc=org
objectClass: top
objectClass: person
objectClass: inetOrgPerson
objectClass: posixAccount
cn: BOURG Marine
sn: Bourg
givenName: Marine
uid: mly
uidNumber: 1006
gidNumber: 1006
homeDirectory: /home/mly
loginShell: /bin/bash
userPassword:: e2NyeXB0fTIveWFqQm1xYzN0WXc=
mail: bingo@point-libre.org
telephoneNumber: 00-00-00-00-00
street: none
postalCode: 87400
postalAddress: Chabant
preferredLanguage: fr
```

52.2. L'annuaire ldap vu de konqueror

Figure 52-1. LDAP sous konqueror

52.3. L'annuaire ldap vu de gq

Figure 52-2. LDAP sous gq

52.4. Le schéma vu de gq

Figure 52-3. Schéma LDAP sous gq

52.5. Authentification avec php sur LDAP

```
<?
$baseDN = "dc=foo,dc=org";
$ldapServer = "localhost";
$ldapServerPort = 389;
$rdn="admin";
$mdp="secret";
$dn = 'cn=manager,dc=foo,dc=org';

echo "Connexion au serveur <br />";
```

```
$conn=ldap_connect($ldapServer);

// on teste : le serveur LDAP est-il trouvé ?
if ($conn)
 echo "Le résultat de connexion est ".$conn ."<br />";
else
 die("connexion impossible au serveur LDAP");



/* 2ème étape : on effectue une liaison au serveur, ici de type "anonyme"
 * pour une recherche permise par un accès en lecture seule
 */

// On dit qu'on utilise LDAP V3, sinon la V2 par défaut est utilisé
// et le bind ne passe pas.
if (ldap_set_option($conn, LDAP_OPT_PROTOCOL_VERSION, 3)) {
    echo "Utilisation de LDAPv3 \n";
 } else {
    echo "Impossible d'utiliser LDAP V3\n";
    exit;
 }


// Instruction de liaison.
// Décommenter la ligne pour une connexion authentifiée
// ou pour une connexion anonyme.
// Connexion authentifiée
// print ("Connexion authentifiée ...<br />");
// $bindServerLDAP=ldap_bind($conn,$dn,$mdp);
// print ("Connexion anonyme...<br />");
// $bindServerLDAP=ldap_bind($conn);

print ("Liaison au serveur : ". ldap_error($conn)."\n");
// en cas de succès de la liaison, renvoie Vrai
if ($bindServerLDAP)
  echo "Le résultat de connexion est $bindServerLDAP <br />";
else
  die("Liaison impossible au serveur ldap ...");

/* 3ème étape : on effectue une recherche anonyme, avec le dn de base,
 * par exemple, sur tous les noms commençant par B
 */
echo "Recherche suivant le filtre (sn=B*) <br />";
$query = "sn=B*";
$result=ldap_search($conn, $baseDN, $query);
echo "Le résultat de la recherche est $result <br />";

echo "Le nombre d'entrées retournées est ".ldap_count_entries($conn,$result)."<p />";
echo "Lecture de ces entrées ....<p />";
$info = ldap_get_entries($conn, $result);
echo "Données pour ".$info["count"]." entrées:<p />";

for ($i=0; $i alt; $info["count"]; $i++) {
        echo "dn est : ". $info[$i]["cn"] ."<br />";
        echo "premiere entree cn : ". $info[$i]["cn"][0] ."<br />";
        echo "premier email : ". $info[$i]["mail"][0] ."<p />";
}
/* 4ème étape : cloture de la session  */
echo "Fermeture de la connexion";
ldap_close($conn);

>
```

Chapitre 53. Planification prévisionnelle des séquences LDAP

53.1. Prévision des séquences

1. Première fiche de cours (fait)
2. Install d'un serveur LDAP et authentification système (fait)
3. Installation d'un secondaire (en cours)
4. Développement via les lib php, c, php et/ou perl (en cours)
5. Intgration de l'authentification Samba ou d'autres services (en cours)
6. Rajouter une note sur les différences de notation X500 et ldap dans le cours (en cours)

Chapitre 54. Synchroniser ses machines avec NTP

54.1. Introduction à ntpdate et ntpd

NTP (Network Time Protocol) permet de synchroniser sa machine avec des serveurs via le reseau et de garder sa machine à l'heure. Nous allons voir deux methodes pour se synchroniser: ntpdate et ntpd. Ces deux outils peuvent s'utiliser de manière complementaire ou bien de manière indépendante.

54.2. ntpdate

ntpdate est un des moyens de synchroniser la machine. Il va synchroniser quelle que soit la différence d'horaire avec le serveur.

54.2.1. Installation de ntpdate

```
apt−get install ntpdate
```

dpkg demande un ou plusieurs serveurs NTP. Vous pouvez entrer les suivants par exemple:

```
ntp.metas.ch
swisstime.ethz.ch
chronos.cru.fr
ntp.univ−lyon1.fr
```

Confirmez l'enregistrement du fichier de configuration de ntp.

Il est possible d'utiliser ntpdate directement sur la ligne de commande avec par exemple:

```
# ntpdate ntp.metas.ch
31 May 19:07:54 ntpdate[3085]: step time server 193.5.216.14 offset −7192.143171 sec
```

Votre machine est maintenant à l'heure du serveur ntp.metas.ch

54.2.2. Configuration de ntpdate

Par defaut l'installation avec apt−get configure pour executer ntpdate à chaque démarrage. On peut d'ailleurs le vérifier facilement:

```
# ls /etc/rcS.d/ | grep ntpdate
S51ntpdate
```

Il est possible de verifier ce que dpkg a ajouté dans le fichier de configuration /etc/defaults/ntp−servers. Normalement vous devriez avoir ceci:

```
NTPSERVERS="ntp.metas.ch"
```

Désormais, au démarrage de la machine, ntpdate se lancera et synchronisera la machine.

54.3. ntpd

ntp et ntp−simple font tourner un daemon pour synchroniser en permanence la machine et ntp−doc contient toute la documentation au format HTML de NTP

54.3.1. Installation de ntpd

```
apt−get install ntp ntp−doc ntp−simple
```

dpkg detecte que vous avez déjà un fichier /etc/default/ntp−servers et propose de modifier votre configuration:

choisissez no pour pourvoir modifier vous−même le fichier de configuration.

les fichiers principaux se trouvent aux endroits suivants:

- /etc/ntp.conf

  Fichier de configuration de NTP, c'est là qu'on indique les serveurs avec lesquels ntpd doit se synchroniser.
- /usr/share/doc/ntp−doc/html/index.htm

  La documentation ntp au format HTML
- /usr/sbin/ntpd

  L'executable principal
- /etc/init.d/ntp

  Le script d'initialisation de ntp

54.3.2. Configuration de ntpd

Il suffit d'éditer le fichier /etc/ntp.conf (qui n'existe pas pour l'instant) et d'y mettre une liste de serveurs NTP valides.

Utiliser plusieurs serveurs devrait augmenter la précision et éviter les problemes si un des serveurs ne répond pas.

```
server ntp.metas.ch
server swisstime.ethz.ch
server chronos.cru.fr
server ntp.univ-lyon1.fr
```

Il suffit maintenant de sauvegarder et relancer ntpd avec la commande:

```
/etc/init.d/ntp restart
```

54.4. Conclusion

On le voit ces deux outils se completent: ntpdate "remet les pendules à l'heure" au boot alors que ntpd va synchroniser la machine durant l'uptime, de maniere tres précise et par petits règlages. .Il est parfaitement possible d'utiliser ntpdate en crontab afin de synchroniser la machine toutes les heures par exemple, et dans ce cas ntpd n'est plus du tout indispensable cependant il faut garder à l'esprit qu'il est bien plus précis (encore qu'à ce degré là vous ne verrez probablement pas la différence).

54.5. Liens utiles

- Une liste de serveurs http://www.eecis.udel.edu/~mills/ntp/servers.html
- Tout ce qu'il y a à savoir sur le NTP http://www.eecis.udel.edu/~mills/ntp.html

Chapitre 55. Eléments de cours sur le routage et le filtrage de paquets IP

Le document présente des éléments sur le filtrage de paquets par le noyau linux.

55.1. Routage, Filtrage sur les paquets IP

Les paquets ip sont routés en fonction de leur adresse de destination et de leur adresse d'émission. Ils utilisent un protocole de transport (UDP ou TCP). La session est identifiée par un port source et par un port de destination.

Une connexion (session au sens OSI du terme) entre un processus client et un processus serveur est matérialisée par le couple de triplets :

```
(@IP-source, TCP/UDP, port source) , (@IP-dest., TCP/UDP, portdest.)
```

Une session nous permet de déterminer quelle application (service serveur) est sollicitée.

Figure 55-1. Squelette de trame IP

Ce sont les principaux éléments que nous utiliseront dans le cadre du routage et du filtrage sur un réseau IP. Le schéma nous montre qu'il sera possible de filtrer sur les adresses (hôtes ou réseaux), les protocoles de transport ou les applications en filtrant sur les ports.

La capture de trame ci-dessous, indique bien les éléments présents dans une trame.

Figure 55-2. Capture de trame sur le port 80

L'analyse de traffic va consister à traiter les paquets en fonction d'un certain nombres de critères.

55.2. Technique de masquage et de translation d'adresse

Le terme "translation" est la reprise du terme Anglais. Les documentations en Français tendent à utiliser plutôt le terme de "traduction" dont la signification est plus proche de ce que fait réellement le noyau (et de la traduction française du mode anglais "translation"). Une adresse est ainsi "traduite en une autre".

Cette technique est utilisée pour sécuriser un réseau en entrée, limiter les sorties ou pour partager une ou plusieurs adresses publiques à un réseau privé (pénurie d'adresses publiques).

Dans la traduction d'adresse ou le masquage (ip masquerade), le routeur modifie dans le paquet l'adresse source (SNAT ou ip masquerade) ou l'adresse de destination (DNAT). La modification est réalisée lors de l'envoi et du retour du paquet.

Il y a bien deux fonctions différentes : le routage et le filtrage. Le filtrage va consister à appliquer des règles supplémentaires aux paquets qui sont routés.

Ces deux fonctions sont prises en charge par le noyau Linux. Elles sont prises en charges par "ipchains" pour les noyaux 2.2.x et par "netfilter" pour les noyaux 2.4.

Le filtrage consiste à mettre en place des règles qui seront appliquées aux paquets. Pour netfilter, on utilise la commande "iptables" qui permet de mettre en place ou modifier des chaînes de règles. Les chaînes sont des ensemble de règles qui sont appliquées séquentiellement aux paquets jusqu'à ce que l'une d'entre elles soit applicable. Il y a toujours au moins une chaîne par défaut. Ces chaînes, sont placées dans des tables.

Il y a 3 tables principales pour netfilter (filter, nat et mangle).

La table "filter" est la table par défaut qui contient les règles de filtrage.

La table "nat" contient les règles pour faire de la traduction d'adresse.

La table "mangle" contient les règles qui permettent de modifier les paquets ip, par exemple le champ TOS.

Dans la table filter, il y a 3 principales chaînes : (INPUT, OUTPUT, FORWARD).

La chaîne "INPUT" contient les règles appliquées aux paquets entrants qui sont généralement destinées aux processus locaux.

La chaîne "OUTPUT" contient les règles appliquées aux paquets sortants qui sont généralement émis par les processus locaux.

La chaîne "FORWARD" contient les règles appliquées aux paquets qui traversent.

Figure 55−3. Routage pris en charge par le noyau

Le routeur est une "boîte noire". Chaque paquet entrant, quelque soit l'interface d'entrée est "routé" par le noyau. Les paquets qui ne font que traverser le routeur Linux, sont concernés par la chaîne FORWARD. Ceux qui sont destinés aux processus internes, c'est à dire qui entrent, sont concernés par la chaîne INPUT, ceux émis par les processus internes, c'est à dire qui sortent, par la chaîne OUTPUT.

55.3. Masquerading et Forwarding

Le forwarding est une fonction qui permet de router les paquets entre deux réseaux.

Le masquerading est le fait de permettre aux machines de votre réseau interne de pouvoir sortir sur votre réseau externe en utilisant une seule adresse IP officielle. Cette adresse officielle est mise à la place de l'adresse IP de votre machine cliente et re−remplacée au retour du paquet. Dans le cas du masquerading, les machines internes ne peuvent pas être atteintes sans règles supplémentaires par une machine de l'extérieur.

Les techniques de masquerading sont vues dans les parties applications sur ipchains et iptables.

Chapitre 56. ICMP

Le protocole ICMP

56.1. ICMP et le filtrage de paquets

Les réseaux fonctionnant en IP ont besoin de s'envoyer des messages de contrôles, de temps à autre. Par exemple, cela permet de de dire qu'un hôte est inaccessible ou encore que le paquet n'est pas arrivé à destination faute de destination trop lointaine. Ces messages sont regroupés dans le protocole ICMP, Internet Control Message Protocol, rfc 792.

Ce protocole peut facilement se filtrer avec les firewalls, il est donc important de savoir quels messages on doit/veut bloquer et ceux que l'on veut voir traverser notre pare−feu.

Chaque paquet ICMP a un code et un type. Ceux−ci établissent une correspondance exacte sur le "but " du paquet :

```
type 0 Réponse Echo
code 0 = réponse à une demande d'echo

type 3 Destination non accessible
code 0 = réseau inaccessible
code 1 = hôte inaccessible
code 2 = protocole non disponible
```

```
code 3 = port non accessible
code 4 = fragmentation nécessaire mais interdite
code 5 = échec d'acheminement source

type 4 Contrôle de flux
code 0 = ralentir le flux de l'émission

type 5 Redirection
code 0 = redirection de datagramme sur la base du réseau
code 1 = redirection de datagramme sur la base de l'adresse d'hôte
code 2 = redirection de datagramme sur la base du réseau et du Type de Service
code 3 = redirection de datagramme sur la base de l'hôte et du Type de Service


type 8 Echo
code 0 = envoi d'un echo


type 11 Durée de vie écoulée
code 0 = durée de vie écoulée avant arrivée à destination
code 1 = temps limite de réassemblage du fragment dépassé


type 12 Erreur de Paramètre
code 0 = l'erreur est indiquée par le pointeur

type 13 Marqueur temporel
code 0 = envoi d'une étiquette temporelle d'émission

type 14 Réponse à marqueur temporel
code 0 = envoi d'une étiquette temporelle de réception et de transmission


type 15 Demande d'information
code 0 = demande du numéro de réseau

type 16 Réponse à demande d'information

code 0 = réponse du numéro de réseau
```

Chapitre 57. Ipchains

Résumé des commandes et exemples Ipchains.

57.1. Langage d'Ipchains

Ipchains est utilisable avec les anciennes versions de netfilter sur les versions des noyaux 2.2.

D'après la page de man d'ipchains :

```
ipchains −[ADC] chain rule−specification [options]
ipchains −[RI] chain rulenum rule−specification [options]
ipchains −D chain rulenum [options]
ipchains −[LFZNX] [chain] [options]
ipchains −P chain target [options]
ipchains −M [ −L | −S ] [options]
```

Ipchains est utilisé pour configurer, maintenir et surveiller les règles du firewall IP du noyau Linux. Ces règles sont stockées dans quatre catégories de chaînes différentes, la chaîne input, la chaîne output, la chaîne forward, les chaînes utilisateurs. Pour chacune de ces catégories une table de règles propre à la chaîne est maintenue.

target : une règle de firewall spécifie un ensemble de critères à appliquer pour un paquet et une destination. Si le paquet ne peut être traité, la règle suivante est examinée, si le paquet peut être traité, alors la règle est appliquée. Cette règle peut être :

```
* JUMP RegleUtilisateur, le paquet sera traité par une règle d'une chaîne utilisateur,
* une des valeurs ACCEPT, DENY, REJECT, MASQ, REDIRECT, RETURN.
* ACCEPT laisse le paquet traverser la chaîne,
* DENY jette purement et simplement le paquet,
* MASQ est utilisé pour masquer les adresses sources des paquets.
```

Commandes : ces options spécifient les actions à réaliser. Il est possible d'en spécifier **qu'une seule à la fois** sur la ligne de commande.

```
−A, −−append Ajoute une règle à la fin de la chaîne.
−D, −−delete Supprime une ou n règles dans une chaîne.
−I, −−insert Insère une ou plusieurs règles dans la chaîne mentionnée.
−L, −−list Donne une liste des règles en cours d'application.
−Z, −−zero Initialise un compteur de paquet et de bytes des règles.
−p, −−protocol[!] protocol Spécifie le protocole à traiter dans la règle.
      tcp, udp, icmp, all.
−s, −−source [!] address[/mask] [!] [port[:port]] permet de définir la source.
```

```
      Exemple −s 192.168.1.0/24 port 80 signifie tous les paquets adressés
      sur le port 80 et en provenance des machines du réseau d'adresse 192.168.1.0.
      Si le port est précisé, la règle doit préciser à quel protocole elle s'applique
      (icmp, tcp, udp, all).
−d, −−destination [!] address[/mask] [!] [port[:port]]
      Même remarque que pour le paramètre "−−source" mais appliqué à la destination.
−j, −−jump target  utilisé dans une règle pour définir une autre chaîne
      de traitement du paquet (chaîne utilisateur) ou pour indiquer le traitement
      à réaliser.
−i, −−interface [!] name Paramètre optionnel.
      Permet de spécifier une règle qui va traiter les paquets entrants
      ou à destination d'une interface réseau.
```

Attention à l'écriture. Les cibles (target) sont sensibles à la casse.

Exemples d'écriture :

```
Voir toutes les règles actives :
    ipchains −L ou ipchains −n −L (mode numérique sans résolution de nom)

Supprimer les règles du firewall et fermer maintenant toutes les portes
avec les commandes :

ipchains −A input −p all −j DENY ou ipchains −A input −j DENY
ipchains −A output −p all −j DENY
ipchains −A forward −p all −j DENY
```

La première chaîne signifie que pour tous les protocoles qui entrent les paquets seront détruits. Pour modifier les règles par défaut des chaînes input, output ou forward, utiliser −P

```
ipchains −P forward −j DENY
#par défaut les paquets seront refusés sur la chaîne forward.

; Supprime la première règle de la chaîne input
ipchains −D input 1

; Refuser tous les paquets sur la chaîne input
ipchains −I input −j DENY ou alors ipchains −P input DENY

; Crée une nouvelle chaîne utilisateur "machaine"
ipchains −N machaine

; Ajoute à la chaîne "machaine" une autorisation pour tous les paquets
; de type HTTP à destination de la machine 192.168.90.1
ipchains −A machaine −s 0.0.0.0/0 www −d 192.168.90.1 −j ACCEPT

; Redirige tous les paquets sur le port 8080 (service mandataire)
ipchains −A input  −p tcp −s 192.168.1.1/24 −d 0.0.0.0/0 80 −j redirect 8080

; Accepter les requêtes DNS sur la machine 192.168.1.1
ipchains −A input −p udp −s 0/0 dns −d 192.168.1.1/24 −j ACCEPT

; Les deux règles suivantes permettent de se protéger de
; l'ip−spoofing si eth0 est l'interface du réseau privé.
ipchains −A input −i eth0 −s ! 192.168.1.0/255.255.255.0 −j DENY
ipchains −A input  −i ! eth0  −s 192.168.1.0/255.255.255.0 −j DENY

−− Configuration du Firewall − masquage  :

; Ouvrez toutes les portes du firewall à l'aide des commandes suivantes :
ipchains −A input −j ACCEPT
ipchains −A output −j ACCEPT
ipchains −A forward −j ACCEPT
# Pensez à vérifiez l'état des chaînes à l'aide de la commande" ipchains −n −L"

; Activer l'ip masquerading à l'aide de la commande :
ipchains −A forward −p all −s 192.168.1.0/24 −d 0.0.0.0/0 −j MASQ

; Cela signifie qu'il faudra masquer (effectuer une translation
; d'adresse −j MASQ) pour tous les paquets qui proviennent (−s source)
; du réseau 192.168.1.0 et vont à destination de n'importe quelle adresse.

; Autres exemples commentés de règles.
; Autoriser la circulation des protocoles FTP, HTTP et refuser tous les autres.
ipchains −A output −p tcp −s 192.168.1.0/24 −d 0.0.0.0/0  80 −j ACCEPT
# pour les requêtes HTTP
ipchains −A output −p tcp −s 192.168.1.0/24 −d 0.0.0.0/0  21 −j ACCEPT
# pour les requêtes FTP
ipchains −A output −p tcp −s 192.168.1.0/24 −d 0.0.0.0/0  20 −j ACCEPT
# pour les requêtes FTP−DATA
ipchains −A output −p tcp −s 192.168.1.0/24 −d 0.0.0.0/0  113 −j
; ACCEPT
# pour les requêtes AUTH
ipchains −A output −s 192.168.1.0/24 −d 0.0.0.0/0 −j DENY

; Le port tcp/80 permet de laisser sortir les requêtes HTTP,
; Le port tcp/21 permet d'ouvrir une session sur un hôte distant,
```

```
; Le port tcp/20 permet l'échange de donnée (put/get),
; Le port tcp/113 permet l'authentification.
; On refuse d'envoyer tout le reste.


; Interdire toutes les requêtes qui sortent vers l'extérieur,
; sauf celles qui vont sur un port 80 (requêtes http),
; Ici, on utilise les noms de protcoles définis dans /ets/services
ipchains −A input −p tcp −d 0.0.0.0/0 ! www −j DENY

; Créer une nouvelle chaîne qui aura en charge les paquets icmp.
; Tous les paquets icmp seront détruits,
ipchains −N ch−icmp
ipchains −I ch−icmp −j DENY
# détruire les paquets
ipchains −I input −p icmp −j ch−icmp
# si c'est un paquet icmp alors, traiter dans ch−icmp

; Remarque : Le traitement des paquets icmp demande une
; attention particulière. Voir ipchains−HOWTO pour cela.

; Créer une nouvelle chaîne qui aura en charge les paquets provenant
; du réseau interne et qui passent sur l'interface eth0,
ipchains −N int−ext /* paquets qui vont de l'intérieur vers l'extérieur */
ipchains −I int−ext −j ACCEPT /* on accepte tout c'est juste pour tester */
ipchains −I input −i eth0 −j int−ext /* si c'est un paquet qui vient du réseau
                                       interne alors, traiter dans int−ext */

; Interdire la sortie de tous les protocoles sauf le protocole HTTP,
; (deux formes d'écriture possible),

; Première solution :
ipchains −A output −p tcp −s 192.168.1.0/24 −d 0.0.0.0/0 80 −j ACCEPT
ipchains −A output −s 192.168.1.0/24 −d 0.0.0.0/0 −j DENY

; Deuxième solution
ipchains −A output −p tcp −s 192.168.1.0/24 −d 0.0.0.0/0  !80 −j DENY
; /* Rejette tout ce qui n'est pas à destination d'un port 80.

; Scénario, on souhaite :
; 1 − pour toutes les machines d'un réseau privé sauf les machines
; " .1 " et " .2 ",  laisser sortir tous les paquets par translation
; d'adresse sur les ports 25, 110 et 53 (smtp, pop3 et DNS) interdire tout le reste.
; 2 − interdire pour toutes les machines, tout passage des protocoles MS.

#Règles IPCHAINS du FW en date du 24/10/2000
# On ferme d'abord tout
ipchains −P forward  DENY
#SMTP on laisse passer
ipchains −A forward −j MASQ −p tcp −s 192.168.90.0/24 −d 0.0.0.0/0 25
g
ipchains −A forward −j MASQ −p tcp −s 192.168.90.0/24 −d 0.0.0.0/0 110

#DNS oui avec les protocoles de transport UDP et TCP
ipchains −A forward −j MASQ −p tcp −s 192.168.90.0/24 −d 0.0.0.0/0 53
ipchains −A forward −j MASQ −p udp −s 192.168.90.0/24 −d 0.0.0.0/0 53

#machines .1 et .2
ipchains −A forward −j MASQ  −s 192.168.90.1/32 −d 0.0.0.0/0
ipchains −A forward −j MASQ  −s 192.168.90.2/32 −d 0.0.0.0/0

#Les protocoles de MS (transport UDP et TCP)
ipchains −A input −j DENY −p udp −s 192.168.90.0/24 −d 0.0.0.0/0  137
ipchains −A input −j DENY −p udp −s 192.168.90.0/24 −d 0.0.0.0/0  138
ipchains −A input −j DENY −p tcp −s 192.168.90.0/24 −d 0.0.0.0/0  137
ipchains −A input −j DENY −p tcp −s 192.168.90.0/24 −d 0.0.0.0/0  138
```

Chapitre 58. Iptables

Langage et commandes Iptables

58.1. Langage d'Iptables

Extrait de la page de manuel et de l'aide :

```
Synopsis :
Usage iptables −commande chaine spécification_de_règle [options]

Commandes :                             Fonction :
iptables −h                             aide en ligne
iptables −L                             lister les chaînes et règles actives
iptables −N  chain                      créer une nouvelle chaîne utilisateur
iptables −X chain                       supprimer chaîne utilisateur
iptables −F [chain]                     vider  une chaîne (ou toutes)
```

```
iptables −P chain cible                 traitement par défaut pour une règle
iptables −A chain règle                 ajouter une règle à la chaîne
iptables −I chain [numéro] règle        insérer une chaîne en position numéro
iptables −D chain [numéro] [règle]      effacer une règle
iptables −R chain [numéro] [règle]      remplacer une règle
iptables −C chain                       tester un paquet dans une règle
iptables −Z [chain]                     remettre à zéro les compteurs
```

Options : Les principaux paramètres qui peuvent apparaître dans la partie règle sont :

```
Paramètres :
−i interface                            appliquer la règle sur cette interface d'entrée
−o interface                            appliquer la règle sur cette interface de sortie
−t table                                table concernée ( filter par défaut)
−s [!] X.X.X.X/lg                       adresse source, longueur du masque
                                        le ! signifie la négation (cf. exemple)
−d [!] X.X.X.X/lg                       adresse dest (et longueur du masque)
−p protocole                            tcp / udp / icmp / all
−−source−port [!] [port[:port]]         port source (ou intervalle de ports)
−−destination−port [!] [port[:port]]    port destination (ou intervalle de ports)
−−tcp−flags [!] masq comp               flags de paquets tcp ( ex. SYN)
−j cible                                ACCEPT/DROP/QUEUE/RETURN/REDIRECT/
                                        MASQUERADE / DNAT/ SNAT/ LOG
```

Exemple d'utilisation des règles :

```
# iptables −F INPUT [−t filter]
# iptables −P INPUT ACCEPT
# iptables −A INPUT −i eth0 −s ! 192.168.0.0.0/24 −j DROP
```

Attention à l'écriture. Les cibles (target) sont sensibles à la casse.

Les tables passées en lignes de commandes ou dynamiquement, ne sont pas conservées lors du redémarage de la machine. Il est nécessaire d'utiliser les commandes iptables−save et iptables−restore pour les sauvegarder, ou les mettre dans un script qui est exécuté au démarrage de la machine, par exemple dans /etc/init.d/rc.local.

58.2. Exemples d'utilisation d'iptables

Prenez l'habitude avant toute chose de vérifier l'état des règles avant de procéder à des tests. Videz les au besoin avant de commencer.

```
iptables −F INPUT && iptables −P INPUT ACCEPT
iptables −F FORWARD && iptables −P FORWARD ACCEPT
iptables −F OUTPUT && iptables −P OUTPUT ACCEPT
```

Interdire en entrée toute requête HTTP

```
iptables −A INPUT  −s 192.168.0.0/24  −p tcp −−dport 80 −j REJECT
```

Iptables permet de journaliser les informations avec l'option −−log−prefix

```
iptables −A INPUT  −s 192.168.0.0/24  −p tcp −−dport 80 −j REJECT \
 −−log−prefix "Session http rejetée"
```

58.3. La traduction d'adresse − NAT

La traduction d'adresse est gérée avec les chaînes PREROUTING, POSTROUTING et OUTPUT. Dans la chaîne PREROUTING (avant routage), on ne peut modifier que l'adresse de destination. L'adresse source est conservée. On fait donc du DNAT. On dit qu'on fait du "NAT destination". Dans la chaîne POSTROUTING, (après routage) on ne peut modifier que l'adresse source. L'adresse de destination est conservée. On fait donc du SNAT. On dit qu'on fait du "NAT source".

Certaines applications arp, ftp, irc ... nécessitent des options. Ces options sont compilées directement dans votre noyau ou sous forme de modules.

Figure 58−1. Compilation du noyau pour netfilter

Les modules sont dans :

```
 /lib/modules/VotreNoyau/kernel/net/ipv4/netfilter/
```

58.3.1. Le DNAT ou NAT Destination

On substitue à l'adresse de destination des paquets provenant du réseau public, une adresse du réseau local privé. Dans l'exemple, les paquets à destination de la machine 195.x sont redirigés vers la machine 172.y. On ne tient pas compte du port.

```
iptables −F INPUT  ; iptables −P INPUT ACCEPT
```

```
iptables -F OUTPUT  ; iptables -P OUTPUT ACCEPT
iptables -F FORWARD ; iptables -P FORWARD ACCEPT
iptables -t nat -F PREROUTING
iptables -t nat -A PREROUTING -d  195.115.19.35/32 \
         -j  DNAT --to-destination  172.16.0.1/32
```

58.3.2. Le SNAT ou NAT Source

Le SNAT consiste à substituer une adresse source dans un paquet sortant à son adresse source d'origine. On substitue ici, aux requêtes provenant du réseau 192.168.0.0/24, une des 10 adresses publiques.

```
iptables -F INPUT  ; iptables -P INPUT ACCEPT
iptables -F OUTPUT  ; iptables -P OUTPUT ACCEPT
iptables -F FORWARD ; iptables -P FORWARD ACCEPT
iptables -t nat -F POSTROUTING
iptables -t nat -A POSTROUTING -s 192.168.0.0/24 \
         -j  SNAT --to-source   195.115.90.1-195.115.90.10
```

58.3.3. L'IP Masquerade

Dans ce cas, les adresses privées, utilisent toutes la même adresse publique. C'est le procédé qui est utilisé avec ipchains. Il s'agit en fait de translation de port avec ipchains ou de SNAT (Nat Source) avec iptables.

```
iptables -F INPUT  ; iptables -P INPUT ACCEPT
iptables -F OUTPUT  ; iptables -P OUTPUT ACCEPT
iptables -F FORWARD ; iptables -P FORWARD ACCEPT
iptables -t nat -F POSTROUTING
iptables -t nat -A POSTROUTING -s 10.10.10.0/8 \
         -j  SNAT --to-source 139.63.83.120
```

Une autre option consiste à utiliser l'option "MASQUERADE"

```
iptables -t nat -F POSTROUTING
iptables -t nat -A POSTROUTING -s 10.10.10.0/8 -j  MASQUERADE
```

58.3.4. Exemple sur un réseau privé

L'exemple ci-dessous indique comment partager un accès internet chez soi pour plusieurs machines. Vous voulez partager un accès "dial" sur votre réseau.

```
#ifconfig
eth0      Lien encap:Ethernet  HWaddr 00:80:C8:7A:0A:D8
          inet adr:192.168.0.1  Bcast:192.168.0.255  Masque:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:19950 errors:0 dropped:0 overruns:0 frame:0
          TX packets:24988 errors:0 dropped:0 overruns:0 carrier:0
          collisions:2 lg file transmission:100
          RX bytes:2830076 (2.6 Mb)  TX bytes:12625623 (12.0 Mb)
          Interruption:5 Adresse de base:0x240

ppp0      Lien encap:Protocole Point-à-Point
          inet adr:212.47.248.114  P-t-P:212.47.251.49  Masque:255.255.255.255
          UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1524  Metric:1
          RX packets:27 errors:0 dropped:0 overruns:0 frame:0
          TX packets:29 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 lg file transmission:3
          RX bytes:4924 (4.8 Kb)  TX bytes:1365 (1.3 Kb)

#echo 1 >  /proc/sys/net/ipv4/ip_forward
#iptables -t nat -A POSTROUTING -s 192.168.0.0/24 -j MASQUERADE

iptables -t nat -L
Chain PREROUTING (policy ACCEPT)
target     prot opt source               destination

Chain POSTROUTING (policy ACCEPT)
target     prot opt source               destination
MASQUERADE  all  --  192.168.90.0/24      anywhere

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

Le tour est joué, vous n'avez plus qu'à configurer les autres clients avec un DNS et une passerelle par défaut. Votre accès wan est partagé.

Chapitre 59. Application sur le routage et le filtrage de paquets IP

L'objectif est de configurer un Pentium sous Linux, muni de trois cartes réseau 10/100 Base T, pour en faire un firewall. Voir schéma réseau et maquette. Le TP est réalisable avec Ipchains ou Iptables.

59.1. Introduction

L'objectif est de configurer une machines sous Linux, munie de trois cartes réseau 10/100 Base T, pour en faire un firewall. Pour visualiser les trames qui sont échangées sur le réseau, vous utiliserez un outil d'analyse de trames comme tcpdump, ethereal ou autre. Enfin pour filtrer le trafic et faire de la translation de port (PAT) la fonctionnalité ipchains du noyau Linux sera utilisée. Un outil comme iptraf est intéressant pour visualiser la translation de port sur le routeur.

Le rôle d'un routeur (ou passerelle/gateway en terminologie IP) est de "router" les paquets entrants par une interface, vers une de ses interfaces de sortie, en fonction de l'adresse IP du destinataire (en fait du réseau auquel il appartient).

Le routeur dispose d'une table de routage interne, visible avec la commande route.

Exemple de table de routage :

```
Destination     Passerelle      Genmask         Indic Metric Ref    Use Iface
192.168.0.0     *               255.255.255.0   U     0      0        0 eth0
172.16.0.0      *               255.255.0.0     U     0      0        0 eth1
10.0.0.0        *               255.0.0.0       U     0      0        0 eth2
127.0.0.0       *               255.0.0.0       U     0      0        0 lo
default         192.168.0.254   0.0.0.0         UG    0      0        0 eth0
```

Le noyau Linux sait router les paquets entre différentes interfaces et vers des réseaux. Il faut que la fonction "ip forwarding" soit activée. Cela est faisable soit dynamiquement :

```
echo 0/1 > /proc/sys/net/ipv4/ip_forward
```

soit en modifiant un fichier de configuration.

```
/etc/network/options sur debian
/etc/sysconfig/network sur mandrake
```

59.2. Fonctions de filtrage

Le filtrage des paquets au niveau IP, transport ou adresse est utilisé pour des raisons de sécurité. Par exemple autoriser ou interdire l'accès à un service, à un réseau ou à une machine(Voir fiche de cours).

Avec les noyaux 2.2, il est possible de faire du filtrage et du masquage d'adresse IP par translation de port (PAT). Ipchains utilise, de base, trois chaînes (input, output, forward) qui contiennent des règles de filtrage/masquage.

Les règles de filtrage sont analysées de manière séquentielle, dès qu'une règle correspond au paquet analysé, elle est appliquée.

Chaque paquet est analysé au travers des chaînes. Si un filtre correspond , la règle associée est appliquée au paquet (ACCEPT, DENY, MASQ ...). Sinon le filtre suivant est testé. A la fin de chaque chaîne un traitement "par défaut" est appliqué en dernier ressort (ACCEPT/DENY)

Vous trouverez à la fin du document, un récapitulatif des commandes.

Références : les HOWTOs (Linux Ipchains et IP Masquerade l'on peut télécharger sur http://www.linuxdoc.org)

59.3. TD

Utiliser la documentation donnée à la fin du document.

Figure 59−1. Schéma maquette TD

Ecrire les règles de filtrage iptables qui répondent aux différents problèmes exprimés ci−dessous.

```
Notation :
A = 195.0.0, B = 172.16,
A.0 = toutes les machines du réseau 1, B.0 = toutes les machines du
réseau B
A.1, B.1, indiquent respectivement 195.0.0.1 et 172.16.0.1
```

1. Interdire tous les paquets de A.0 vers B.0
2. Interdire tous les paquets de A.0 vers B.1
3. Interdire tous les paquets NetBIOS sur A.254
4. Masquer toutes les adresses de A.0 pour tous les protocoles
5. Masquer toutes les adresses de A.0 pour tous les protocoles uniquement pour les services SMTP et POP3
6. N'autoriser que les paquets de A.1 vers B.1, interdire tout le reste
7. Interdire les paquets de A.1 vers B.1, autoriser tout le reste

8. N'autoriser que les paquets telnet de A.1 vers B.1, interdire tout le reste
9. N'autoriser que les paquets TCP de A.0 vers B.0, interdire tout le reste.

59.4. Schéma de la maquette pour le TP

Schéma d'un réseau simple. Le routeur à trois interfaces réseau. Les segments de classe A et le segment de classe B peuvent être réalisés à l'aide de simple machines et de câbles croisés.

Figure 59−2. Réseau simple

Schéma d'un réseau simple plus complexe. Quatre maquettes construites sur le schéma précédent sont reliées à un segment fédérateur. Ce dernier peut être le réseau de l'établissement.

Figure 59−3. Réseau intégré

Vous allez réaliser ce TP sur la maquette que vous avez monté pour le routage. Vous utiliserez trois machines C, R et S. R fait référence à votre routeur. C fait référence au client installé sur le réseau de classe B, S fait référence au serveur installé sur le réseau de classe A.

C est sur un réseau "privé" (C sera client pour les tests)

S est sur un réseau "public"

R servira de routeur et pare−feu (firewall) entre votre réseau privé et le réseau public.

Vous allez réaliser ce TP en trois parties :

1. Première partie : vérification du routage sur le routeur logiciel R,
2. Deuxième partie : mise en oeuvre de règles de filtrage simples et de la translation d'adresse sur R
3. Troisième partie : mise en place de règles de filtrages par adresse, par port et par protocole.

Vous utiliserez la documentation de ipchains ou iptables ainsi que les exemples commentés qui vous sont fournis.

59.4.1. Première partie : installation et configuration du routage

1. Installez les interfaces réseau sur le routeur et démarrer la machine.
2. Créez les fichiers de configuration des interfaces eth0, eth1 et eth2.
3. Installez le module de la carte dans le fichier " /etc/module.conf " si nécessaire
4. Lancez le service réseau sur R " /etc/rc.d/init.d/network restart ", relevez les routes.
5. Vérifiez à l'aide de la commande " ifconfig " que les interfaces sont bien actives. Corrigez tant que ce n'est pas le cas. Configurez et installez C sur votre réseau privé,
6. Testez l'accès de C vers les deux interfaces de R. Pourquoi l'accès vers le réseau public ne fonctionne pas ?
7. Activez le routage (ip forward) entre les interfaces réseau. (NETWORKING=yes dans /etc/sysconfig/network pour Mandrake ou ip_forward=yes dans /etc/network/options pour Debian.

   Vous pouvez également utiliser la commande :

   ```
   echo 1 > /proc/sys/net/ipv4/ip_forward
   ```

   Relancez le service réseau, relevez les routes de R à l'aide de la commande " route ".
8. Vérifiez avec une commande " ping " que les deux interfaces de R sont bien visibles à partir de C.
9. Vérifiez l'installation du programme iptraf sur R

59.4.2. Règles de filtrage simples

1. Interdisez toutes les requêtes de C vers S (extérieur) (tester)
2. Autorisez toutes les requêtes du client C vers S (tester)
3. Interdisez tout passage sur la chaîne FORWARD en laissant l'option " ACCEPT " sur les chaînes INPUT et OUTPUT (tester)
4. Activez le masquage d'adresse pour toutes les machines du réseau auquel appartient C vers tous les autres réseaux (tester)
5. Activez le masquage d'adresse pour toutes les machines du réseau auquel appartient C vers un seul des autres réseaux. Exemple si vous êtes sur le domaine vert.org, activer le masquage vers bleu.org. Les paquets ne devront pas passer vers noir.org ou rouge.org. (tester)
6. Restaurez l'état réalisé au point 4 (après masquage). Activez " iptraf " sur l'interface publique. Vérifiez que seule, l'adresse ip de l'interface du réseau privé apparaît et que toutes les autres sont " masquées ". Notez la translation de port et identifiez les sessions.

59.4.3. Règles de filtrage par adresse, port et protocoles

1. Relevez dans le fichier " /etc/protocol " et " /etc/services " les ports et noms des protocoles utilisés par les services ftp et http. Refusez tout trafic de C vers l'extérieur à destination de ces ports. Vérifier que telnet est accepté. (tester puis restaurer)
2. N'autorisez à C l'accès qu'à une seule machine (par exemple le client C de rouge.org si vous êtes sur vert.org), refusez tout le reste. (tester puis restaurer)
3. Refusez tout accès à la machine (192.168.x.1) de votre réseau (192.168.x.0) vers le réseau (192.168.y.0). (tester.) (tester aussi en modifiant l'adresse ip de votre client que celui−ci passe.)
4. Relevez les ports et protocoles utilisés par les services de résolution de noms Construisez les règles afin qu'elles répondent au problème suivant. On désire que : toutes les machines du réseau privé puissent avoir accès à tous les autres services de n'importe quel réseau, la machine C de votre réseau privé ne peut pas envoyer de requête UDP/DOMAIN vers la machine S.

   (tester pour les requêtes sur le serveur de noms puis pour des requêtes sur le serveur ftp en utilisant l'adresse IP)
5. Pour les deux traitements qui suivent, vous utiliserez deux écritures, dont l'opérateur " ! " (not) qui permet d'obtenir des compléments.

   Autorisez toutes les requêtes qui sortent vers l'extérieur, sauf celles qui vont sur un port ftp

   N'autorisez aucune requête hormis celles qui sont à destination du port 80

Chapitre 60. Outils et ressources complémentaires pour les TP

Outils et ressources complémentaires

60.1. Iptraf

Iptraf est utile pour la visualisation des connexions tcp, de la translation de port et de la translation d'adresse.

Figure 60−1. iptraf

Pour démarrer automatiquement le moniteur de traffic IP

```
iptraf −i all
```

Pour démarrer automatiquement les statistiques générales des interfaces

```
iptraf −g
```

Pour démarrer automatiquement les informations détaillées d'une interface

```
iptraf −d eth0
```

60.2. Documentations complémentaires

http://www.freenix.org/unix/linux/HOWTO/Liste−des−HOWTO.html

http://www.linuxguruz.org/iptables/

Le guide de l'administration de la couche IP avec Linux : http://linux−ip.net/

fwbuilder : http://www.fwbuilder.org/

Chapitre 61. Initiation au routage

Initiation au routage statique

61.1. Initiation au routage

Comment, au travers d'un réseau étendu comme Internet, un ordinateur arrive−t−il à communiquer avec un autre situé à des kilomètres et dont il ne connaît à peu près rien si ce n'est son adresse IP ? Les réseaux de nombreux opérateurs publics et privés assurent une connexion physique entre les deux appareils. Mais ce n'est pas tout. Cela fonctionne parce que le protocole IP est routable. Les objectifs de cette série d'articles sont de vous présenter les principes du routage ainsi que la configuration des routeurs dans un réseau.

D'un point de vue utilisateur, nous considérons Internet comme sur la figure A : un immense et unique réseau. En réalité, Internet est composé d'un ensemble de réseaux reliés via des appareils particuliers : les routeurs (figure B).

Figure 61−1. Internet

Le protocole IP est capable de choisir un chemin (une route) suivant lequel les paquets de données seront relayés de proche en proche jusqu'au destinataire. À chaque relais sur la route correspond un routeur. L'ordinateur émetteur du paquet de données doit trouver le premier relais. Ensuite, chaque routeur est chargé de trouver le suivant. Enfin, le dernier routeur remet le paquet sur le réseau du destinataire. Le routage IP fonctionne de façon totalement décentralisée au niveau des appareils qui constituent le réseau. Aucun n'a une vision globale de la route que prendront les paquets de données.

61.1.1. Les principes du routage

Avant d'aborder la partie pratique, je vais vous présenter quelques explications théoriques qui me semblent un préalable indispensable à une compréhension précise du routage. Je vais essayer de ne pas être trop long. Le routage IP repose sur quatre principes :

61.1.1.1. Une adresse IP est structurée

Chaque interface réseau d'un appareil possède une adresse IP unique dans tout le réseau global. Cette adresse est structurée en deux partie : la première partie (ou préfixe) donne le numéro du réseau. La seconde partie (ou suffixe) donne le numéro de l'interface dans ce réseau. Un masque est associé à cette adresse et permet au logiciel IP de déterminer le préfixe réseau d'une adresse en calculant un ET logique avec le masque. Exemple :Interface : eth0 Adresse IP : 192.168.2.254 Masque réseau : 255.255.255.0

192.168.2.1 ET 255.255.255.0 donne le préfixe réseau de l'adresse soit : 192.168.2.0

Si vous ne vous sentez pas à l'aise avec ces notions, inutile d'aller plus loin : je vous renvoie vers les précédents numéros de Linux magazine qui ont déjà traité ce point.

61.1.1.2. Les paquets de données comportent l'adresse IP de l'émetteur et du destinataire

Lors de l'émission, le protocole découpe les données en petits paquets (souvent appelés datagrammes IP). Ces paquets ont tous la même structure :

Figure 61−2. Datagramme

C'est l'en−tête qui contient, entre autre, les adresses de l'émetteur et du destinataire. Un appareil chargé du routage analysera l'adresse du destinataire afin d'aiguiller le paquet vers le prochain routeur menant à sa destination.

61.1.1.3. Chaque appareil possède une table de routage gérée par le logiciel IP

Une table de routage est une liste contenant essentiellement trois types d'information : des adresses réseau avec le masque réseau associé et le moyen de les atteindre. Soit le réseau est directement connecté à l'appareil, dans ce cas le moyen de l'atteindre est le nom de l'interface, soit, il s'agit de l'adresse du prochain routeur situé sur la route vers ce réseau. Par exemple, considérons sur un appareil quelconque, sa table de routage :

Réseau	Masque	Moyen de l'atteindre
192.168.2.0	255.255.255.0	eth0
100.0.0.0	255.0.0.0	eth1
101.0.0.0	255.0.0.0	eth2
192.168.1.0	255.255.255.0	100.0.0.1
192.168.3.0	255.255.255.0	101.0.0.2

Cette table est riche d'enseignements. On apprend très précisemment que l'appareil possède trois interfaces réseau (eth0, eth1, eth2) ainsi que les adresses IP des réseaux qui sont directement reliés à ces interfaces. On connaît les adresses IP de deux routeurs. On sait qu'il existe deux réseaux 192.168.1.0 et 192.168.3.0 et qu'ils sont respectivement derrière les routeurs 100.0.0.1 et 101.0.0.2. Par contre, il est impossible d'affirmer que ces deux réseaux sont directement reliés à ces routeurs. Pour résumer, on peut dresser le schéma suivant :

Figure 61−3. Topologie 1

Quelques observations complémentaires :

1. − étant donné que l'appareil observé possède trois interfaces, c'est très probablement un routeur. Cependant, notez que tout appareil fonctionnant sous TCP/IP possède une table de routage (qu'il soit routeur ou non);
2. − pour que le routage fonctionne, il est impératif que toutes les interfaces réseau possédant le même préfixe réseau soient reliées au même réseau physique.

61.1.1.4. Tous les appareils sous IP exécutent le même algorithme

Lors de l'émission d'un paquet de données, le logiciel IP recherche une correspondance dans la table en appliquant le masque réseau de chaque ligne avec l'adresse IP de destination du paquet. Notez qu'il parcourt la table dans l'ordre décroissant des masques afin de garantir le *best match* (la correspondance la plus précise entre l'adresse dans la table et l'adresse de destination).

Au total, seules quatre possibilités sont imaginables :

- ce préfixe correspond à celui d'un réseau directement connecté : il y a remise directe du paquet sur le réseau et le routage est terminé.
- ce préfixe correspond à celui d'un réseau accessible via un routeur : on récupère l'adresse physique de ce routeur et on lui transmet le paquet. Notez que l'adresse IP de l'émetteur reste inchangée.
- ce préfixe n'a pas de correspondance dans la table mais il existe un routeur par défaut dans la table : on transmet au routeur par défaut.
- si aucun des trois cas précédents n'est rempli, on déclare une erreur de routage.

Si tous les appareils exécutent le même algorithme de routage, alors qu'est−ce qui différencie un simple ordinateur d'un routeur ? Un élément fondamental : un routeur est en mesure de relayer des paquets reçus et dont il n'est pas l'émetteur.

61.1.2. Place à la pratique

Voilà, vous savez tout sur les aspects théoriques. Une mise en pratique est maintenant indispensable. Je vous propose de travailler avec le routeur logiciel GNU Zebra. C'est, à mon sens, un excellent logiciel, pour les raisons suivantes :

- c'est un logiciel libre sous licence GNU ;
- il propose une interface de configuration interactive accessible via telnet ;
- il fonctionne selon une philosophie et un langage de configuration proche de routeurs répandus dans les entreprises (ce qui permet d'avoir accès à une bonne bibliographie);
- il supporte les principaux protocoles de routage (nous développerons ce point dans la deuxième partie de cet article);
- il fonctionne avec Ipv6.

Les manipulations présentées ci−après ont été réalisées avec Zebra 0.91a sous une Redhat 7.2. Ce logiciel est prévu pour fonctionner sous Linux (noyau 2.0.37 et suivants) et BSD. Une version pour Hurd est prévue.

Je vous propose de travailler avec la maquette suivante :

Figure 61−4. Topologie pratique

Ce réseau est composé de 3 routeurs (R1, R2 et R3) et de trois stations (S1, S2, S3). Je suppose que toutes les interfaces réseau sont actives et correctement configurées. Les masques réseau à utiliser sont les masques par défaut de la classe d'adresse (255.0.0.0 pour les adresses commençant par 100 et 101 et 255.255.255.0 pour les adresses commençant par 192).

61.1.2.1. Mise en place des routeurs

Zebra doit être installé sur chaque ordinateur qui fera office de routeur. Vous pouvez vous procurer des paquetages d'installation ou bien compiler le logiciel à partir des fichiers sources. Dans ce cas, l'installation se fait par les habituels ./configure ; make ; make install . Cette phase produit plusieurs exécutables (un par protocole de routage) mais nous n'utiliserons pour l'instant que zebra. En principe, les exécutables ont été copiés dans /usr/local/bin et les fichiers de configuration dans /usr/local/etc (ils portent le même nom que l'exécutable avec l'extension .conf). Suivant la méthode d'installation, ils pourront être situés ailleurs, ce n'est pas un problème. Pour une aide plus détaillée, reportez−vous sur mon site, vous y trouverez une traduction française du manuel.

Avant de charger le démon de routage, sur chaque routeur créez un fichier /usr/local/etc/zebra.conf. Insérez les deux lignes suivantes :hostname Rx(Zebra) ! remplacez le x par le numéro du routeurpassword foo

À la place de foo, indiquez le mot de passe que vous souhaitez saisir lorsque vous vous connecterez au routeur via telnet. Maintenant, vous pouvez lancer le démon de routage avec la commande zebra −d afin qu'il s'exécute en tâche de fond.

61.1.2.2. Configuration des stations

Plaçons−nous dans le shell de S1 et observons la configuration des interfaces :

```
S1 # ifconfigeth0 Lien encap:Ethernet HWaddr 00:50:56:40:40:98
inet adr:192.168.1.1 Bcast:192.168.1.255 Masque:255.255.255.0
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RXpackets:89 errors:0 dropped:0 overruns:0 frame:0*
```

```
TX packets:58 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 lg file transmission:100
RX bytes:6771 (6.6 Kb) TX bytes:3357 (3.2 Kb)
Interruption:10 Adresse de base:0x1080

lo Lien encap:Boucle locale
inet adr:127.0.0.1 Masque:255.0.0.0
UP LOOPBACK RUNNING MTU:16436 Metric:1
RX packets:77 errors:0 dropped:0 overruns:0 frame:0
TX packets:77 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 lg file transmission:0
RX bytes:4758 (4.6 Kb) TX bytes:4758 (4.6 Kb)
```

Cet appareil dispose d'une interface Ethernet active nommée eth0 ainsi que de l'interface de bouclage logiciel lo. Toute machine fonctionnant avec IP possède cette interface.

Je vous ai dit tout à l'heure que tout appareil fonctionnant sous IP disposait d'une table de routage. Listons le contenu de cette table sur S1 :

```
 S1 # route
```

Destination	Passerelle	Genmask	Indic	Metric	Ref	Use	Iface
192.168.1.0	*	255.255.255.0	U	0	0	0	eth0
127.0.0.0	*	255.0.0.0	U	0	0	0	lo

À partir des adresses IP des interfaces de l'ordinateur, le logiciel IP en a déduit cette table de routage élémentaire. Pour lui, toutes les machines qui disposent d'une adresse commençant par 192.168.1.0 sont forcément sur le réseau physique connecté à l'interface eth0.

Conclusion, si je tente un ping vers une adresse de ce réseau (et si une machine possède cette adresse), j'obtiens une réponse. Essayons entre S1 et R1 qui font partie du même réseau :

```
S1 # ping 192.168.1.254
PING 192.168.1.254 (192.168.1.254) from 192.168.1.1 : 56(84) bytes of data.
64 bytes from 192.168.1.254: icmp_seq=0 ttl=255 time=23.411 msec
Warning: time of day goes back, taking countermeasures.
64 bytes from 192.168.1.254: icmp_seq=1 ttl=255 time=2.308 msec
--- 192.168.1.254 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max/mdev = 2.308/12.859/23.411/10.552 ms
```

Je cherche à contacter un appareil dont l'adresse commence par 192.168.1.0. Cette adresse figure dans la table routage, donc tout va bien.

Essayons maintenant de contacter l'interface eth1 de R1 :

```
S1 # ping 100.0.0.1 connect: Network is unreachable
```

La sanction est immédiate. En clair, votre système d'exploitation favori vous répond : "désolé, mais je n'ai absolument aucune idée de la façon dont je pourrais bien atteindre le réseau 100.0.0.0".

Pour résoudre ce problème, il faut que je renseigne ma table de routage et que j'indique comment atteindre le réseau 100.0.0.0. Sur le schéma, c'est très clair. Pour aller sur ce réseau, il faut passer par R1. Comme S1 ne peut joindre pour l'instant que les appareils dont l'adresse commence par 192.168.1.0, j'indiquerai comme moyen d'atteindre le réseau, l'adresse 192.168.1.254 qui est l'adresse IP de l'interface du routeur qui se situe sur le réseau de S1.Pour réaliser cette configuration, tapons la commande :

```
S1 # route add -net 100.0.0.0 netmask 255.0.0.0 gw 192.168.1.254
```

Félicitations ! Vous venez de saisir votre première commande de configuration de routage. Elle signifie que le réseau 100.0.0.0/8 (masque réseau sur 8 bits) est situé derrière le routeur (gw = gateway) d'adresse 192.168.1.254.Vous pouvez le tester, cette configuration fonctionne pour le réseau 100.0.0.0 mais si l'on généralise, il faudrait saisir pour tous les réseaux que l'on cherche à contacter, une commande identique ! Observez bien le schéma. R1 est le seul routeur directement accessible par S1. Quel que soit le réseau que S1 cherche à contacter, il ne peut être que derrière R1. Par conséquent, il existe une commande qui permet d'indiquer une route par défaut :

```
S1 # route add default gw 192.168.1.254
```

Listons le contenu de la table de routage :

```
s1 # route Table de routage IP du noyau
```

Destination	Passerelle	Genmask	Indic	Metric	Ref	Use	Iface
192.168.1.0	*	255.255.255.0	U	0	0	0	eth0

127.0.0.0	*	255.0.0.0	U	0	0	0	lo
100.0.0.0	192.168.1.254	255.0.0.0	U	0	0	0	eth0
default	192.168.1.254	0.0.0.0	UG	0	0	0	eth0

La ligne commençant par 100.0.0.0 est devenue inutile. Supprimons la :

```
S1 # route del −net 100.0.0.0 netmask 255.0.0.0 gw 192.168.1.254
```

La plupart du temps, il n'existe qu'un seul routeur pour sortir d'un réseau d'extrémité. On configure alors sur chaque station l'adresse IP de ce routeur par défaut. Le logiciel IP crée une entrée dans sa table de routage identique à celle que nous venons d'observer. Vous devrez donc, sur chaque station de notre réseau définir l'adresse de son routeur par défaut, soit en tapant une commande route, soit en modifiant les fichiers de configuration des cartes réseau et en redémarrant.

Revenons sur S1 et testons notre configuration. Contactons à nouveau l'interface 100.0.0.1 du routeur R1 :

```
S1 # ping 100.0.0.1 PING 100.0.0.1 (100.0.0.1) from 192.168.1.1 :
56(84) bytes of data. Warning: time of day goes back, taking countermeasures.
64 bytes from 100.0.0.1: icmp_seq=0 ttl=255 time=3.746 msec
64 bytes from 100.0.0.1: icmp_seq=1 ttl=255 time=1.812 msec

−−− 100.0.0.1 ping statistics −−−
2 packets transmitted, 2 packets received, 0% packet
loss round−trip min/avg/max/mdev = 1.812/2.779/3.746/0.967 ms
```

Parfait ça marche ! Je devrais donc pouvoir également contacter R2 puisqu'il est lui aussi dans le réseau 100.0.0.0 :

```
S1 # ping 100.0.0.2 PING 100.0.0.2 (100.0.0.2) from 192.168.1.1 :
56(84) bytes of data. (il ne se passe rien, donc CTRL−C)

−−−  100.0.0.2 ping statistics −−−
10 packets transmitted,
0 packets received, 100% packet loss
```

Eh bien, ce n'est pas brillant. Certes, S1 n'indique plus de message d'erreur mais les paquets transmis ne sont jamais retournés. Vous vous doutez qu'il existe une solution. Profitons−en, c'est l'occasion de vous donner quelques éléments pour repérer un problème de routage.

61.1.2.3. Configuration des routeurs

Puisque le routage est une chaîne, il faut suivre les paquets dans chaque maillon afin de trouver l'origine du problème. Nous savons que l'interface eth1 du routeur R1 reçoit les paquets puisqu'elle nous les retourne. Donc, le problème vient de R2. Positionnons−nous dans le shell de R2, la commande tcpdump va nous aider à observer ce qu'il se passe sur son interface eth1 :

```
R2 # tcpdump −nt −i eth1 tcpdump: listening on eth1 192.168.1.1 >
100.0.0.2: icmp: echo request (DF) 192.168.1.1 >
100.0.0.2: icmp: echo request (DF)
```

Oui, elle reçoit les paquets... mais elle n'en retourne aucun. Vous l'avez compris : comme pour S1 tout à l'heure, R2 n'a aucune idée de l'endroit où se trouve le réseau de l'émetteur des paquets (192.168.1.0) puisque celui−ci n'est pas directement connecté. Il faut donc configurer sa table de routage. Pour ce faire, nous allons cette fois travailler avec Zebra. Zebra possède une interface telnet sur le port 2601. Dans le shell de R2, tapez :

```
R2 # telnet localhost 2601 Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'. Hello, this is zebra (version 0.91a).
Copyright 1996−2001 Kunihiro Ishiguro. User Access Verification Password:
```

Tapez le mot de passe que vous avez saisi dans le fichier zebra.conf , vous arrivez dans le mode de visualisation de la configuration du routeur. Ensuite, passez en mode de configuration (mode privilégié appelé mode enable dans le logiciel) :

```
R2(Zebra)> enable
```

Pour vous repérer dans Zebra, observez bien le prompt, il vous indique dans quel mode vous vous trouvez (par exemple, le # indique que vous êtes en mode privilégié). Ensuite, dans l'interpréteur de commande, vous pouvez saisir à tout moment un ? pour obtenir la liste contextuelle des commandes. Enfin, lorsque vous appuyez sur la touche tabulation, Zebra complète la commande en cours de saisie.

Observons la table de routage gérée par Zebra :

```
R2(Zebra)# show ip route
Codes: K − kernel route, C − connected, S − static,
       R − RIP, O − OSPF, B − BGP, > − selected route,
       * − FIB route
C>* 100.0.0.0/8 is directly connected, eth1
C>* 101.0.0.0/8 is directly connected, eth2
C>* 127.0.0.0/8 is directly connected, lo
C>* 192.168.2.0/24 is directly connected, eth0
```

Nous ne voyons aucune trace du réseau 192.168.1.0. C'est pour cela que R2 ne peut retourner les paquets ICMP à S1. Bien sûr, Zebra permet d'ajouter une route. Passons en mode « terminal de configuration » :

```
R2(Zebra)# configure terminal
```

Puis ajoutons la route :

```
R2(Zebra)(config)# ip route 192.168.1.0/24 100.0.0.1
```

Revenons au mode enable et listons à nouveau la table :

```
R2(Zebra)(config)# end
R2(Zebra)# show ip route
Codes: K − kernel route, C − connected, S − static,
       R − RIP, O − OSPF, B − BGP, > − selected route,
       * − FIB route
C>* 100.0.0.0/8 is directly connected, eth1
C>* 101.0.0.0/8 is directly connected, eth2
C>* 127.0.0.0/8 is directly connected, lo
S>* 192.168.1.0/24 [1/0] via 100.0.0.1, eth1
C>* 192.168.2.0/24 is directly connected, eth0
```

Une route statique (notée S) est apparue (vous apprendrez dans le prochain article comment configurer une route dynamique ainsi que la signification des nombres entre crochets [1/0]).

Un ping 100.0.0.2 depuis S1 passe désormais sans problème. Si l'on reprend le schéma du réseau, vous vous doutez que des manipulations similaires sont à réaliser pour le réseau de S3. Il faut donc ajouter une route vers 192.168.3.0/24. Vous connaissez maintenant les commandes :

```
R2(Zebra)# configure terminal
R2(Zebra)(config)# ip route 192.168.3.0/24 101.0.0.2
R2(Zebra)(config)# end
```

Visionnons la configuration en mémoire de Zebra :

```
R2(Zebra)# show running−config

Current configuration:
!
hostname R2(Zebra)
password foo
!
interface lo
!
interface eth0
!
interface eth1
!
interface eth2
!
ip route 192.168.1.0/24 100.0.0.1
ip route 192.168.3.0/24 101.0.0.2
!
line vty
!
end
```

Afin qu'à chaque démarrage de Zebra les routes statiques soient prises en compte, il faut enregistrer cette configuration « mémoire » vers le fichier zebra.conf :

```
R2(Zebra)# copy running−config startup−config
Configuration saved to /etc/zebra/zebra.conf
```

Bien, nous avons presque fini. Quelques routes sont à créer sur R1 et R3 pour que le routage sur notre réseau soit complet.

61.1.2.3.1. Une erreur à éviter

Sur R1, il faut créer une route vers 192.168.3.0. Une erreur fréquente consiste à créer la route suivante :

```
R1(Zebra)(config)# ip route 192.168.3.0/24 101.0.0.2
```

Ce qui signifie : le réseau 192.168.3.0/24 est situé derrière le routeur R3. Bien sûr, cette phrase est juste, mais souvenez−vous de ce que nous disions en introduction : le routage fonctionne de proche en proche. Ainsi, comme nous sommes sur R1, il suffit d'indiquer que le routeur nous permettant d'atteindre le réseau de S3 est R2 et non R3.

61.1.2.3.2. Configuration de R1

Ceci étant dit, voici la configuration de R1 :

```
R1(Zebra)# show running−config
Current configuration:
```

```
!
hostname
R1(Zebra) password foo
!
interface lo
!
interface eth0
!
interface eth1
!
ip route 101.0.0.0/8 100.0.0.2
ip route 192.168.2.0/24 100.0.0.2
ip route 192.168.3.0/24 100.0.0.2
!
line vty
!
end
```

61.1.2.3.3. Configuration de R3

Avant de lire ci−dessous, essayez de déterminer la configuration de R3. Elle est très proche de celle de R1.

```
R3(Zebra)# show running−config

Current configuration:
!
hostname
R3(Zebra) password foo
!
interface lo
!
interface eth0
!
interface eth1
!
ip route 100.0.0.0/8 101.0.0.1
ip route 192.168.1.0/24 101.0.0.1
ip route 192.168.2.0/24 101.0.0.1
!
line vty
!
end
```

Ouf ! Voilà, c'est fini. Vous devez pouvoir réaliser des ping de n'importe quelle machine vers n'importe quelle autre.

61.1.3. Conclusion

La configuration des routeurs peut vous sembler fastidieuse, voire impossible si le réseau comporte beaucoup de routeurs et que sa topologie évolue fréquemment. Il faudrait sans cesse reconfigurer les routeurs. Heureusement, le monde est bien fait : il existe des protocoles qui permettent aux routeurs de s'échanger les informations de routage dont ils disposent afin que les tables s'adaptent aux évolutions du réseau comme le protocole RIP par exemple.

Chapitre 62. Le routage dynamique avec RIP

Initiation au routage RIP

62.1. Introduction

Le premier article sur le routage statique a présenté les concepts nécessaires à la bonne compréhension du routage IP. Nous avons vu que les routeurs sont de véritables postes d'aiguillage qui acheminent de proche en proche les paquets IP dans l'inter−réseau. On peut configurer manuellement des routes statiques sur chaque routeur. Mais dans un réseau important, cette tâche devient rapidement cauchemardesque ! Heureusement, des protocoles de routage ont été développés afin que les routeurs s'échangent les informations dont ils disposent. On parle dans ce cas de routage dynamique. L'objet de cet article est de vous présenter le fonctionnement et la mise en oeuvre d'un protocole de routage des plus élémentaires : RIP (Routing Information Protocol).

Avant d'aborder la partie pratique avec Zebra, nous évoquerons les avantages du routage dynamique en comparaison du routage statique. Nous détaillerons ensuite le fonctionnement du protocole RIP.

62.1.1. Pourquoi le routage dynamique ?

Comme nous l'avons défini dans le précédent article, le routage statique consiste à indiquer l'adresse IP des réseaux que l'on cherche à atteindre. On associe à chaque adresse, le nom de l'interface du routeur ou l'adresse IP du routeur voisin se situant sur la route vers ces réseaux de destination. Si le réseau global est complexe, la configuration peut être fastidieuse et source d'erreurs. De plus, lorsque un nouveau réseau est ajouté, il faut reconfigurer l'ensemble. Enfin, pour prévenir tout dysfonctionnement (panne d'un routeur, ligne coupée, etc.), il faut effectuer une surveillance permanente et reconfigurer chaque routeur le cas échéant. Si la route est rétablie, il faut recommencer la manipulation.

L'idée générale du routage dynamique est la suivante : plutôt que de centraliser la configuration du routage dans les mains d'un individu dont le temps de réaction est fatalement long et les risques d'erreurs importants, nous allons délocaliser cette tâche au niveau des routeurs. En effet, chaque appareil n'est−il pas le mieux placé pour connaître les adresses des réseaux auxquels il est directement relié puisque chacune de ses interfaces possède une adresse IP ? De plus, étant directement au contact des supports de communication, il peut établir un diagnostic sur l'état des liaisons. Fort de ces informations, il n'a plus qu'à les partager avec ses voisins. De proche en proche, les nouvelles se répandront à chaque routeur du réseau. L'intervention humaine se situera en amont dans la définition de directives et de règles à appliquer par les routeurs pour la diffusion des routes.

62.1.2. Le protocole RIP

Comme toujours, pour qu'une communication puisse s'établir, chaque interlocuteur doit parler la même langue. Il a donc été nécessaire de concevoir un protocole. RIP a été défini, pour sa version 1 dans la RFC 1058 et pour sa version 2 dans la RFC 2453. Par la suite, je ne traiterai que RIPv2. Toutefois, avant de passer à la partie pratique, nous évoquerons rapidement les différences entre ces deux versions.

62.1.2.1. Quelles informations de routage s'échanger ?

Le principe général est très simple. Un routeur RIP transmet à ses voisins les adresses réseau qu'il connaît (soit les adresses de ses interfaces, soit les adresses découvertes via les autres routeurs) ainsi que la distance pour les atteindre. Ces couples adresse/distance sont appelés vecteurs de distance.

62.1.2.2. La notion de distance

Nous touchons ici au concept de *métrique*, fondamental dans le domaine du routage. En effet, il arrive fréquemment (c'est même une situation recherchée pour des raisons de tolérance aux pannes) que le réseau ait une topologie maillée. Dans ce cas, plusieurs routes mènent à la même destination. Le routeur doit alors choisir la route qu'il considère la meilleure vers une destination donnée.

La seule métrique utilisée par RIP est la distance correspondant au nombre de routeurs à traverser (*hop* ou nombre de sauts) avant d'atteindre un réseau. Pour chaque route, RIP calcule la distance. Ensuite, si des routes redondantes apparaissent, RIP retient celle qui traverse le moins de routeur (donc avec la distance la plus faible).

Du fait de la méthode utilisée pour diffuser les routes, la longueur d'une route (et par voie de conséquence le diamètre du réseau) est limitée. La norme limite la distance maximale d'une route à quinze. Cela signifie que deux réseaux ne peuvent être éloignés de plus de quinze routeurs. Nous verrons ci−après qu'une distance égale à seize (distance "infinie" pour RIP) joue un rôle particulier en indiquant qu'une route est devenue inaccessible.

62.1.2.3. Un exemple

Prenons l'exemple simple du réseau sur lequel nous avons travaillé dans l'article précédent :

Figure 62−1. Topologie du réseau

Afin de bien comprendre le routage dynamique, supposons la situation initiale suivante : sur chaque routeur, toutes les interfaces réseau sont actives, aucune route statique n'est définie et le routage RIP est inactif.

Sur R1, lorsque l'on active le processus de routage RIP, une première table est constituée à partir des adresses IP des interfaces du routeur. Pour ces réseaux directement connectés au routeur, la distance est égale à un puisqu'il faut au moins traverser ce routeur pour les atteindre. On obtient :

Adresse/Préfixe	Moyen de l'atteindre	Distance
100.0.0.0/8	eth1	1
192.168.1.0/24	eth0	1

Tableau 1 : table initiale constituée par R1

R1 transmet à ses voisins immédiats (ici, il n'y a que R2) un seul vecteur de distance {192.168.1.0/24, 1} qui signifie : "je suis le routeur d'adresse IP 100.0.0.1 et je connaîs un moyen d'atteindre le réseau 192.168.1.0/24 en un saut". Aucune information sur le réseau commun aux deux routeurs (100.0.0.0/8) n'est transmise car R1 considère que R2 connaît déjà ce réseau.

Ensuite, lorsque l'on active RIP sur R2, il constitue la table ci−après à partir de ses propres informations et de celles reçues de R1 :

Adresse/Préfixe	Moyen de l'atteindre	Distance
100.0.0.0/8	eth1	1
101.0.0.0/8	eth2	1
192.168.1.0/24	100.0.0.1	2

192.168.2.0/24	eth0	1

Tableau 2 : table constituée par R2

Sur R2, RIP a calculé que la distance pour atteindre 192.168.1.0/24 est égale à deux puisqu'il faut traverser R2 puis R1. R2 a déduit le "moyen de l'atteindre" à partir de l'adresse IP de l'émetteur contenue dans le paquet RIP.

Lorsque RIP sera démarré sur R3, la route vers 192.168.3.0/24 avec une distance de deux sera ajoutée dans la table ci−dessus.

Dans ce petit exemple, aucune restriction n'a été définie sur la diffusion des routes. Donc, à l'issue d'un certain délai appelé temps de convergence, variable selon la taille du réseau, chaque routeur connaît un moyen d'atteindre chaque réseau.

62.1.2.4. Algorithme général de RIP

Examinons un peu plus en détail le fonctionnement de RIP. Lors de l'initialisation du routeur, celui−ci détermine l'adresse réseau de ses interfaces puis envoie sur chacune une demande d'informations (table RIP complète) aux routeurs voisins. Lors de la réception d'une demande, un routeur envoie sa table complète ou partielle suivant la nature de cette demande. Lors de la réception d'une réponse, il met à jour sa table si besoin. Deux cas peuvent se présenter :

- pour une nouvelle route, il incrémente la distance, vérifie que celle−ci est strictement inférieure à 15 et diffuse immédiatement le vecteur de distance correspondant ;
- pour une route existante mais avec une distance plus faible, la table est mise à jour. La nouvelle distance et, éventuellement, l'adresse du routeur si elle diffère sont intégrées à la table.

Bien sûr, si l'appareil reçoit une route dont la distance est supérieure à celle déjà connue, RIP l'ignore. Ensuite, à intervalles réguliers (les cycles durent 30 secondes environ), la table RIP est diffusée qu'il y ait ou non des modifications.

Des routes doivent être retirées de la table gérée par RIP dans deux situations.

En premier lieu, un réseau immédiatement connecté devient inaccessible (panne de l'interface, de la ligne, modification de la topologie par l'administrateur, etc.). Les routeurs RIP reliés à ce réseau affectent dans leur table une distance "infinie" (16 comme indiqué plus haut) à cette route. Elle est conservée pendant la durée d'un temporisateur de "maintien" (ou garbage collect) de 120 secondes puis est supprimée. Immédiatement puis pendant toute la durée de ce délai, le vecteur est diffusé. Un routeur qui reçoit un vecteur avec une distance de 16 comprend : "il faut que tu retires cette route de ta table car elle est devenue invalide !". De proche en proche, cette information se propage.

En second lieu, un routeur du réseau tombe en panne. Cela veut peut−être dire que les réseaux situés derrière cet appareil sont devenus inaccessibles. Mais comment savoir si un routeur est en panne ? RIP considère qu'un routeur qui n'a pas donné de nouvelles depuis trois minutes est hors−service. Pour gérer cette situation, il attribue à toutes les routes dynamiques un temporisateur initialisé à 180 secondes (par défaut). A chaque réception d'un vecteur de distance déjà présent dans la table, le compteur est réinitialisé. Mais si jamais ce compteur atteint zéro, la route est considérée comme invalide. On se retrouve alors dans la situation précédente (distance infinie, temporisateur de maintien, diffusion de l'information puis suppression de la route). Maintenant, si un autre routeur connaît une route menant vers un des réseaux que l'on vient de retirer, c'est parfait ! Notre routeur intègrera cette nouvelle route dans sa table : RIP permet la tolérance aux pannes.

Comment justifier l'existence de ces mécanismes qui peuvent paraître un peu complexes ? Cela est dû à une faiblesse des algorithmes à vecteurs de distance que l'on appelle "problème de la convergence lente". Dans certains cas, après la panne d'un accès réseau, deux routeurs voisins risquent de se transmettre mutuellement puis, ensuite, de propager des informations contradictoires au sujet de ce réseau et créer ainsi une boucle de routage infinie. Zebra met en oeuvre les mécanismes nommés "split horizon" (une information de routage reçue d'une interface n'est jamais retransmise sur celle−ci), "poison reverse" (temporisateur de maintien) et "triggered update" (une panne est immédiatement diffusée sans attendre le prochain cycle de diffusion des tables) afin d'empêcher ce phénomène et de réduire le délai de convergence.

62.1.2.5. Améliorations de RIPv2 par rapport à RIPv1

Même si les principes évoqués ci−dessus sont valables quelle que soit la version de RIP, les différences restent intéressantes à relever. Les améliorations de RIPv2 sont :

- diffusion des masques de sous−réseaux associés aux adresses réseaux (RIPv1 n'utilisait que les masques réseau par défaut) ;
- utilisation d'adresses multicast pour diffuser les vecteurs de distance au lieu d'adresses de broadcast, ce qui réduit l'encombrement sur le réseau ;
- support de l'authentification en transportant un mot de passe crypté avec MD5 ;
- interopérabilité entre protocoles de routage en diffusant des routes apprises à partir d'autres protocoles.

L'ensemble de ces raisons rendent RIPv1 obsolète bien qu'il soit supporté par la plupart des routeurs logiciels ou matériels.

62.1.3. Place à la pratique

Afin de mieux apprécier les facilités offertes par le routage dynamique, je vous propose de travailler sur une topologie légèrement modifiée afin d'introduire un lien redondant. Voici le plan :

Figure 62−2. Topologie de travail

Que vous ayez suivi ou non la première partie de cet article, vous devez partir sur chaque appareil avec un fichier de configuration du routeur Zebra (/usr/local/etc/zebra.conf) vierge à l'exception de ces deux lignes :

```
hostname Rx(Zebra) ! remplacez le x par le numéro du routeur
password foo
```

Vous devez également créer un fichier de configuration pour le routeur RIP (/usr/local/etc/ripd.conf) ayant une apparence très proche de celui de Zebra :

```
hostname Rx(RIP) ! remplacez le x par le numéro du routeur
password foo
```

Lorsque ces manipulations sont faites, lancez les deux démons de routage sur les trois routeurs en respectant l'ordre des commandes, par exemple sur R1 :

```
R1 # zebra −d
R1 # ripd −d
```

Zebra nécessite que les deux démons soient présents car son architecture est la suivante :

Figure 62−3. Architecture de Zebra

Le démon zebra est un intermédiaire entre le noyau de Linux et les démons de routage dynamique. Il peut récupérer les routes statiques définies directement sous Linux afin de les diffuser via le routage dynamique.

zebra lui−même permet de définir des routes statiques. Enfin, il peut récupérer des routes dynamiques pour les intégrer à la table de routage gérée par le noyau Linux. Routages statique et dynamique peuvent cohabiter sans problème avec Zebra mais les concepteurs du logiciel conseillent fortement de ne définir les routes statiques que dans zebra (évitez de les définir dans le shell Linux ou dans les démons de routage dynamique).

62.1.3.1. Activation de RIP sur le premier routeur

Afin d'observer la diffusion des routes qu'opère RIP, je vous propose de saisir la commande suivante dans le shell d'un routeur immédiatement voisin de R1, R2 par exemple :

```
R2 # tcpdump −i eth1 −nt −s 0 src host
      100.0.0.1tcpdump: listening on eth1
```

Ensuite, connectons−nous au routeur RIP sur R1 avec un telnet sur le port 2602. Dans le shell de R1 :

```
R1 # telnet localhost 2602
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.

Hello, this is zebra (version 0.91a).
Copyright 1996−2001 Kunihiro Ishiguro.

User Access Verification
Password:
R1(RIP) >
```

Cette opération étant réalisée, comme pour zebra il faut activer le mode privilégié, passer dans le terminal de configuration et enfin, entrer dans la configuration du routeur RIP :

```
R1(RIP)> enable
R1(RIP)#conf t
R1(RIP)(config)# routerrip
R1(RIP)(config−router)#
```

La première tâche consiste à déterminer les types de routes en notre "possession" que nous souhaitons voir diffuser à nos voisins. Cette configuration se fait par la commande redistribute. Voici les différents paramètres de cette commande :

```
R1(RIP)(config−router)# redistribute ?
```

```
bgp Border Gateway Protocol (BGP)
connected Connected
kernel Kernel routes
ospf Open Shortest Path First (OSPF)
static Static routes
```

On constate que l'on peut diffuser des routes propres à la machine comme les routes statiques et les adresses des réseaux directement connectés. Mais nous pouvons également utiliser RIP pour diffuser des routes dynamiques apprises via RIP ou d'autres protocoles de routage comme OSPF ou BGP. Dans tous les cas, les routes diffusées aux voisins seront vues par eux comme des routes étiquetées "découvertesgrâce à RIP".

Nous choisissons de diffuser les adresses des réseaux directement connectés :

```
R1(RIP)(config−router)# redistribute connected
```

Pour l'instant, rien ne se produit. Il faut indiquer à RIP sur quels réseaux nous souhaitons voir la diffusion des routes s'opérer. Nous retrouvons ici une commande commune avec le routage statique. Avant de la valider, pensez à observer le résultat du tcpdump sur l'écran de R2 :

```
R1(RIP)(config−router)#  network 100.0.0.0/8
```

À ce stade, R1 diffuse sur le réseau 100.0.0.0/8 la table RIP à intervalles de 30 secondes. Le résultat sur R2 doit ressembler à ceci :

```
R2 # tcpdump −i eth1 −nt −s  0 src host 100.0.0.1
tcpdump: listening on eth1
100.0.0.1.router > 224.0.0.9.router: RIPv2−req 24 (DF) [ttl 1]
100.0.0.1 > 224.0.0.9: igmp v2 report 224.0.0.9 (DF) [ttl 1]
100.0.0.1.router > 224.0.0.9.router: RIPv2−resp [items 2]:
{102.0.0.0/255.0.0.0}(1)
{192.168.1.0/255.255.255.0}(1) (DF) [ttl 1]
```

Les messages adressés par R1 se font via une adresse multicast convenue pour les routeurs RIP : 224.0.0.9. Les dernières lignes montrent clairement que RIP diffuse deux vecteurs de distance : un concernant le réseau 102.0.0.0/8 et un autre concernant le réseau 192.168.1.0/24. Observons sur R1 la table avec laquelle RIP travaille :

```
R1(RIP)(config−router)# end
R1(RIP)# show ip rip
Codes: R − RIP, C − connected, O − OSPF, B − BGP
```

	Network	Next Hop	Metric	From	Time
C	100.0.0.0/8		1		
C	102.0.0.0/8		1		
C	192.168.1.0/24		1		

```
 R1(RIP)#
```

RIP a été activé sur le réseau 100.0.0.0/8, donc aucune information le concernant n'est diffusée sur ce même réseau pour des raisons évidentes d'optimisation mais aussi, pour la gestion du problème de la convergence lente.

62.1.3.2. Activation de RIP sur le deuxième routeur

Bien, nous avons fait la moitié du travail. Un routeur diffuse grâce à RIP les informations de routage qu'il possède. Mais pour l'instant, c'est inefficace car personne n'est là pour les écouter et les exploiter. Il faut donc faire les mêmes manipulations sur R2 puis à terme sur R3. Passons dans le shell de R2 :

```
R2 # telnet localhost 2602
...
R2(RIP)> enable
R2(RIP)# conf t
R2(RIP)(config)# router rip
R2(RIP)(config−router)# redistribute connected
R2(RIP)(config−router)# network 100.0.0.0/8
R2(RIP)(config−router)# end
R2(RIP)# show ip rip
Codes: R − RIP, C − connected, O − OSPF, B − BGP
```

	Network	Next Hop	Metric	From	Time
C	100.0.0.0/8		1		
C	101.0.0.0/8		1		
R	102.0.0.0/8	100.0.0.1	2	100.0.0.1	02:52
R	192.168.1.0/24	100.0.0.1	2	100.0.0.1	02:52
C	192.168.2.0/24		1		

```
R2(RIP)#
```

La table ci−dessus a été constituée par le processus RIP tournant sur R2. Le routeur d'adresse 100.0.0.1 (R1) l'a informé de la présence de deux routes vers deux réseaux, ce qui est conforme aux informations affichées par tcpdump tout à l'heure. La distance (Metric) est égale à deux puisque ces réseaux sont directement connectés à R1. Un compteur est activé pour chaque route dynamique notée R (pour RIP). C'est un compte à rebours qui périodiquement repart de 03:00 à chaque diffusion reçue de R1.

Vous pouvez faire un show ip rip sur R1 afin de constater qu'il a opéré un travail similaire.

62.1.3.3. Filtrer la diffusion des routes

Lorsque l'on saisit un "redistribute connected" dans RIP, le routeur diffuse toutes les routes de type "directement connectées", sans distinction. Difficile de garder une certaine "intimité" dans ces conditions ! Zebra, qui est bien conçu, propose des mécanismes pour filtrer la diffusion des routes grâce aux "listes de distribution".

Supposons qu'un nouveau réseau soit connecté à R2. Pour les besoins de l'exemple, vous pouvez créer une interface fictive simulant ce réseau. Dans le shell Linux, créons cette interface :

```
R2 # ifconfig dummy0 111.0.0.1/8 up
```

Zebra détecte cette nouvelle interface et transmet l'information à RIP. Comme à ce stade, RIP doit diffuser toutes les routes connectées. Il informe immédiatement ses voisins. Vérifions ceci sur R1 :

```
R1(RIP)# show ip rip
Codes: R − RIP, C − connected,O − OSPF, B − BGP
```

	Network	Next Hop	Metric	From	Time
C	100.0.0.0/8		1		
R	101.0.0.0/8	100.0.0.2	2	100.0.0.2	02:43
C	102.0.0.0/8		1		
R	111.0.0.0/8	100.0.0.2	2	100.0.0.2	02:43
C	192.168.1.0/24		1		
R	192.168.2.0/24	100.0.0.2	2	100.0.0.2	02:43

```
R1(RIP)#
```

On constate que R1 a appris l'existence de 111.0.0.0/8. Nous allons interdire à R2 de diffuser l'existence de ce réseau à ses petits camarades. Pour ce faire, il faut créer une règle indiquant que l'adresse 111.0.0.0/8 est bloquée grâce à une liste d'accès. Ensuite, il faut affecter cette règle à une liste de distribution qui indiquera sur quelle interface l'appliquer. Retournons sur R2, dans le terminal de configuration de RIP :

```
R2(RIP)> enableR2(RIP)# conf t
```

Définition de la règle :

```
R2(RIP)(config)# access−list 1 deny 111.0.0.0/8
R2(RIP)(config)# access−list 1 permit any
```

Le "1" après "access−list" identifie la liste d'accès. Ce numéro sera utilisé pour l'associer à la liste de distribution. N'oubliez pas la deuxième ligne. Il faut dire explicitement à RIP que toutes les autres adresses ne sont pas bloquées.

Maintenant, affectons la liste d'accès à une liste de distribution. Il faut indiquer sur quelles interfaces ces règles sont à appliquer :

```
R2(RIP)(config)# router rip
R2(RIP)(config−router)# distribute−list 1 out eth1
R2(RIP)(config−router)# distribute−list 1 out eth2
```

À partir de cet instant, plus aucune information n'est diffusée par R2 concernant 111.0.0.0/8. Sur R1, avec un

```
show ip rip
```

, vous constaterez que le temporisateur de la route tombe à 0. Elle se voit ensuite attribuer une métrique infinie pendant le délai du temporisateur "garbage collect" puis elle disparaît.

Dans notre exemple, le résultat de cette manipulation est que les réseaux directement connectés au routeur R2, en particulier 192.168.2.0/24 qui contient des ordinateurs, peuvent communiquer avec 111.0.0.0/8. En revanche, l'extérieur n'a pas connaissance du réseau 111.0.0.0/8 qui ne peut pas communiquer avec les réseaux situés derrière les autres routeurs.

Cet article n'a pas la prétention de présenter toutes possibilités offertes par les listes d'accès et les listes de distribution qui sont, en fait, très nombreuses. La documentation du logiciel indique l'ensemble des paramètres de ces différentes commandes.

62.1.3.4. Paramétrage de RIP

Toute la configuration de RIP peut être affichée sous une forme synthétique. Par exemple, sur le routeur R1, en mode privilégié (#) :

```
R1(RIP)# show ip protocols Routing Protocol is "rip"
      Sending updates every 30 seconds with +/−50%, next due in 35 Timeout
      after 180 seconds, garbage collect after 120 seconds Outgoing update
      filter list for all interface is not set Incoming update filter list for
      all interface is not set Default redistribution metric is 1
      Redistributing: connected Default version control: send version 2,
      receive version 2
```

Interface	Send	Recv	Key−chain
eth1	2	2	

```
Routing for Networks: 100.0.0.0/8 Routing Information
      Sources:
```

Gateway	BadPackets	BadRoutes	Distance	Last Update
100.0.0.2	0	0	120	00:00:34

```
Distance: (default is 120)
```

Examinons brièvement les principaux champs. Les différents temporisateurs sont fixés aux valeurs par défaut. Aucun filtrage des routes en entrée comme en sortie n'est défini. La métrique par défaut de ce routeur est égale à un (c'est cette valeur qui sera ajoutée aux distances des routes apprises dynamiquement). Zebra supporte les deux versions de RIP que l'on peut faire cohabiter mais par défaut, Zebra n'autorise en réception comme en émission que la version 2. Le routage n'est activé pour l'instant que sur l'interface Ethernet 1. Aucun mot de passe n'est défini (nous aborderons cette notion un peu plus loin). La dernière ligne concerne la distance administrative. Comme cette notion est importante, nous la développons ci−dessous.

62.1.3.5. La distance administrative

La dernière ligne du listing précédent évoque une "distance" dont la valeur par défaut est 120. Il s'agit de la distance administrative. Elle n'a aucun rapport avec la distance (métrique) en nombre de sauts calculée par RIP.

Zebra peut constituer une table de routage à partir de routes apprises de différentes manières (réseau directement connecté, route statique, RIP, OSPF, BGP). Si Zebra se trouve avec plusieurs routes menant vers un même réseau mais rapportée par des moyens différents, il doit en choisir une. Il a été décidé d'attribuer à chaque moyen d'apprendre une route un score. La route découverte par un moyen dont le score est le plus faible sera élue. Les distances administratives standards sont les suivantes :

Moyen de découvrir une route	Distance administrative
Connected	0
Static	1
BGP	20
OSPF	110
RIP	120

Tableau 3 : distances administratives par défaut

Ainsi, une route configurée de façon statique (donc par un administrateur) est jugée plus crédible qu'une même route rapportée par RIP (notez au passage que RIP est considéré comme le moins crédible...). On retrouve cette notion de distance administrative dans la table de routage de Zebra. Sur R1, connectez−vous avec telnet au terminal de configuration de Zebra (dans le shell, faites un

```
telnet localhost 2601
```

, puis saisissez le mot de passe) :

```
R1(Zebra)> show ip route
Codes: K −  kernel route, C − connected,
       S − static, R − RIP, O − OSPF, B − BGP,
       > − selected route, * −  FIB route
C>* 100.0.0.0/8 is directly connected, eth1
R>* 101.0.0.0/8 [120/2] via 100.0.0.2, eth1, 00:08:11
C>* 102.0.0.0/8 is directly connected, eth2
C>* 127.0.0.0/8 is directly connected, lo
C>* 192.168.1.0/24 is directly connected, eth0
R>* 192.168.2.0/24 [120/2] via 100.0.0.2, eth1, 00:08:11
R1(Zebra)>
```

Les deux routes dynamiques notées R comportent deux nombres entre crochets ([120/2]). Le premier correspond à la distance administrative et le deuxième à la distance en nombre de sauts.

Remarque importante

J'en profite pour bien préciser que la table ci−dessus est la table de routage, donc utilisée par l'appareil pour router les paquets IP reçus sur ses interfaces réseau. La table que vous consultez dans RIP en faisant un

```
show ip rip
```

n'est pas la table de routage, c'est la table qui sera diffusée aux routeur voisins. La signification de ces deux tables est donc radicalement différente.

62.1.3.6. Avant de continuer

Je vous invite maintenant à activer RIP sur vos trois routeurs en redistribuant les adresses des réseaux immédiatement connectés sur tous les réseaux. Vous connaissez les manipulations à effectuer. A la fin du processus, chaque routeur doit connaître les adresses des six réseaux ainsi que le moyen de les atteindre. Pour information, je vous donne le contenu du fichier de configuration de R3 (ripd.conf) :

```
hostname R3(RIP)
password foo
!
interface lo
!
interface eth0
!
interface eth1
!
interface eth2
!
router rip
redistribute connected
network 101.0.0.0/8
network 102.0.0.0/8
!
line vty
!
end
```

Vous pouvez également, si vous le souhaitez, modifier la valeur par défaut des temporisateurs utilisés par RIP afin de visionner plus rapidement le résultat des manipulations que nous allons réaliser par la suite. Ceci se fait de la façon suivante, par exemple dans R1 :

```
R1(RIP)# conf t
R1(RIP)(config)# router rip
R1(RIP)(config−router)# timers basic 10 30 20
```

Notez bien qu'en exploitation, je vous conseille vivement de conserver ces compteurs à leur valeur par défaut. Avec les durées que nous avons indiqué ici, une partie importante de votre bande passante va être consommée par les diffusions de RIP.

62.1.3.7. La tolérance aux pannes

Supposons que la liaison entre R1 et R2 va tomber en panne. Visionnons la table RIP de R1 avant ce triste événement :

```
R1(RIP)> show ip rip
Codes: R − RIP, C − connected, O − OSPF, B − BGP
```

	Network	Next Hop	Metric	From	Time

C	100.0.0.0/8		1		
R	101.0.0.0/8	100.0.0.2	2	100.0.0.2	02:52
C	102.0.0.0/8		1		
C	192.168.1.0/24		1		
R	192.168.2.0/24	100.0.0.2	2	100.0.0.2	02:52
R	192.168.3.0/24	102.0.0.2	2	102.0.0.2	02:36

Le réseau 100.0.0.0/8 tombe en panne. R1 ne reçoit donc plus d'informations de routage à partir de R2. Si vous observez la table RIP sur R1, vous verrez que toutes les routes issues de R2 finissent par disparaître. Mais pendant ce temps, R3 continue à envoyer des mises à jour via le réseau 102.0.0.0/8. R3 connaît un moyen d'atteindre les réseaux que l'on pouvait joindre auparavant par R2. Aussi, au bout d'un certain délai de convergence, R1 construit la table suivante :

```
R1(RIP)> show ip rip
Code: R − RIP, C − connected, O − OSPF, B − BGP
```

	Network	Next Hop	Metric	From	Time
R	100.0.0.0/8	102.0.0.2	3	102.0.0.2	02:34
R	101.0.0.0/8	102.0.0.2	2	102.0.0.2	02:34
C	102.0.0.0/8		1		
C	192.168.1.0/24		1		
R	192.168.2.0/24	102.0.0.2	3	102.0.0.2	02:34
R	192.168.3.0/24	102.0.0.2	2	102.0.0.2	02:34

Tous les réseaux sont à nouveau accessibles à partir de R1 ! Cela démontre que RIP a su digérer une panne de liaison. Rétablissons le lien entre R1 et R2. Progressivement, on retourne vers la première table car les métriques via R2 sont plus faibles.

62.1.3.8. Un problème de sécurité

Le routage dynamique est pratique car avec très peu de commandes de configuration on arrive à une solution qui fonctionne correctement et qui est même capable de prendre en compte automatiquement des modifications de la topologie. Seulement voilà : imaginez qu'un petit malin insère sur le réseau un routeur RIP et qu'il lui fasse diffuser des routes totalement farfelues. Cela peut créer un certain nombre de désagréments comme des dénis de service par exemple. Pour limiter ce risque, RIPv2 permet d'associer un mot de passe crypté à chaque diffusion de vecteurs de distance. Seuls les routeurs ayant connaissance de ce mot de passe traiteront les informations de routage. Mettons en place ce mécanisme entre R1 et R2 :

```
R1(RIP)# conf t
R1(RIP)(config)# key chain test
R1(RIP)(config−keychain)# key 1
R1(RIP)(config−keychain−key)# key−string motdepasse
R1(RIP)(config−keychain−key)# exit
R1(RIP)(config−keychain)# exit
R1(RIP)(config)# int eth1
```

```
R1(RIP)(config−if)# ip rip authentication mode  md5
R1(RIP)(config−if)# ip rip authentication key−chain test
R1(RIP)(config−if)#
```

Nous créons le porte−clé (keychain) nommé "test" avec le mot de passe "motdepasse". Ce mot de passe est associé à l'interface eth1, il sera transmis au format MD5 (sinon, il est transmis en clair !). Pour que cela fonctionne, vous devrez faire des manipulations identiques sur R2.

Examinons sur le réseau avec une capture de paquets, le contenu des informations de routage reçues de R2 :

```
R1 # tcpdump −i eth1 −nt −s0 src host 100.0.0.2
tcpdump: listening on eth1
100.0.0.2.router > 224.0.0.9.router: RIPv2−resp [items 6]:
[auth 3: 0068 0114 3cfb 0c6f 0000 0000 0000  0000]
{101.0.0.0/255.0.0.0}(1)
{102.0.0.0/255.0.0.0}(2)
{192.168.2.0/255.255.255.0}(1)
{192.168.3.0/255.255.255.0}(2)
[auth 1: 4d71 f8e0 077c cc58 8247 6656 17c3 95f2]
(DF) [ttl 1]
```

Notez au passage que seul le mot de passe est crypté, les informations de routage continuent à circuler en clair.

62.1.4. Conclusion

RIP constitue un excellent moyen pédagogique pour aborder la problématique du routage dynamique. Mais il est peu utilisé en exploitation car il souffre de certaines limitations et défauts qui le cantonnent à des réseaux de taille moyenne. Nous avons vu que le diamètre maximum d'un réseau géré avec RIP est limité à 15 routeurs soit 16 segments de réseau. RIP est un gros consommateur de bande passante du fait de la méthode utilisée pour diffuser les informations de routage (toutes les 30 secondes, l'intégralité de la table RIP est diffusée même si elle n'a subi aucune modification). C'est fâcheux, en particulier sur des liaisons lentes ou facturées au volume de données transférées. La métrique utilisée ne garantit pas que le routage soit optimal. En effet, la distance masque les caractéristiques réelles de la voie de transmission (débit ou coût en particulier). Enfin, le temps de convergence, délai avant que tous les routeurs ne possèdent des tables cohérentes peut être long dans certaines situations. Pour toutes ces raisons, on a cherché à développer un protocole de routage beaucoup plus efficace : OSPF, objet du prochain article.

Chapitre 63. Le routage dynamique avec OSPF

Initiation au routage OSPF

63.1. Introduction

Ce texte suit le la séquence sut le routage statique et le routage dynamique avec RIP.

63.1.1. Rappels sur les éléments vus

Le routeur est un élément essentiel dans l'aiguillage des paquets de données dans un inter−réseau. Pour chaque paquet reçu, il extrait le préfixe réseau de l'adresse IP de destination du paquet et le recherche dans une table qu'il possède en mémoire. Cette table de routage contient essentiellement une liste d'adresses réseau et, pour chacune, le moyen de l'atteindre, à savoir l'adresse d'un routeur immédiatement voisin et situé sur la route vers la destination. Si le routeur trouve dans cette table le préfixe réseau, il transmet le paquet sur le réseau du routeur voisin concerné. Ce processus sera renouvelé par le routeur voisin et ainsi de suite, de proche en proche le paquet sera orienté vers sa destination.

Seulement voilà, il faut saisir les tables de routage ! Travail fastidieux pour les petits doigts agiles de l'administrateur lorsque les réseaux sont de grande taille. De plus, compte tenu de l'évolution du nombre de réseaux à interconnecter dans le cas d'internet, il est de toute façon devenu impossible de se cantonner au routage statique (voir séquence sur le routage statique). C'est pourquoi, le routage dynamique a été imaginé afin d'alléger la charge d'administration mais aussi pour réaliser des réseaux tolérants aux pannes d'un routeur ou d'une liaison. RIP est un bon exemple de protocole de routage dynamique*. Les routeurs supportant RIP s'échangent des informations sur les routes qu'ils possèdent (les fameux "vecteurs de distance"). Si une panne se produit, les routeurs immédiatement voisins notent que certaines routes sont devenues inaccessibles et propagent l'information aux autres. Mais hélas, RIP souffre de certaines limitations qui ont poussé l'IETF (Internet Engineering Task Force) à plancher sur un protocole plus robuste, plus efficace, plus paramétrable et supportant des réseaux de grande taille. Cette merveille s'appelle OSPF (Open Shortest Path First), protocole supporté par Zebra.

63.1.2. Les grands principes

OSPFest un protocole de routage dynamique défini par l'IETF à la fin des années 80. Il a fait l'objet d'un historique relativement complexe de RFC. Ce protocole a deux caractéristiques essentielles : − il est ouvert (le Open de OSPF), son fonctionnement peut être connu de tous ; − il utilise l'algorithme SPF (Shortest Path First), plus connu sous le nom d'algorithme de Dijkstra, afin d'élire la meilleure route vers une destination donnée.

Examinons une topologie qui nous servira de support pour les explications :

Figure 63−1. Exemple de topologie

63.1.2.1. La notion de coût

Supposons que du routeur R1 on cherche à atteindre le réseau 192.168.1.0. Dans une telle situation, RIP aurait élu la route passant par R5 puisque c'est la plus courte en termes de saut. Cependant, imaginez que les liens représentés sous forme d'éclairs soient "rapides" (type Ethernet à 100 Mbps par exemple) et que les liens "droits" soient "lents" (type Ethernet à 10 Mbps par exemple). Le choix de RIP n'est plus du tout pertinent !

OSPF fonctionne différemment. Il attribue un coût à chaque liaison (dénommée lien dans le jargon OSPF) afin de privilégier l'élection de certaines routes. Plus le coût est faible, plus le lien est intéressant. Par défaut, les coûts suivants sont utilisés en fonction de la bande passante du lien :

Type de réseau	Coût par défaut
Ethernet > = 100 Mbps	1
FDDI	1
Ethernet 10 Mbps	10
E1 (2,048 Mbps)	48
T1 (1,544 Mbps)	65
64 Kbps	1562
56 Kbps	1785
19.2 Kbps	5208

La formule de calcul est simplissime : coût = référence / bande passante du lien. Par défaut, la référence est 100 000 000 correspondant à un réseau à 100 Mbps.

OSPF privilégie les routes qui ont un coût faible, donc celles qui sont supposées rapides en terme de débit théorique.

63.1.2.2. La base de données topologique

Avec OSPF, tous les routeurs d'un même réseau (on parle de "zone" dans le vocabulaire OSPF, ceci vous sera expliqué avant la mise en pratique) travaillent sur une base de données topologique identique qui décrit le réseau. Cette base a été constituée pendant une première phase de découverte qui vous sera expliquée un peu plus loin. Examinons la base de données suivante qui décrit la topologie de la figure 1 :

Arc	Coût
R1, R2	1
R1, R5	10
R2, R3	1
R3, R4	10
R3, R5	1
R4, R5	10
R4, 192.168.1.0	10

63.1.2.3. L'élection des meilleures routes

L'algorithme du SPF de Dijsktra va traiter cette base de données afin de déterminer les routes les moins coûteuses. Une fois le traitement réalisé, chaque routeur se voit comme la racine d'un arbre contenant les meilleures routes. Par exemple :

Figure 63−2. Le réseau vu de R1

Figure 63−3. Le réseau vu de R5

Dans l'exemple, entre R1 et 192.168.1.0, la meilleure route passe par R2, R3 et R4 pour un coût total de 1 + 1 + 10 + 10 soit 22.

63.1.2.4. La détermination d'une table de routage

La base de données topologique décrit le réseau mais ne sert pas directement au routage. La table de routage est déterminée par l'application de l'algorithme du SPF sur la base topologique. Sur R1, voici un extrait de la table de routage calculée par SPF au sujet du réseau 192.168.1.0 :

Réseau de destination	Moyen de l'atteindre	Coût
192.168.1.0	R2	22

Sur R5, on aura l'extrait suivant :

Réseau de destination	Moyen de l'atteindre	Coût
192.168.1.0	R4	20

63.1.3. Le fonctionnement d'OSPF un peu plus en détail

Pour administrer un réseau OSPF correctement, il est indispensable de comprendre le fonctionnement interne du protocole.

à l'intérieur d'une même zone, les routeurs fonctionnant sous OSPF doivent préalablement remplir les tâches suivantes avant de pouvoir effectuer leur travail de routage :

1. établir la liste des routeurs voisins ;
2. élire le routeur désigné (et le routeur désigné de secours) ;
3. découvrir les routes ;
4. élire les routes à utiliser ;
5. maintenir la base de donnée topologique.

63.1.3.1. 0. état initial

Le processus de routage OSPF est inactif sur tous les routeurs de la figure 1.

63.1.3.2. Établir la liste des routeurs voisins : Hello, my name is R1 and I'm an OSPF router.

Les routeurs OSPF sont bien élevés. Dès qu'ils sont activés, ils n'ont qu'une hâte : se présenter et faire connaissance avec leurs voisins. En effet, lorsque le processus de routage est lancé sur R1 (commande `router ospf`), des paquets de données (appelés paquets HELLO) sont envoyés sur chaque interface où le routage dynamique a été activé (commande `network`). L'adresse multicast 224.0.0.5 est utilisée, tout routeur OSPF se considère comme destinataire. Ces paquets ont pour but de s'annoncer auprès de ses voisins. Deux routeurs sont dits voisins s'ils ont au moins un lien en commun. Par exemple, sur la figure 1, R1 et R2 sont voisins mais pas R1 et R3.Lorsque le processus de routage OSPF est lancé sur R2, celui−ci récupère les paquets HELLO émis par R1 toutes les 10 secondes (valeur par défaut du temporisateur appelé hello interval). R2 intègre l'adresse IP de R1 dans une base de données appelée "base d'adjacences" (adjacencies database). Cette base contient les adresses des routeurs voisins. Vous pourrez visionner son contenu grâce à la commande `show ip ospf neighbor`. R2 répond à R1 par un paquet IP unicast. R1 intègre l'adresse IP de R2 dans sa propre base d'adjacences. Ensuite, généralisez ce processus à l'ensemble des routeurs de la zone.

Cette phase de découverte des voisins est fondamentale puisque OSPF est un protocole à état de liens. Il lui faut connaître ses voisins pour déterminer s'ils sont toujours joignables et donc déterminer l'état du lien qui les relie.

63.1.3.3. Élire le routeur désigné : c'est moi le chef !

Dans une zone OSPF, l'un des routeurs doit être élu "routeur désigné" (DR pour Designated Router) et un autre "routeur désigné de secours" (BDR pour Backup Designated Router). Le DR est un routeur particulier qui sert de référent au sujet de la base de données topologique représentant le réseau.

Pourquoi élire un routeur désigné ? Cela répond à trois objectifs :

- réduire le trafic lié à l'échange d'informations sur l'état des liens (car il n'y a pas d'échange entre tous les routeurs mais entre chaque routeur et le DR) ;
- améliorer l'intégrité de la base de données topologique (car il y a une base de données unique) ;
- accélérer la convergence (souvenez−vous, c'était le talon d'Achille de RIP).

Comment élire le DR ? Autrement dit, qui va se taper la corvée d'expliquer à ses petits camarades la topologie du réseau ? On ne demande pas qui sait parler anglais ou couper les cheveux comme au temps de la conscription Mais comme il faut bien un critère, le routeur élu est celui qui a la plus grande priorité. La priorité est un nombre sur 8 bits fixé par défaut à 1 sur tous les routeurs. Pour départager les routeurs ayant la même priorité, c'est celui avec la plus grande adresse IP qui est élu. Le BDR sera le routeur avec la deuxième plus grande priorité. Afin de s'assurer que votre routeur préféré sera élu DR, il suffit de lui affecter une priorité supérieure à 1 avec la commande `ospf priority`. Vous devrez faire ceci avant d'activer le processus de routage sur les routeurs car, une fois élu, le DR n'est jamais remis en cause même si un routeur avec une priorité plus grande apparaît dans la zone.

63.1.3.4. Découvrir les routes

Il faut maintenant constituer la base de données topologique. Les routeurs communiquent automatiquement les routes pour les réseaux qui participent au routage dynamique (ceux déclarés avec la commande `network`). Zebra étant multiprotocole, il peut également diffuser des routes provenant d'autres sources que OSPF, grâce à la commande `redistribute`.

Chaque routeur (non DR ou BDR) établit une relation maître/esclave avec le DR. Le DR initie l'échange en transmettant au routeur un résumé de sa base de données topologique via des paquets de données appelés LSA (Link State Advertisement). Ces paquets comprennent essentiellement l'adresse du routeur, le coût du lien et un numéro de séquence. Ce numéro est un moyen pour déterminer l'ancienneté des informations reçues. Si les LSA reçus sont plus récents que ceux dans sa base topologique, le routeur demande une information plus complète par un paquet LSR (Link State Request). Le DR répond par des paquets LSU (Link State Update) contenant l'intégralité de l'information demandée. Ensuite, le routeur (non DR ou BDR) transmet les routes meilleures ou inconnues du DR.

L'administrateur peut consulter la base de données topologique grâce à la commande `show ip ospf database`.

63.1.3.5. Élire les routes à utiliser

Lorsque le routeur est en possession de la base de données topologique, il est en mesure de créer la table de routage. L'algorithme du SPF est appliqué sur la base topologique. Il en ressort une table de routage contenant les routes les moins coûteuses.

Il faut noter que sur une base de données topologique importante, le calcul consomme pas mal de ressources CPU car l'algorithme est relativement complexe.

63.1.3.6. Maintenir la base topologique

Lorsqu'un routeur détecte un changement de l'état d'un lien (cette détection se fait grâce aux paquets HELLO adressés périodiquement par le routeur à ses voisins), celui−ci émet un paquet LSU sur l'adresse multicast 224.0.0.6 : le DR et le BDR de la zone se considèrent comme destinataires. Le DR (et le BDR) intègre cette information à sa base topologique et diffuse l'information sur l'adresse 224.0.0.5 (tous les routeurs OSPF sans distinction). C'est le protocole d'inondation.Toute modification de la topologie déclenche une nouvelle exécution de l'algorithme du SPF et une nouvelle table de routage est constituée.

Voilà pour les principes fondamentaux d'OSPF mais des notions importantes restent à évoquer si vous souhaitez déployer OSPF sur de grands réseaux (en particulier sur le fonctionnement d'OSPF sur un réseau point à point et sur l'agrégation de routes). Si vous voulez approfondir, reportez−vous au livre de C. Huitema cité en bibliographie qui, bien qu'un peu ancien est très complet sur la question. Bien sûr, vous pouvez toujours vous plonger dans les différentes RFC qui constituent OSPF (la RFC 2328 en particulier) et dont la lecture est toujours aussi agréable et passionnante ! (je plaisante, bien sûr).

Avant d'attaquer la pratique, un dernier concept : les zones OSPF.

63.1.3.7. Le concept de zone (area)

Contrairement à RIP, OSPF a été pensé pour supporter de très grands réseaux. Mais, qui dit grand réseau, dit nombreuses routes. Donc, afin d'éviter que la bande passante ne soit engloutie dans la diffusion des routes, OSPF introduit le concept de zone (area). Le réseau est divisé en plusieurs zones de routage qui contiennent des routeurs et des hôtes. Chaque zone, identifiée par un numéro, possède sa propre topologie et ne connaît pas la topologie des autres zones. Chaque routeur d'une zone donnée ne connaît que les routeurs de sa propre zone ainsi que la façon d'atteindre une zone particulière, la zone numéro 0. Toutes les zones doivent être connectées physiquement à la zone 0 (appelée backbone ou réseau fédérateur). Elle est constituée de plusieurs routeurs interconnectés. Le backbone est chargé de diffuser les informations de routage qu'il reçoit d'une zone aux autres zones. Tout routage basé sur OSPF doit posséder une zone 0.

Figure 63−4. Un réseau découpé en trois zones

Le réseau est découpé en trois zones dont le backbone. Les routeurs de la zone 1, par exemple, ne connaissent pas les routeurs de la zone 2 et encore moins la topologie de la zone 2. L'intérêt de définir des zones est de limiter le trafic de routage, de réduire la fréquence des calculs du plus court chemin par l'algorithme SPF ainsi que d'avoir une table de routage plus petite (ce qui accélère la convergence). Les routeurs R1 et R4 sont particuliers puisqu'ils sont "à cheval" entre plusieurs zones (on les appelle ABR pour Area Border Router ou routeur de bordure de zone). Ces routeurs maintiennent une base de données topologique pour chaque zone à laquelle il sont connectés. Les ABR sont des points de sortie pour les zones ce qui signifie que les informations de routage destinées aux autres zones doivent passer par l'ABR local à la zone. L'ABR se charge alors de retransmettre les informations de routage au backbone. Les ABRs du backbone ensuite redistribueront ces informations aux autres zones auxquelles ils sont connectés.

63.1.4. Place à la pratique

Nous allons travailler avec le réseau suivant :

Figure 63−5. Topologie de travail

Le réseau a été découpé en trois zones. Vous remarquez que la zone 0 permet de fédérer l'ensemble du réseau. Il s'agit du backbone dont nous avons déjà discuté. Le découpage de ce réseau en trois zones est un cas d'école dont le but est d'examiner la configuration d'OSPF dans un contexte multi−zone. Généralement, on considère qu'une zone peut accueillir plusieurs dizaines de routeurs.

Pour ne pas surcharger ces lignes inutilement, nous nous en tiendrons ici à la configuration de R1, R2 et R3. Vous verrez que la configuration n'est pas très complexe. Par symétrie, il est facile de l'adapter aux autres routeurs. Pour votre service, chers lecteurs, j'ai mis en ligne une carte cliquable (http://perso.club−internet.fr/pmassol/lm/ospf.html) qui vous permettra de consulter l'état, la configuration complète et la table de routage des six routeurs.

Enfin, avant de commencer, vous trouverez sur http://perso.club−internet.fr/pmassol/zebra.html une traduction (partielle) de la documentation de Zebra.

63.1.4.1. Situation de départ

Vous devez créer des fichiers de configuration pour zebra (`/etc/zebra/zebra.conf`) et ospfd (`/etc/zebra/ospfd.conf`) rudimentaires sur chaque routeur. Par exemple, pour R1 :

− fichier zebra.conf :

```
hostname R1(ZEBRA) password foo
```

− fichier ospfd.conf :

```
hostname R1(OSPF) password foo
```

Vous devez ensuite démarrer (ou redémarrer), dans l'ordre s'il vous plaît, les deux démons zebra et ospfd sur chaque routeur. Enfin, sur R1 entrez dans le terminal de configuration de ospfd via le port telnet 2604 :

```
L
inux# telnet localhost 2604
Hello, this is zebra  (version 0.91a).
Copyright 1996−2001 Kunihiro Ishiguro.

User Access Verification
Password:
R1(OSPF)> enable
R1(OSPF)#
```

Si vous avez envie de suivre précisément les échanges de messages entre routeurs, Zebra propose un puissant mécanisme de débogage grâce à la commande `debug` (je vous laisse découvrir tous ses paramètres). Supposons que je veuille garder une trace de tous les messages HELLO émis par R1 :

```
R1(OSPF)# conf t
R1(OSPF)(config)# log file /var/log/zebra/ospfd.log
R1(OSPF)(config)# debug ospf packet hello send detail
```

Il faut que le répertoire /var/log/zebra existe.

63.1.4.2. Activation du processus de routage

Dans le mode "config", nous allons activer le processus OSPF :

```
R1(OSPF)(config)# routerospf
R1(OSPF)(config−router)#
```

63.1.4.3. 2. Activation des annonces de routes

Le processus de routage OSPF est activé mais rien ne se passe. Comme pour RIP, il faut indiquer sur quel(s) réseau(x) on souhaite que le routage dynamique soit opérationnel. Ceci se fait par la commande `network`. Mais, nouveauté par rapport à RIP qui n'intègre pas le concept de zone, il faut indiquer à quelle zone sera rattaché le réseau. Sur la figure 5, on voit que R1 est relié à deux réseaux. Le réseau 30.0.0.0/8 est attaché à la zone 0 et le réseau 11.0.0.0/8 à la zone 1. La configuration se fait donc de cette manière :

```
R1(OSPF)(config−router)# network 30.0.0.0/8 area 0
R1(OSPF)(config−router)# network 11.0.0.0/8 area 1
```

Que se passe−t−il sur le réseau ? R1 envoie des paquets HELLO sur les interfaces pour lesquelles la commande `network` a été saisie. Mais personne n'est là pour les écouter. Activez le routage sur R2 en adaptant les commandes aux spécificités du routeur. Je vous aide un peu. Sur R2, vous réaliserez les configurations suivantes :

```
R2(OSPF)(config−router)# network 11.0.0.0/8 area 1
R2(OSPF)(config−router)# network 12.0.0.0/8 area 1
```

Enfin, sur R3, vous réaliserez les configurations suivantes :

```
R3(OSPF)(config−router)# network 12.0.0.0/8 area 1
R3(OSPF)(config−router)# network 192.168.3.0/24 area 1
```

Mais sur R3, il y a une particularité. Le réseau 192.168.3.0/24 contient des ordinateurs mais aucun routeur. La commande `network` va diffuser sur ce réseau des annonces de routes ce qui consomme inutilement de la bande passante. Par conséquent, nous allons désactiver cette diffusion :

```
R3(OSPF)(config−router)# passive−interface eth1
```

Ainsi, aucune route n'est diffusée sur cette interface. De même, aucune annonce de route ne sera prise en compte. Le réseau sera considéré comme étant d'extrémité (stub).

63.1.4.4. Affichage de la configuration

Affichons la configuration complète de R1 :

```
R1(OSPF)(config−router)# end
R1(OSPF)# show running−config

Current configuration:
!
hostname R1(OSPF)
password foo
!
!
!
interface lo
!
interface eth0
!
interface eth1
!
router ospf
network 11.0.0.0/8 area 1
network 30.0.0.0/8 area 0
!
line vty
!
end
```

Affichons la configuration complète de R3 :

```
R3(OSPF)# show running−config
Current configuration:
!
hostname R3(OSPF)
password foo
!
!
!
interface lo
!
interface  eth0
!
interface eth1
!
router ospf
passive−interface eth1
network 12.0.0.0/8 area 1
network 192.168.3.0/24 area 1
!
line vty
!
end
```

J'espère que vous avez la même configuration. Si ce n'est pas le cas, vous pouvez annuler une ligne contenant une erreur en vous remettant au même endroit où vous avez saisi la commande et en saisissant à nouveau la commande, mais en la faisant précéder de no.

Pour enregistrer la configuration, je vous rappelle que l'on saisit :

```
R1(OSPF)# copy running−config  startup−config
```

Reproduisez maintenant ces manipulations sur l'ensemble des routeurs du réseau.

63.1.4.5. État des routeurs

Nos petits routeurs ont, en principe, bien travaillé. Dans chaque zone, ils ont élu leur chef (le DR), ils ont échangé leurs connaissances et calculé une magnifique table de routage, ultra−optimale. En résumé, les deux stations d'extrémité de la figure doivent pouvoir s'atteindre avec une commande `ping`. Si jamais ce n'est pas le cas, c'est que probablement vous vous êtes trompé dans une configuration. Dans ce cas, reprenez la configuration de chaque appareil. Utilisez les outils `ping`, `tcpdump` et `traceroutE` pour contrôler votre configuration et suivre les paquets. Et n'oubliez pas que dans un `ping`, il y a un aller mais aussi un retour !

Afin d'illustrer ce dont nous avons discuté dans la toute première partie de cet article, examinons l'état du routeur R1. Nous pouvons faire un diagnostic très complet de l'appareil en utilisant les nombreuses sous−commandes de `show ip ospf`. Vous constaterez que les informations fournies par ospfd sur son état sont beaucoup plus conséquentes que celles que l'on pouvait extirper de ripd.

Dans un premier temps, je vous propose d'examiner l'état de santé général du routeur R3 :

```
R3(OSPF)# show ip ospf
OSPF Routing Process, Router ID: 192.168.3.254
Supports only single TOS (TOS0) routes
This implementation conforms to RFC2328
RFC1583Compatibility flag is disabled
SPF schedule delay 5 secs, Hold time between two SPFs 10 secs
Refresh timer 10 secs
Number of external LSA 0
Number of areas attached to this router: 1
Area ID: 0.0.0.1
Shortcutting mode: Default, S−bit consensus: no
Number of interfaces in this area: Total: 2, Active: 2
Number of fully adjacent neighbors in this area: 1
Area has no authentication
Number of full virtual adjacencies going through this area:
           0
SPF algorithm executed 13 times
Number of LSA 9
```

Le premier bloc décrit le fonctionnement général du routeur : l'ID du routeur (égale à sa plus grande adresse IP), conformité aux RFC, valeurs des temporisateurs. Une seule zone est attachée à ce routeur. C'est la zone 1 (exprimée en notation décimale pointée). Notre routeur a deux interfaces dans la zone, il n'a qu'un seul voisin. L'algorithme du SPF a été exécuté 13 fois. La base de données topologique contient neuf états de liens (LSA). Si notre routeur était attaché à plusieurs zones, le deuxième bloc serait répété autant de fois que de zones. Vous pourrez le constater sur R1.

Maintenant, listons nos informations sur les routeurs voisins :

```
R3(OSPF)# show ip ospf neighbor
```

Neighbor ID	Pri	State	Dead Time	Address	Interface	RXmtL	RqstL	DBsmL
12.0.0.1	1	Full/Backup	00:00:34	12.0.0.1	eth0	0	0	0

Déchiffrons ces informations. La différence entre la colonne ID et la colonne Address, c'est que l'ID identifie l'appareil dans le réseau alors que l'adresse correspond à l'interface à laquelle nous sommes relié avec ce routeur. La colonne State nous apprend deux choses : il est synchronisé avec le routeur désigné grâce à la mention "Full", c' est le "routeur désigné de secours" de la zone grâce à l'indicateur Backup. Ce routeur sera déclaré comme inactif si nous ne recevons pas de message HELLO d'ici 34 secondes (Dead Time).

Voyons le contenu de la base de données topologique de R3 :

```
R3(OSPF)# show ip ospF database
   OSPF Router with ID  (192.168.3.254)
      Router Link States (Area 0.0.0.1)
```

Link ID	ADV Router	Age	Seq#	CkSum	Link count
12.0.0.1	12.0.0.1	981	0x80000006	0xf9e2	2
30.0.0.1	30.0.0.1	952	0x80000003	0xb13e	1
192.168.3.254	192.168.3.254	1063	0x80000005	0x15b7	2

```
        Net Link States (Area 0.0.0.1)
```

Link ID	ADV Router	Age	Seq#	CkSum
11.0.0.2	12.0.0.1	981	0x80000001	0xda39
12.0.0.2	192.168.3.254	1063	0x80000001	0x5d0b

```
        Summary Link States (Area 0.0.0.1)
```

Link ID	ADV Router	Age	Seq#	CkSum	Route
21.0.0.0	30.0.0.1	837	0x80000001	0x0d08	21.0.0.0/8
22.0.0.0	30.0.0.1	702	0x80000001	0x64a5	22.0.0.0/8
30.0.0.0	30.0.0.1	976	0x80000001	0x33e2	30.0.0.0/8
172.18.0.0	30.0.0.1	599	0x80000001	0x4a0d	172.18.0.0/24

Ces trois tableaux présentent de façon synthétique l'ensemble des LSA stockés dans la base topologique. Le premier tableau contient des LSA diffusés par chaque routeur. Ils décrivent l'état des interfaces de chaque routeur. Le deuxième tableau contient des LSA diffusés par le routeur désigné. Ils décrivent la liste des routeurs présents dans chaque réseaux. Le dernier tableau contient un résumé des routes diffusées par le routeur de bordure de zone (ABR). Ce sont des routes qu'il a reçu via le backbone par les routeurs des autres zones. L'âge et le numéro de séquence sont utilisés pour mettre à jour la base lorsque des LSA sont reçus. Le check sum est utilisé pour contrôler l'intégrité des LSA.

Pour obtenir des informations détaillées sur chaque LSA, vous pouvez compléter la commande `show ip ospf database` par `router`, `network` ou `summary`. Par exemple : `show ip ospf database router 192.168.3.254` (qui correspond à la troisième ligne du premier tableau) vous apprendra que ce router est relié à deux réseaux : un de transit (12.0.0.0/8) et un d'extrémité (stub) 192.168.3.0/24.

Enfin, si vous voulez consulter la table de routage obtenue après traitement par SPF des différents LSA, vous n'aurez qu'à saisir un `show ip ospf route`. Rappel important : il y a une différence entre cette table et celle utilisée par le démon zebra pour le routage proprement dit. Souvenez−vous que Zebra est multi−protocole et qu'il a une architecture modulaire (voir LM 43). Chaque démon calcule une table de routage à partir des informations dont il dispose (et qui ne sont pas nécessairement les mêmes pour chaque démon). Ensuite, ils transmettent chacun leur table au démon zebra qui en fait la synthèse. Cette synthèse constitue la véritable table de routage utilisée pour router les paquets.

Nous avons fait un tour d'horizon des principales commandes de Zebra permettant de surveiller l'état de ospfd. Il y en a encore beaucoup d'autres que je vous laisse découvrir (faites un show ip ospf ? par exemple). Il nous reste à observer la table de routage obtenue par zebra. Quittez ospfd et connectez−vous sur le démon zebra (telnet localhost 2601) :

```
R3(Zebra)> show ip route
Codes: K − kernel route, C − connected, S − static,
       R − RIP, O − OSPF,B − BGP,
      > − selected route, * − FIB route

O>* 11.0.0.0/8 [110/20] via 12.0.0.1, eth0, 00:12:48
O  12.0.0.0/8 [110/10] is directly connected, eth0,
            00:14:09
C>* 12.0.0.0/8 is directly connected, eth0
O>* 21.0.0.0/8 [110/40] via 12.0.0.1, eth0, 00:11:18
O>* 22.0.0.0/8 [110/50] via 12.0.0.1, eth0, 00:10:16
O>* 30.0.0.0/8 [110/30] via 12.0.0.1, eth0, 00:12:13
C>* 127.0.0.0/8 is directly connected, lo
O>* 172.18.0.0/24 [110/60] via 12.0.0.1, eth0, 00:08:48
O  192.168.3.0/24 [110/10] is directly connected, eth1,
            00:14:19
C>* 192.168.3.0/24 is directly connected, eth1
```

Les routes notées O ont été découvertes par OSPF. Entre crochets, on observe la distance administrative du protocole (110 par défaut pour OSPF) et le coût de la route pour accéder au réseau. Dans ma topologie, il n'y a que des réseaux à 10 Mbits/s, donc avec un coût par défaut de 10 pour chaque lien.

63.1.4.6. Quelques éléments sur la sécurité

63.1.4.6.1. Filtrer la diffusion des routes

Le premier inconvénient d'un protocole de routage dynamique comme OSPF est sa volubilité. Il a tendance à dévoiler tout un tas d'informations sur les réseaux qu'un administrateur consciencieux n'a pas forcément envie de révéler. Pour limiter la diffusion des routes au strict minimum, ospfd intègre, à l'instar de ripd, un mécanisme d'access−lists. Reportez−vous à l'article publié sur Zebra dans Linux Magazine 43. La configuration est strictement identique. J'en profite pour faire un peu de publicité : si vous êtes intéressés par les problèmes de sécurité, je vous conseille l'excellent magazine M.I.S.C. La série d'articles sur la "protection de l'infrastructure réseau IP" constitue sur certains points un approfondissement intéressant.

63.1.4.6.2. Protéger les annonces de routes

Le deuxième inconvénient d'un protocole de routage dynamique comme OSF est sa naïveté. Il croit tout ce qu'on lui dit ! Un petit malin pourrait s'amuser à diffuser des routes farfelues à vos routeurs, ce qui pourrait provoquer des dénis de service. Pour pallier à cela, on peut activer l'authentification des annonces sur une zone. Voici les manipulations à réaliser sur chaque routeur :

```
Routeur(OSPF)(config−router)# area 1 authentication message−digest
```

Ensuite, pour chaque interface participant à la diffusion des routes :

```
Routeur(OSPF)(config)# int ethx
Routeur(OSPF)(config-if)# ospf message-digest-key 1 md5 motdepasse
```

Vous adapterez motdepasse à vos besoins. Ce mot de passe doit bien sûr être connu de tous les routeurs.

63.1.5. Conclusion

OSPF est un protocole de routage dynamique moderne, robuste et conçu pour les grands réseaux. On constate qu'il est nettement plus complexe que RIP. Pas forcément dans sa configuration mais dans son fonctionnement interne. Un inconvénient de ce protocole est qu'il peut être gourmand en puissance de calcul et en mémoire lorsque le réseau comporte beaucoup de routes ou qu'il y a de fréquentes modifications de topologie.

OSPF est un protocole IGP (Interior Gateway Protocol), c'est-à-dire qu'il agit au sein d'un système autonome. Un AS (Autonomous System) est un ensemble de réseaux gérés par un administrateur commun. Chaque système autonome possède un numéro identifiant sur 16 bits délivré par l'IANA (Internet Assigned Numbers Authority) ou ses délégations. Classiquement, les multinationales, les opérateurs de télécom ou les fournisseurs d'accès à Internet détiennent un système autonome. Pour assurer le routage entre les systèmes autonomes, un protocole de type EGP (Exterior Gateway Protocol) doit être mis en oeuvre. Dans le cas d'Internet, c'est généralement BGP (Border Gateway Protocol) qui assume cette mission. BGP, protocole supporté par Zebra, constitue un vaste terrain d'investigation.

Chapitre 64. Le routage dynamique avec BGP

Initiation au routage BGP

64.1. Introduction

Internet relie des réseaux appartenant à des acteurs (entreprises, administrations, opérateurs de télécommunication, fournisseurs d´accès...) très différents. Il est peu probable qu´un consensus se dégage naturellement autour d´un même algorithme de routage dynamique et d´une même métrique. Ajoutons que des accords sont passés entre ces acteurs afin d´acheminer le trafic dans l´Internet et que celui-ci franchit des frontières, ce qui impose de respecter les réglementations locales. Le protocole BGP (Border Gateway Protocol) a été conçu pour répondre à ces problèmes. Comme toujours, nous allons aborder brièvement son fonctionnement avant de passer à une mise en pratique avec Zebra.

64.1.1. Les grands principes

BGP achemine les informations de routage entre les réseaux reliés à Internet.

64.1.1.1. Le concept de système autonome

Au sein d´une même organisation, les décisions concernant la topologie ou la politique de routage sont, en général, prises par une autorité unique. Par conséquent, le routage à l´intérieur de l´organisation est basé sur la confiance et un protocole de type IGP comme OSPF est mis en oeuvre. Un réseau fonctionnant sous une autorité unique est appelé un système autonome (AS). En pratique, un AS regroupe un ou plusieurs réseaux et un ou plusieurs routeurs ainsi que le montre l´exemple de la figure 1 :

Figure 64-1. Un système autonome constitué de réseaux

Les AS sont identifiés par un numéro sur 16 bits unique attribué par les mêmes organismes qui affectent les adresses IP. Il existe une plage de numéros d´AS privés de 64 512 à 65 535 pour ceux qui ne possèdent pas de numéro d´AS public.

À l´intérieur de votre AS, vous pouvez bien faire ce que vous voulez. Vous préférez tel protocole de routage ? Vous estimez que telle métrique est plus pertinente ? Tant mieux, cela ne regarde que vous. La seule contrainte, c´est que vous devrez désigner un ou plusieurs routeurs, à la frontière de votre AS, pour propager les informations d´accessibilité de vos réseaux et collecter les informations d´accessibilité des autres ASBR.

Par exemple, si l´on reprend la topologie présentée dans la séquence sur OSPF, que l´on en fait un AS, on obtiendrait :

Figure 64-2. Un AS découpé en zones OSPF

Ce système autonome utilise OSPF en tant qu´IGP. Il est découpé en trois zones. R4 est le routeur de bordure de zone. Celui-ci doit fonctionner sous BGP.

64.1.1.2. Les politiques de routage

Internet est un maillage de réseaux. Cela signifie que plusieurs chemins existent entre deux réseaux d´extrémité (figure 3). Mais ceux-ci ne sont pas tous équivalents.

Figure 64−3. Réseaux d'AS

En effet, imaginez vos datagrammes circulant dans cette jungle cruelle et barbare qu´est Internet. Vous préfèrerez peut être qu´ils transitent par tel AS, car l´administrateur est un pote ou que vous avez négocié un tarif avantageux. Vous refuserez peut−être qu´ils transitent par tel AS car vos flux sont incompatibles avec la législation en vigueur dans le pays ou pour toute autre considération politique, économique ou de sécurité que l´on peut imaginer. Dans ce cas, vous aurez besoin de définir une politique de routage sur Internet. BGP vous rendra ce service.

Dans le réseau, l´administrateur de l´AS 600 définit une politique de routage. Il veut joindre l´AS 200 en passant par les AS 300 et 100.

64.1.1.3. Les informations de routage échangées par BGP

Une des particularités de BGP−IV (RFC 1771) est qu´il s'appuie sur la couche TCP (port 179), ce qui permet de s'affranchir de la nécessité de supporter les fonctions de fragmentation, de retransmission, d'acquittement et de séquencement.

Deux routeurs BGP s´échangent des messages pour ouvrir et maintenir la connexion. Le premier flot de données est la table entière de routage. Ensuite, des mises à jours incrémentielles sont envoyées lorsque la table de routage change : BGP ne nécessite pas de mises à jour périodiques des tables de routage. Par contre, un routeur BGP doit retenir la totalité des tables de routage courantes de tous ses pairs durant le temps de la connexion. Des messages « keepalive » sont envoyés périodiquement pour maintenir la connexion.

BGP est un protocole de type " Path Vector ". Les routeurs s´échangent des informations du type :

Adresse IP du réseau de destination	Adresse IP du prochain routeur (next hop)	Liste des AS traversés pour atteindre le réseau

On constate qu´avec BGP, la granularité du routage est l´AS. Par défaut, lorsque plusieurs route existent entre deux AS, BGP choisit la route qui traverse le moins d'AS.

64.1.1.4. Quand utiliser BGP ?

Ce protocole est généralement utilisé par les fournisseurs d´accès à Internet, les administrateurs des points d´échange (IXP) et les administrateurs de systèmes autonomes qui souhaitent mettre en oeuvre leur propre politique de routage.

64.1.2. Place à la pratique

Voyons sur quelle topologie nous allons travailler :

Figure 64−4. Topologie

Nous avons trois systèmes autonomes numérotés 10, 20 et 30. Dans un souci de simplification, chaque système autonome a une topologie relativement simple. De ce fait, aucun IGP n'est utilisé. On constate que chaque routeur est un ASBR. L'AS 20 a une particularité car pour les AS 10 et 30, c'est un AS de transit. Nous reviendrons sur ce point dans la mise en oeuvre.

64.1.2.1. Tâches de configuration

Sur chaque routeur, il faudra réaliser les tâches suivantes :

0. préparer des fichiers `/etc/zebra/zebra.conf` et /`etc/zebra/bgpd.conf` rudimentaires, ne comportant que le nom de l'hôte et le mot de passe pour accéder à la console d'administration. Ensuite, dans le shell Linux, il faut lancer les démons zebra et `bgpd` dans l'ordre ;

1. activer le processus de routage BGP en indiquant dans quel AS de situe le routeur (commande `router bgp <numero_AS>`) ;
2. spécifier les routes à annoncer via BGP (commande `network <préfixe_IP_du_réseau>/<masque>`) ;
3. établir une relation avec les routeurs voisins (commande `neighbor <IP_du_voisin> remote−as <AS_du_voisin>`).

64.1.2.2. Situation de départ

Vous devez créer des fichiers de configuration pour zebra (`/etc/zebra/zebra.conf`) et bgpd (`/etc/zebra/bgpd.conf`) rudimentaires sur chaque routeur. Par exemple, pour R1 :

− fichier zebra.conf :

```
hostname R1(ZEBRA)
password foo
```

− fichier bgpd.conf :

```
hostname R1(BGP) password foo
```

Vous devez ensuite démarrer (ou redémarrer), les deux démons zebra et bgpd sur chaque routeur. Enfin, sur vous entrez dans le terminal d'administration de bgpd via le port telnet 2605. Par exemple, sur R1 :

Linux# telnet localhost 2605 Hello, this is zebra (version 0.91a). Copyright 1996−2001 Kunihiro Ishiguro. User Access Verification Password: R1(BGP)> enable R1(BGP)#

Si vous avez envie de suivre précisément les échanges de messages entre routeurs, Zebra propose un mécanisme de débogage grâce à la commande `debug`. Supposons que nous voulions garder une trace de toutes les mises à jour d'informations de routage BGP émises et reçues par R1 :

```
R1(BGP)# conf t
R1(BGP)(config)# log file /var/log/zebra/bgpd.log
R1(BGP)(config)# debug bgp  updates
```

Il faut s'assurer que le répertoire `/var/log/zebra` existe.

64.1.2.3. Activation du processus de routage

Dans le mode « config », nous allons activer le processus BGP en n'oubliant pas d'indiquer le numéro d'AS. Par exemple, sur R1 :

```
R1(BGP)(config)# router bgp 10
R1(BGP)(config−router)#
```

64.1.2.4. Spécification des routes à annoncer

Pour R1, le seul réseau que l'on souhaite annoncer via BGP aux autres routeurs est 191.10.0.0/24 car il contient des ordinateurs :

```
R1(BGP)(config−router)# network 191.10.0.0/24
```

Cette commande est à renouveler autant de fois qu'il y a de réseaux à annoncer. Dans notre exemple, nous avons choisi volontairement de ne pas annoncer le réseau 10.0.0.0/8 car celui−ci ne contient que des routeurs. Vous noterez au passage que cette commande n'a pas la même vocation que dans RIP ou OSPF.

64.1.2.5. Établissement d'une connexion avec les voisins

Il faut distinguer deux cas : mon voisin est−il dans un autre AS ou dans le même AS que moi ?

1. Cas où le voisin est dans un autre AS :

Je rappelle que "un voisin" est un routeur avec lequel on est immédiatement connecté. Il faut indiquer son adresse IP ainsi que son numéro d'AS. Dans le jargon BGP, une connexion entre deux routeurs BGP s'appelle peering. Sur R1, on saisit la commande :

```
R1(BGP)(config−router)# neighbor 10.0.0.2 remote−as 20
```

Cette commande est à renouveler autant de fois qu'il y a de routeurs BGP immédiatement connectés (et avec qui on souhaite échanger des informations de routage bien sûr).

2. Cas où le voisin est dans le même AS :

L'AS 20 est particulier car il véhicule des données qui ne lui sont pas destinées, en particulier les informations de routage en provenance et à destination des AS 10 et 30. En effet, pour que l'AS 30 connaisse l'existence du réseau 191.10.0.0/24, R2 doit récupérer cette information auprès de R1 puis la transmettre à R3 qui lui−même la transmettra à R4. R2 et R3 étant dans le même AS, un "sous−protocole" de BGP appelé iBGP est mis en oeuvre automatiquement par le routeur. Mais, celui−ci a la particularité de ne pas modifier les next hop lorsqu'il relaie des informations de routage. Ainsi, sans intervention, R4 apprendrait que le next hop pour joindre 192.10.0.0/24 est 10.0.0.1. Or, celui−ci n'est pas joignable depuis l'AS 30 (puisque nous ne faisons pas de `network 10.0.0.0/8`). Pour pallier à cette lacune, il existe la directive `next−hop−self`. Ainsi sur R2 et sur R3 nous réaliserons les configurations suivantes :

```
R2(BGP)(config−router)# neighbor 20.0.0.2 remote−as 20
R2(BGP)(config−router)# neighbor 20.0.0.2 next−hop−self

R3(BGP)(config−router)# neighbor 20.0.0.1 remote−as 20
R3(BGP)(config−router)# neighbor 20.0.0.1 next−hop−self
```

64.1.2.6. Affichage de la configuration

Affichons la configuration complète de R1 :

```
R1(BGP)(config-router)# end
R1(BGP)# show running-config
Current configuration:
!
hostname R1(BGP)
password foo
log file /var/log/zebra/bgpd.log
!
debug bgp updates
!
router bgp 10
network 191.10.0.0/24
neighbor 10.0.0.2 remote-as 20
!
line vty
!
end
```

Affichons la configuration complète de R2 :

```
R2(BGP)# show running-config
Current configuration:
!
hostname R2(BGP)
password foo
!
router bgp 20
network  192.20.0.0/24
neighbor 10.0.0.1 remote-as 10
neighbor 20.0.0.2 remote-as 20
neighbor 20.0.0.2 next-hop-self
!
line vty
!
end
```

J´espère que vous avez la même configuration. Si une commande est incorrecte, vous devez faire comme si vous vouliez la saisir à nouveau mais en la faisant précéder de no.

Pour enregistrer la configuration, je vous rappelle que l´on saisit :

```
R1(OSPF)# copy running-config startup-config
```

Par symétrie, vous devez être en mesure de configurer les autres routeurs du réseau.

64.1.2.7. Affichage de l'état des routeurs

64.1.2.7.1. Le débogage

Si vous avez activé le débogage, le fichier de log doit vous révéler des informations de ce style :

```
R1 # cat /var/log/bgpd.log

2003/05/24 03:29:03
   BGP: BGPd 0.91a starting: vty@2605, bgp@179
2003/05/24 03:29:09
   BGP: 10.0.0.2 send UPDATE 191.10.0.0/24 nexthop 10.0.0.1,
   origin i,  mp_nexthop ::, path
2003/05/24 03:29:09
   BGP: 10.0.0.2 rcvd UPDATE w/  attr: nexthop 10.0.0.2,
   origin i, path
202003/05/24 03:29:09
   BGP: 10.0.0.2 rcvd 192.20.0.0/24...
```

On voit les annonces de routes émises (send UPDATE) et reçues (recv UPDATE) par R1. Si votre routage ne fonctionne pas, vous trouverez dans les journaux toutes les informations nécessaires à la résolution de problèmes.

64.1.2.7.2. La table de routage

Observons, par exemple, la table BGP calculée par R2 :

```
R2(BGP)# show ip bgp
BGP table version is 0, local router ID is 192.20.0.254
Status codes:
    s suppressed, d damped, h history, * valid, > best, i - internal
    Origin codes: i - IGP, e -  EGP, ? - incomplete
```

	Network	Next Hop	Metric	LocPrf	Weight	Path
*>	191.10.0.0/24	10.0.0.1			0	10 i
*>	192.20.0.0	0.0.0.0			32768	i
*>i	193.20.0.0	20.0.0.2		100	0	i
*>i	194.30.0.0	20.0.0.2		100	0	30 i

```
Total number of prefixes 4
```

La première colonne indique si la route est valide (*), si BGP considère que c'est la meilleure (>) et si elle provient d'un routeur interne (i pour internal) à l'AS. Les paramètres Metric, LocPrf (Local Preference) et Weight sont utilisés par BGP pour lire la meilleure route vers une destination. BGP applique l'algorithme (simplifié) suivant :

1. BGP choisit la route avec le plus grand poids (weight). Par défaut, une route directement connectée se voit attribuer le poids 32768 ;
2. Si une route n'est toujours pas choisie, BGP choisit celle avec la plus grande "préférence locale". Par défaut, une route issue d'un routeur interne à l'AS se voit attribuer une préférence locale de 100 ;
3. Si une route n'est toujours pas choisie, BGP choisit une route de type "internal" ;
4. Si une route n'est toujours pas choisie, BGP retiens la route avec le chemin (As−Path) le plus court ;
5. Si une route n'est toujours pas choisie, BGP choisit la route avec la métrique la plus faible.

La colonne Path indique la liste des systèmes autonomes à traverser avant d'atteindre le réseau. Le i indique ici que les routes sont de type IGP. En fait, BGP regroupe sous ce terme les routes statiques et découvertes par un protocole dynamique tel RIP ou OSPF.

Vous pouvez obtenir des informations détaillées sur une route particulière en tapant :

```
R2(BGP)# show ip bgp 194.30.0.0
BGP routing table  entry for 194.30.0.0/24
Paths: (1 available, best #1, table Default−IP−Routing−Table) 30
       20.0.0.2 from 20.0.0.2 (193.20.0.254)
Origin IGP, localpref 100, valid, internal, best
Last update: Sat May 24 03:39:16 2003
```

64.1.2.7.3. Informations sur les voisins

Zebra propose de nombreuses commandes permettant de connaître l'état des voisins. Parmi les plus intéressantes, on trouve :

show ip bgp neighbors	Donne de nombreuses sur la connexion avec les voisins
show ip bgp summary	Synthétise les informations ci−dessus dans un tableau
show ip bgp <IP_voisin> routes	Routes découvertes à partir de ce voisin
show ip bgp <IP_voisin> advertised−routes	Routes annoncées à ce voisin

64.1.3. Cohabitation entre BGP et les IGP

Zebra peut tout à fait annoncer des routes dynamiques via BGP (commande `redistribute`). Toutefois, cette fonctionnalité est à envisager avec beaucoup de prudence car toute changement de topologie sera visible à l'extérieur du système autonome.

64.1.4. Conclusion

CIDR

Route server

Route reflector

Cluster

Chapitre 65. TP sur le routage statique avec Zebra

> Le document décrit les principes du routage IP et l'architecture du routeur logiciel Zebra. Il propose une prise en main de Zebra et une mise en pratique du routage statique.

65.1. Introduction

Description et objectifs de la séquence. Une fois la séquence finie :

1. vous saurez utiliser l'interface de configuration de Zebra ;
2. saisir des routes statiques ;

3. déboguer un routage incorrect.

65.1.1. Présentation des concepts importants

65.1.1.1. Routage, routeur, route

Imaginez que vous souhaitiez rendre visite à quelqu'un. Vous connaissez son adresse et, dans votre tête, vous avez la plupart des éléments importants pour vous diriger (sortir de la maison, prendre à droite, au feu tourner à gauche, marcher jusqu'au rond−point, tourner à droite, etc.).

Dans l'univers informatique, c'est sensiblement différent. Lorsqu'un ordinateur émet un paquet de données à destination d'un ordinateur situé dans un autre réseau, il ne sait pas quelle route il va prendre. La seule chose connue est l'adresse de destination du paquet ainsi que l'adresse d'une machine à proximité, située dans son réseau, et qui joue le rôle de la porte. On l'appelle la passerelle. Elle oriente le paquet vers le prochain carrefour. À ce carrefour, il y a un appareil qui oriente le paquet vers le prochain carrefour. À ce nouveau carrefour, il y a un nouvel appareil qui oriente le paquet vers le prochain carrefour. Et ainsi de suite jusqu'à arriver à la passerelle du réseau de destination qui remet le paquet dans le réseau.

L'appareil dont nous parlons s'appelle un routeur (la passerelle est également un routeur). Le routage consiste à faire circuler de routeur en routeur les paquets de données. L'administration d'un routeur consiste à configurer les routes d'un routeur. Une route est définie par un réseau de destination et l'adresse d'un routeur voisin, prochaine étape vers le réseau de destination. Une table de routage est une liste de routes utilisée par le routeur pour prendre des décisions quant à la direction à donner pour un paquet reçu sur l'une de ses interfaces. Le routage statique par opposition au routage dynamique, consiste à saisir manuellement les routes dans le routeur.

65.1.1.2. Prise de décision

Lorsqu' un routeur reçoit un paquet sur l'une de ses interfaces. Il extrait l'adresse IP de destination. En appliquant le masque réseau, il détermine le préfixe réseau de cette adresse. Il recherche ce préfixe dans sa table de routage. Si le préfixe est trouvé, il émet le paquet sur l'interface réseau adapté. Si le préfixe n'est pas trouvé, deux cas peuvent se présenter. S'il connaît un routeur par défaut, il lui transmet le paquet, sinon il le détruit (sans en informer l'émetteur).

65.1.2. Architecture de Zebra

Zebra fonctionne sous Linux et BSD. C'est un routeur multi−protocole composé d'une suite de démons, un par protocole de routage dynamique plus un démon central (zebra) utilisé pour le routage statique. De plus, lorsque le routage dynamique est mis en oeuvre, il est chargé de synthétiser dans une table de routage unique, les informations rapportées par les autres démons.

Figure 65−1. Architecture de Zebra

65.1.3. Topologie de travail

Figure 65−2. Topologie 1

65.1.4. Mise en place

Nous avons besoin de trois routeurs disposant chacun de trois interfaces réseau. Sur une interface, un réseau local sera connecté. Les liens représentés par des éclairs correspondent à un réseau étendu. Ils sont simulés avec un câble croisé Ethernet. Les routeurs sont directement reliés de carte réseau à carte réseau par le câble croisé.

Vous devez installer trois ordinateurs avec trois cartes réseau dans chacun. Ensuite, installez le système d'exploitation puis Zebra. Pour les distributions RedHat et Mandrake, il est possible de télécharger les binaires sur http:\\rpmfind.net. Sinon, les sources sont accessibles sur www.zebra.org.

Les cartes réseau doivent être actives, mais ne configurez pas les adresses des cartes réseau, vous pourrez le faire directement à partir de Zebra. Si vous faites un ifconfig vous devez obtenir :

```
[root@linux root]# ifconfig
eth0      Lien encap:Ethernet  HWaddr 00:50:56:40:C0:63
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 lg file transmission:100
          RX bytes:0 (0.0 b)  TX bytes:38159 (0.0 b)
          Interruption:9 Adresse de base:0x1060

eth1      Lien encap:Ethernet  HWaddr 00:50:56:40:C0:64
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
```

```
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 lg file transmission:100
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
          Interruption:11 Adresse de base:0x1080

eth2      Lien encap:Ethernet  HWaddr 00:50:56:40:C0:65
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 lg file transmission:100
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
          Interruption:10 Adresse de base:0x10a0

lo        Lien encap:Boucle locale
          inet adr:127.0.0.1  Masque:255.0.0.0
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 lg file transmission:0
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
```

Les trois cartes Ethernet activées mais sans adresse réseau.

65.1.5. Démarrage du démon zebra

Zebra ne peut pas démarrer sans un fichier de configuration minimal qu'il faudra saisir sur chaque routeur. Éditez un fichier /etc/zebra/zebra.conf :

```
Linux# vi /etc/zebra/zebra.conf
hostname Zebra
password foo
```

Ensuite, chargez le démon zebra :

```
Linux# zebra −d
```

Vous obtiendrez peut−être l'avertissement suivant :

```
2003/02/19 01:08:17 ZEBRA:
   can't create router advertisement socket:
   Address family not supported by protocol
```

Zebra supporte IPv6, mais ce protocole n'est pas activé sur votre noyau. Ce n'est pas gênant pour la suite de l'activité.

65.1.6. Connexion au démon zebra

Zebra dispose d'une interface telnet pour chaque démon. Pour configurer le démon zebra, il faut se connecter sur le port 2601. Par exemple, sur Routeur3 :

```
[root@linux root]# telnet localhost 2601
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.

Hello, this is zebra (version 0.91a).
Copyright 1996−2001 Kunihiro Ishiguro.


User Access Verification

Password:
Zebra>
```

À l'invite Password:, saisissez le mot de passe que vous avez indiqué dans le fichier /etc/zebra/zebra.conf (aucun caractère n'apparaît lors de la saisie, c'est " normal "). L'invite devient Zebra> (Zebra est le nom par défaut que vous changerez plus tard).

65.1.7. Prise en main de Zebra (principe)

Voyons les principaux éléments de l'environnement de Zebra.

65.1.7.1. L'invite

L'invite (prompt) a une importance capitale. À la suite du nom de l'appareil, on peut savoir dans quel " méandre " de l'arborescence des menus de configuration on se trouve.

Une invite terminée par un > indique que vous êtes dans le mode VIEW : c'est le mode activé lorsque l'on se connecte. Comme son nom l'indique, il s'agit d'un mode de visualisation de l'état du routeur. Par exemple :

```
Zebra>
```

Une invite terminée par un # indique que vous êtes dans le mode ENABLE : c'est le mode privilégié qui permet de modifier la configuration du routeur. On y accède à partir du mode VIEW en tapant la commande enable. Par exemple :

```
Zebra> enable
Zebra#
```

Ensuite, vous travaillerez essentiellement avec deux sous−modes :

− le mode "terminal de configuration". On y accède à partir du mode ENABLE en tapant la commande configure terminal. Dans ce mode, on peut (entre autre) saisir des routes statiques. Par exemple :

```
Zebra# configure terminal
Zebra(config)#
```

− le mode "interface". On y accède à partir du mode "terminal de configuration" en tapant la commande interface suivi du nom de l'interface réseau. Dans ce mode, on peut (entre autre) saisir l'adresse IP et le masque de l'interface. Par exemple :

```
Zebra(config)# interface eth0
Zebra(config−if)#
```

65.1.7.2. L'aide en ligne

Retenez les éléments suivants, ils vous seront d'un grand secours :

1. − comme dans l'interpréteur de commandes Linux, le logiciel complète toutes les commandes lorsque vous appuyez sur la touche TAB (par exemple : shTAB devient show);
2. − un simple appui sur ? indique toutes les commandes disponibles dans le mode dans lequel vous êtes;
3. − enfin, la commande list donne la liste de toutes les commandes disponibles disponibles dans le mode dans lequel vous êtes et de leurs paramètres.

65.1.8. Prise en main de Zebra (mise en pratique)

Si vous n'êtes pas dans le mode view, tapez end puis disable :

```
Zebra(config−if)# end
Zebra# disable
Zebra>
```

65.1.8.1. Utilisation de l'aide en ligne

− listez l'ensemble des commandes disponibles à ce stade (tapez simplement un point d'interrogation) :

```
Zebra> ?
  enable    Turn on privileged mode command
  exit      Exit current mode and down to previous mode
  help      Description of the interactive help system
  list      Print command list
  quit      Exit current mode and down to previous mode
  show      Show running system information
  terminal  Set terminal line parameters
  who       Display who is on vty
```

− listez toutes les commandes disponibles à ce stade avec tous leurs paramètres :

```
Zebra> list
  enable
  exit
  help
  list
  quit
  show debugging zebra
  show history
  show interface [IFNAME]
  show ip forwarding
  show ip route
  show ip route (bgp|connected|kernel|ospf|rip|static)
  show ip route A.B.C.D
  show ip route A.B.C.D/M
...
```

− listez tous les paramètres de la commande qui permet d'afficher la configuration (show ?) :

```
Zebra> show ?
  debugging  Zebra configuration
  history    Display the session command history
  interface  Interface status and configuration
  ip         IP information
  ipv6       IPv6 information
  memory     Memory statistics
  table      default routing table to use for all clients
```

```
  version    Displays zebra version
```

65.1.8.2. Affichage de l'état du routeur

− Listez les interfaces disponibles sur votre routeur avec la commande show interface (n'oubliez pas qu'un simple shTAB intTAB suffit !) :

```
Zebra> sh int
Interface lo
  index 1 metric 1 mtu 16436 >UP,LOOPBACK,RUNNING>
  inet 127.0.0.1/8
    input packets 152, bytes 9574, dropped 0, multicast packets 0
    input errors 0, length 0, overrun 0, CRC 0, frame 0, fifo 0, missed 0
    output packets 152, bytes 9574, dropped 0
    output errors 0, aborted 0, carrier 0, fifo 0, heartbeat 0, window 0
    collisions 0
Interface eth0
  index 2 metric 1 mtu 1500 >UP,BROADCAST,RUNNING,MULTICAST>
  HWaddr: 00:50:56:40:c0:63
    input packets 0, bytes 0, dropped 0, multicast packets 0
    input errors 0, length 0, overrun 0, CRC 0, frame 0, fifo 0, missed 0
    output packets 0, bytes 0, dropped 0
    output errors 0, aborted 0, carrier 0, fifo 0, heartbeat 0, window 0
    collisions 0
Interface eth1
  index 3 metric 1 mtu 1500 >UP,BROADCAST,RUNNING,MULTICAST>
  HWaddr: 00:50:56:40:c0:64
    input packets 0, bytes 0, dropped 0, multicast packets 0
    input errors 0, length 0, overrun 0, CRC 0, frame 0, fifo 0, missed 0
    output packets 0, bytes 0, dropped 0
    output errors 0, aborted 0, carrier 0, fifo 0, heartbeat 0, window 0
    collisions 0
Interface eth2
  index 4 metric 1 mtu 1500 >UP,BROADCAST,RUNNING,MULTICAST>
  HWaddr: 00:50:56:40:c0:65
    input packets 0, bytes 0, dropped 0, multicast packets 0
    input errors 0, length 0, overrun 0, CRC 0, frame 0, fifo 0, missed 0
    output packets 0, bytes 0, dropped 0
    output errors 0, aborted 0, carrier 0, fifo 0, heartbeat 0, window 0
    collisions 0
```

Vous avez quatre interfaces (lo et trois Ethernet), aucune ne possède pour l'instant d'adresse IP.

− Visionnez le contenu de votre table de routage : shTAB ip roTAB :

```
Zebra> sh ip ro
Codes: K − kernel route, C − connected, S − static, R − RIP, O − OSPF,
       B − BGP, > − selected route, * − FIB route

C>* 127.0.0.0/8 is directly connected, lo
```

Elle est vide ! sauf 127.0.0.0/8 qui correspond à l'adresse de bouclage logiciel. Cette table étant vide, le routeur n'a aucune idée de la façon d'acheminer les paquets de données vers tel ou tel réseau.

Le but de ce TP est de configurer correctement les interfaces et les tables de routage pour que l'ensemble des appareils des stations puissent s'atteindre au travers des trois routeurs.

65.1.8.3. Configuration générale du routeur

− Toute configuration nécessite de passer en mode privilégié :

```
Zebra> enable
Zebra#
```

Le # indique que vous êtes en mode privilégié.

− Listez les commandes disponibles :

```
Zebra# ?
  configure  Configuration from vty interface
  copy       Copy configuration
  debug      Debugging functions (see also 'undebug')
  disable    Turn off privileged mode command
  end        End current mode and change to enable mode.
  exit       Exit current mode and down to previous mode
  help       Description of the interactive help system
  list       Print command list
  no         Negate a command or set its defaults
  quit       Exit current mode and down to previous mode
  show       Show running system information
  terminal   Set terminal line parameters
  who        Display who is on vty
  write      Write running configuration to memory, network, or terminal
```

Les commandes importantes à ce stade sont : copy pour enregistrer la configuration et configure pour accéder au terminal de configuration des interfaces réseau et des routes.

65.1.8.4. Affichage de la configuration actuelle

La commande write term permet de consulter à tout moment la configuration actuelle du routeur :

```
Zebra# write term

Current configuration:
!
hostname Zebra
password foo
!
interface lo
!
interface eth0
!
interface eth1
!
interface eth2
!
line vty
!
end
```

65.1.8.5. Passage au mode "terminal de configuration"

```
Zebra# conf term
Zebra(config)#
```

Définition du nom du routeur. Cela se fait par la commande hostname. Attribuez à chaque routeur un nom significatif. Par exemple, pour le routeur 3 :

```
Zebra(config)# hostname Routeur3(Zebra)
Routeur3(Zebra)(config)#
```

Sécurisation du routeur. L'accès au routeur et à son mode de configuration doit être protégé.

1. − Activez le chiffrement des mots de passe : Lorsque nous avons utilisé la commande write term, nous avons pu constater que le mot de passe n'était pas chiffré. Résolvons ce problème :

   ```
   Routeur3(Zebra)(config)# service password-encryption
   ```
2. − Attribuez un mot de passe pour l'accès au mode VIEW du routeur (en remplacement de "foo"). Il doit commencer par une lettre ou un chiffre :

   ```
   Routeur3(Zebra)(config)# password ******
   ```
3. − Attribuez un mot de passe pour l'accès au mode ENABLE du routeur. Il doit commencer par une lettre ou un chiffre :

   ```
   Routeur3(Zebra)(config)# enable password ******
   ```

65.1.8.6. Consultez la configuration

```
Routeur3(Zebra)(config)# w t

Current configuration:
!
hostname Routeur3(Zebra)
password 8 kZujW/HWOwF4o
enable password 8 1b5pHVyKjR.5w
service password-encryption
!
interface lo
!
interface eth0
!
interface eth1
!
interface eth2
!
line vty
!
end
```

Les mots de passe sont chiffrés maintenant.

65.1.8.7. Enregistrement de la configuration

Il faut revenir au mode enable. N'oubliez pas de sauvegarder régulièrement !

```
Routeur3(Zebra)(config)# end
Routeur3(Zebra)# copy running-config startup-config
Configuration saved to /etc/zebra/zebra.conf
```

Configuration des interfaces réseau

Il faut attribuer une adresse IP et un masque à chaque carte réseau. Pour cela, il faut se placer dans le mode "configuration d'interface" puis utiliser une commande ip address. Enfin, il faut activer l'interface avec une commande no shutdown. Par exemple, pour l'interface eth0 du routeur 3 :

```
Routeur3(Zebra)# conf t
Routeur3(Zebra)(config)# int eth0
Routeur3(Zebra)(config-if)# ip addr 192.168.3.254/8
Routeur3(Zebra)(config-if)# no shut
```

− Faites la même manipulation, en adaptant, pour les autres interfaces de ce routeur.

Consultez la configuration :

```
Routeur3(Zebra)(config)# wr t

Current configuration:
!
hostname Routeur3(Zebra)
password 8 kZujW/HWOwF4o
enable password 8 1b5pHVyKjR.5w
service password-encryption
!
interface lo
!
interface eth0
 ip address 192.168.3.254/24
!
interface eth1
 ip address 12.0.0.2/8
!
interface eth2
!
line vty
!
end
```

L'interface eth2 n'a pas d'adresse pour l'instant.

Faites ces configurations sur les trois routeurs.

65.1.8.8. Configuration des routes

Allez dans le shell Linux du routeur3 et listez sa table de routage :

```
[root@linux root]# route
Table de routage IP du noyau
Destination     GW  Genmask         Indic Metric Ref    Use Iface
192.168.3.0     *   255.255.255.0   U     0      0        0 eth0
12.0.0.0        *   255.0.0.0       U     0      0        0 eth1
127.0.0.0       *   255.0.0.0       U     0      0        0 lo
```

Cet appareil connaît les adresses des réseaux directement connecté. Ainsi, Routeur3 peut communiquer avec l'interface de Routeur2 situé dans le même réseau :

```
[root@linux root]# ping 12.0.0.1 -c 2
PING 12.0.0.1 (12.0.0.1) from 12.0.0.2 : 56(84) bytes of data.
Warning: time of day goes back, taking countermeasures.
64 bytes from 12.0.0.1: icmp_seq=0 ttl=255 time=4.352 msec
64 bytes from 12.0.0.1: icmp_seq=1 ttl=255 time=1.819 msec

--- 12.0.0.1 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max/mdev = 1.819/3.085/4.352/1.267 ms
```

Tous les appareils peuvent pinger leurs voisins immédiats, mais pas au−delà. Par exemple, Routeur3 ne peut pas atteindre Routeur1 :

```
[root@linux root]# ping 11.0.0.1
connect: Network is unreachable
```

Normal, puisque le réseau 11.0.0.0 ne figure pas dans sa table de routage. Le routage statique va consister à intégrer manuellement les routes dans la table de routage du routeur. Ouvrez un session telnet avec zebra puis allez dans le mode "configuration de terminal".

La saisie des routes se fait avec la commande ip route. Son format général est :

```
ip route <adresse_reseau_destination> <adresse_ip_prochain_routeur>
```

Par exemple, sur Routeur3 pour atteindre Routeur1, il faudra indiquer 11.0.0.0/8 en réseau de destination et l'adresse IP de l'interface de Routeur2 qui est située du côté de Routeur3 :

```
Routeur3(Zebra)(config)# ip route 11.0.0.0/8 12.0.0.1
```

Tentons à nouveau un ping dans le shell Linux :

```
Routeur3(Zebra)(config)# end
Routeur3(Zebra)# quit
Connection closed by foreign host.
[root@linux root]# ping 11.0.0.1
PING 11.0.0.1 (11.0.0.1) from 12.0.0.2 : 56(84) bytes of data.

CTRL−C

−−− 11.0.0.1 ping statistics −−−
13 packets transmitted, 0 packets received, 100% packet loss

Il faut interrompre la commande car elle restait bloquée. Cela ne marche donc pas. Pourquoi ?
```

Pour comprendre, nous allons utiliser la commande Linux tcpdump qui affiche tous les paquets passant par une interface. Sur Routeur3 :

```
[root@linux root]# tcpdump −i eth1
tcpdump: listening on eth1
03:44:45.507835 12.0.0.2 > 11.0.0.1: icmp: echo request (DF)
03:44:46.507835 12.0.0.2 > 11.0.0.1: icmp: echo request (DF)
03:44:47.507835 12.0.0.2 > 11.0.0.1: icmp: echo request (DF)
```

Le routeur émet des paquets mais on constate qu'il n'y a pas de retour. Le problème peut venir de Routeur2 ou de Routeur1. Suivons la chaîne et connectons−nous dans le shell de Routeur2. Utilisons tcpdump sur l'interface du côté de Routeur3 :

```
[root@linux root]# tcpdump −i eth2
tcpdump: listening on eth2
03:52:20.017855 12.0.0.2 > 11.0.0.1: icmp: echo request (DF)
03:52:21.017855 12.0.0.2 > 11.0.0.1: icmp: echo request (DF)
03:52:22.027855 12.0.0.2 > 11.0.0.1: icmp: echo request (DF)
```

Les paquets sont bien reçus. Faisons la même manipulation, toujours sur Routeur2 mais sur l'interface du côté de Routeur1 :

```
[root@linux root]# tcpdump −i eth1
tcpdump: listening on eth1
03:52:46.027855 12.0.0.2 > 11.0.0.1: icmp: echo request (DF)
03:52:47.017855 12.0.0.2 > 11.0.0.1: icmp: echo request (DF)
03:52:48.017855 12.0.0.2 > 11.0.0.1: icmp: echo request (DF)
```

Les paquets sont bien transmis d'une interface à l'autre. Routeur2 fait son travail. Passons sur Routeur1, faisons un tcpdump sur l'interface côté Routeur 2 :

```
[root@linux root]# tcpdump −i eth2
tcpdump: listening on eth2
03:53:53.224868 12.0.0.2 > 11.0.0.1: icmp: echo request (DF)
03:53:54.224868 12.0.0.2 > 11.0.0.1: icmp: echo request (DF)
03:53:55.234868 12.0.0.2 > 11.0.0.1: icmp: echo request (DF)
```

Les paquets sont bien reçus sur Routeur1. Mais il n'est pas capable d'y répondre. D'où vient le problème alors ? Il n'y a qu'une solution. Il vient de la table de routage. Routeur1 ne sait pas comment répondre à Routeur3 car il n'a aucune idée de l'endroit où il se trouve puisqu'il n'est pas directement relié à lui. Connectons−nous au démon zebra sur Routeur1 puis visionnons sa table de routage :

```
Routeur1(Zebra)> sh ip ro
Codes: K − kernel route, C − connected, S − static, R − RIP, O − OSPF,
       B − BGP, > − selected route, * − FIB route

C>* 11.0.0.0/8 is directly connected, eth2
C>* 127.0.0.0/8 is directly connected, lo
C>* 192.168.1.0/24 is directly connected, eth0
```

Elle ne contient aucune entrée pour 12.0.0.0/8, le réseau auquel est connecté Routeur3. Réparons cette erreur en informant Routeur1 sur la façon d'atteindre 12.0.0.0/8. Il faut passer par son voisin Routeur2 dont l'adresse est 11.0.0.2. Sur le routeur (en mode "terminal de configuration"), il faut saisir :

```
Routeur1(Zebra)(config)# ip route 12.0.0.0/8 11.0.0.2
```

Affichons à nouveau la table de routage :

```
Routeur1(Zebra)(config)# end
Routeur1(Zebra)# sh ip ro
Codes: K − kernel route, C − connected, S − static, R − RIP, O − OSPF,
       B − BGP, > − selected route, * − FIB route

C>* 11.0.0.0/8 is directly connected, eth2
S>* 12.0.0.0/8 [1/0] via 11.0.0.2, eth2
C>* 127.0.0.0/8 is directly connected, lo
C>* 192.168.1.0/24 is directly connected, eth0
```

La route apparaît. Elle est notée S pour Statique.

Revenons dans le shell Linux de Routeur1. La commande tcpdump nous montre que maintenant, Routeur sait répondre à Routeur3 :

```
[root@linux root]# tcpdump −i eth2
tcpdump: listening on eth2
03:59:51.904868 12.0.0.2 >> 11.0.0.1: icmp: echo request (DF)
03:59:51.904868 11.0.0.1 > 12.0.0.2: icmp: echo reply
03:59:52.864868 12.0.0.2 > 11.0.0.1: icmp: echo request (DF)
03:59:52.864868 11.0.0.1 > 12.0.0.2: icmp: echo reply
```

Si l'on revient sur Routeur3, le ping fonctionne désormais sans problème :

```
[root@linux root]# ping 11.0.0.1 −c 2
PING 11.0.0.1 (11.0.0.1) from 12.0.0.2 : 56(84) bytes of data.
Warning: time of day goes back, taking countermeasures.
64 bytes from 11.0.0.1: icmp_seq=0 ttl=254 time=8.090 msec
64 bytes from 11.0.0.1: icmp_seq=1 ttl=254 time=3.986 msec

−−− 11.0.0.1 ping statistics −−−
2 packets transmitted, 2 packets received, 0% packet loss
round−trip min/avg/max/mdev = 3.986/6.038/8.090/2.052 ms
```

65.1.8.9. Résumé

1. Le routage est une chaîne : les paquets sont transmis au routeur voisin, qui transmet au routeur voisin, etc.

j. Il faut penser à configurer tous les routeurs pour l'aller ET le retour des paquets.

65.1.8.10. Configuration des routeurs

Maintenant que vous savez tout, saisissez sur chaque routeur l'ensemble minimal des routes qu'il a besoin de connaître. Le cahier des charges est le suivant : les stations de chaque réseau ont besoin de communiquer avec les stations de tous les autres réseaux. Les stations n'ont pas besoin de communiquer avec les routeurs.

D'abord, commencez par détruire les routes statiques que nous venons de saisir pour les besoins de la démonstration. Par exemple, sur Routeur3 pour détruire la route 11.0.0.0/8 :

```
Routeur3(Zebra)> en
Password:
Routeur3(Zebra)# conf t
Routeur3(Zebra)(config)# no ip route 11.0.0.0/8 12.0.0.1
Routeur3(Zebra)(config)#
```

Dernier point : n'oubliez pas de configurer la passerelle par défaut sur les stations, sans quoi elles ne sauraient pas comment faire sortir leurs paquets de LAN. La passerelle par défaut de chaque LAN est le routeur le plus proche. Par exemple sur une station du premier réseau local (192.168.1.0/24), en supposant que celle−ci est sous Windows :

Figure 65−3. Topologie 1

65.1.8.11. Solution

Ci−dessous, vous trouverez la configuration complète des trois routeurs. Vos routes statiques doivent être strictement identiques à celles indiquées. Ci ce n'est pas le cas, vous n'avez pas complètement compris le principe du routage, relisez la partie introductive de ce document.

Pour Routeur1 :

```
Routeur1(Zebra)# sh run
```

```
Current configuration:
!
hostname Routeur1(Zebra)
password 8 .A2UkN/mYwExA
enable password 8 J7XQRuHNCKhOA
service password-encryption
!
interface lo
!
interface eth0
 ip address 192.168.1.254/24
!
interface eth1
!
interface eth2
 ip address 11.0.0.1/8
!
ip route 192.168.2.0/24 11.0.0.2
ip route 192.168.3.0/24 11.0.0.2
!
line vty
!
end
```

Pour Routeur2 :

```
Routeur2(Zebra)# sh run

Current configuration:
!
hostname Routeur2(Zebra)
password 8 wcw4qNEU1QPy.
enable password 8 fqdY3GaicS4XQ
service password-encryption
!
interface lo
!
interface eth0
 ip address 192.168.2.254/24
!
interface eth1
 ip address 11.0.0.2/8
!
interface eth2
 ip address 12.0.0.1/8
!
ip route 192.168.1.0/24 11.0.0.1
ip route 192.168.3.0/24 12.0.0.2
!
line vty
!
end
```

Pour Routeur3 :

```
Routeur3(Zebra)# sh ru

Current configuration:
!
hostname Routeur3(Zebra)
password 8 kZujW/HWOwF4o
enable password 8 1b5pHVyKjR.5w
service password-encryption
!
interface lo
!
interface eth0
 ip address 192.168.3.254/24
!
interface eth1
 ip address 12.0.0.2/8
!
interface eth2
!
ip route 192.168.1.0/24 12.0.0.1
ip route 192.168.2.0/24 12.0.0.1
!
line vty
!
end
```

65.1.9. Problèmes rencontrés

Q − Je ne sais pas retirer les adresses IP déjà configurées sous Linux

R − Éditez le contenu des fichiers /etc/sysconfig/network−scripts/ifcfg−ethx (avec x = numéro de la carte) et supprimez les lignes contenant l'adresse et le masque. Ensuite, redémarrez.

Q − Je n'arrive pas ouvrir de session Telnet.

R − Soit vous n'indiquez pas le numéro de port (2601) après l'adresse IP, soit le démon zebra n'est pas lancé. Faites un ps −ax pour vérifier. S'il n'apparaît pas, faites un zebra −d

Q − J'ai oublié un mot de passe.

R − Bravo. Éditez le fichier /etc/zebra/zebra.conf et retirez les lignes password et service password−encryption

Q − J'ai saisi une mauvaise commande et je ne sais pas la supprimer :

R − Vous devez faire exactement comme si vous vouliez saisir à nouveau cette commande mais vous faites précéder le tout de no. Exemple : J'ai saisi une mauvaise adresse (99.0.0.9/8) sur l'interface eth0 de mon routeur :

```
Routeur1(Zebra)> enable
Password:
Routeur1(Zebra)# conf t
Routeur1(Zebra)(config)# int eth0
Routeur1(Zebra)(config−if)# no ip address 99.0.0.9/8
Routeur3(Zebra)(config−if)#
```

Chapitre 66. Multi−router looking glass

Utilisation de l'interface d'accès php aux services de zebra avec MRLG.

66.1. Présentation

MRLG était écrit en Perl. Il est désormais siponible en php. (http://pilot.org.ua/mrlg/). C'est une interface d'accès qui permet d'afficher les routes et interfaces des roueturs reconnus par zebra.

Figure 66−1. MRLG − Multi−Router Looking Glass

Chapitre 67. Annexe sur le langage de commande de Zebra

Utilisation du mode commande de Zebra.

67.1. Annexe sur le langage de commande de Zebra

Zebra est configuré (/etc/zebra/zebra.conf)

```
[mlx@mr]$ telnet localhost 2601
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.

Hello, this is zebra (version 0.94).
Copyright 1996−2002 Kunihiro Ishiguro.


User Access Verification

Password:      // Entrer le mot de passe
mr> enable     // On passe en mode privilégié
mr# disable    // On revient en mode normal
mr> exit       // Quitter
Connection closed by foreign host.

# Obtenir les commandes de bases
mr>?
  enable    Turn on privileged mode command
  exit      Exit current mode and down to previous mode
  help      Description of the interactive help system
  list      Print command list
  quit      Exit current mode and down to previous mode
  show      Show running system information
  terminal  Set terminal line parameters
  who       Display who is on vty

# Obtenir les commandes en mode privilégié
mr> enable
```

```
mr# ?
  configure  Configuration from vty interface
  copy       Copy configuration
  debug      Debugging functions (see also 'undebug')
  disable    Turn off privileged mode command
  end        End current mode and change to enable mode.
  exit       Exit current mode and down to previous mode
  help       Description of the interactive help system
  list       Print command list
  no         Negate a command or set its defaults
  quit       Exit current mode and down to previous mode
  show       Show running system information
  terminal   Set terminal line parameters
  who        Display who is on vty
  write      Write running configuration to memory, network, or terminal

# list donne la liste de toutes les commandes

mr# list
  configure terminal
  copy running-config startup-config
  debug zebra events
  debug zebra kernel
  debug zebra packet
  debug zebra packet (recv|send)
  debug zebra packet (recv|send) detail
  disable
  end
  exit
  help
  list
  no debug zebra events
  no debug zebra kernel
  no debug zebra packet
  quit
 --More--

# Une commande suivi de "?" indique la liste
# des paramètres attendus
mr# configure ?
  terminal  Configuration terminal
mr# configure
```

Les commandes "show running−config" et "show startup−config" permettent respectivement d'avoir la configuration active du routeur et les commandes de configuration de démarrage lors du démarrage du routeur.

"show interface" donne les indications sur les interfaces.

"show ip route" donne les information sur la table de routage.

"show history" donne la listes des commandes saisies.

"configure terminal" permt de passer en mode configuration.

Chapitre 68. Concepts généraux sur le routage

Éléments retrospectifs de la séquence sur le routage statique et dynamique.

68.1. Présentation

La notion d'adressage ip (réseaux, sous−réseaux, sur−réseaux, hôtes, du rôle et des fonctions de la couche réseau, des protocoles ip, arp, icmp, ont déjà été largement abordées dans d'autres séquences. Elles ne seront pas revues dans ce document.

La notion de routage statique a également également vu dans d'autres séquences. Les principes ne seront pas repris. On retiendra juste qu'avec cette technique, les protocoles de routage n'ont pas le choix de leurs routes. Cette technique convient bien aux petits réseaux ne subissant pas de d'évolutions ou de changements fréquents. Il ne s'agit pas vraiment d'un protocole de routage au sens ou cela peut être pris pour le routage dynamique.

Les protocoles de routage dynamiques répondent à d'autres besoins. Dès que les topologies deviennent complexes. Ce document va présenter un peu du jargon de ce domaine.

68.2. Jargon réseau sur le routage

Cette partie décrit les termes et éléments utilisés sur les réseaux mettant en oeuvre des protocoles de routages dynamiques.

68.2.1. Notion de système autonome (SA)

Un Système Autonome (AS) est un ensemble cohérent de réseaux et de routeurs sous la responsabilité d'une autorité administrative. Les AS ont des architectures de routages indépendantes les unes des autres.

Un AS ont des numéros codés sur 16 bits. Ces numéros attribués par le Network Information Center sont uniques.

Figure 68−1. Système Autonomes

Le schémas représente 2 AS. Les AS sont reliés par des routeurs que l'on nomme "exteriors gateway".

Les IG n'échangent que des informations entre elles. Certaines IG doivent cependant échanger avec des "exteriors gateway" pour prendre connaissance des réseaux étrangers.

Dans les protocoles de routages ds IGP's, on trouve principalement :

1. RIP − Protocole de routage à vecteur de distance.
2. IGRP − Protocole de routage à vecteur de distance de Cisco.
3. OSPF − Protocole de routage à état de liens. (Open Shortest Path First)
4. EIGRP − Protocole de routage hybride symétrique.

EGP (Exterior Gateway Protocol), BGP (Border Gateway Protocol) sont des protocoles de routage inter AS.

68.2.2. Choix d'une route et métrique

Pour déterminer une route à utiloiser, un routeur va se baser sur "métrique". Cette valeur est déterminér par un ou plusieurs critères.

Le protocole RIP ne prend en compte que le nombre de sauts (hop) pour déterminer la chemin le plus court. (sa métrique).

D'autres protocoles peuvent prendre en compte d'autres paramètres comme la charge, le nombre de sauts, la bande passante, le délai, la charge...

68.3. Les protocoles de routages IGP's

On considère deux grandes familles de protocoles utilisant des algorithmes.

68.3.1. Les algorithmes Vector−Distance

Les protocoles comme RIP qui utilisent cete algorithme, utilisent le nombre de sauts (hop) comme métrique pour sélectionner un chemin.

Le routage à vecteur de distance détermine la direction (le vecteur) et la distance par rapport à une liaison du réseau.

Un routeur diffuse régulièrement (toutes les 30 secondes) à ses voisins les routes qu'il connaît. Une route est composée d'une adresse destination, d'une adresse de passerelle et d'une métrique indiquant le nombre de sauts nécessaires pour atteindre la destination. Une passerelle qui reçoit ces informations compare les routes reçues avec ses propres routes connues et met à jour sa propre table de routage.

Le problème vient souvent du fait la taille des tables de routage est proportionnelle au nombre de routeurs du domaine et que cela génère très vite une charge importante sur le réseau.

68.3.2. Algorithme Link State (État de Liens)

L'algorithme Link State, est basé sur la technique "Shortest Path First" (SPF). Les routeurs maintiennent une carte complète du réseau et calculent les meilleurs chemins localement en utilisant cette topologie. Ils ne communiquent pas, comme dans l'algorithme "Vector Distance" la liste de toutes les destinations connues. Les routeurs valident l'état des liens qui les relient et communiquent cet état aux routeurs voisins.

Cette technique recrée un état du réseau. Elle utilise en premier le plus court chemin d'abord mais peut utiliser d'autres paramètres. Les routeurs mettent à jour leur carte et recalculent localement pour chaque lien modifié, la nouvelle route selon l'algorithme de Dijkstra shortest path algorithm qui détermine le plus court chemin pour toutes les destinations à partir d'une même source.Elle est utilisée par le protocole OSPF par exemple.

68.3.3. Les techniques hybrides

Ce sont des protocoles qui utilisent les deux techniques de routages, comme le protocole EIGRP de Cisco.

68.4. Les protocoles de routages extérieurs EGP

Ces protocoles sont utilisés échanger les informations entre les systèmes autonomes. Ils permettent l'échange d'information entre paserelles d'AS différents et le test de la disponibilité des liaisons.

Chapitre 69. Remerciements et licence

69.1. Copyright