Dernière mise-à-jour : 2020/01/30 03:28

LSF117 - Gestion du Réseau

Comprendre les Réseaux

Présentation des Réseaux

La définition d'un réseau peut être résumé ainsi :

- un ensemble d'**Equipements** (systèmes et périphériques) communiquant entre eux,
- une entité destinée au transport de données dans différents environnements.

Pour que la communication soit efficace, elle doit respecter les critères suivants :

- présenter des informations compréhensibles par tous les participants,
- être compatible avec un maximum d'interlocuteurs différents (dans le cas d'un réseau, les interlocuteurs sont des équipements : imprimantes, ordinateurs, clients, serveurs, téléphones...),
- si l'interlocuteur n'est pas disponible, les informations ne doivent pas se perdre,
- permettre une réduction des coûts (par ex. interconnexion à bas coût),
- permettre une productivité accrue (par ex. interconnexion à haut débit),
- être sécurisée si les informations à transmettre sont dites sensibles,
- garantir l'**unicité** et de l'**universalité** de l'**accès à l'information**.

On peut distinguer deux familles d'**Equipements** - les **Eléments Passifs** et les **Eléments Actifs**.

Les **Eléments Passifs** transmettent le signal d'un point à un autre :

- **Les Infrastructures ou Supports** - des câbles, de l'atmosphère ou des fibres optiques permettant de relier **physiquement** des équipements,
- **La Topologie** - l'architecture d'un réseau définissant les connexions entre les **Equipements** et, éventuellement, la hiérarchie entre eux.

Les **Eléments Actifs** sont des équipements qui consomment de l'énergie en traitant ou en interprétant le signal. Les **Equipements** sont classés selon leurs fonctions :

- **Equipement de Distribution Interne au Réseau** - Répartiteur (Hub, Switch, Commutateur etc.), Borne d'accès (Hotspot), Convertisseur de signal (Transciever), Amplificateur (Répéteur) ...,
- **Equipement d'Interconnexion de Réseaux** - Routeurs, Ponts ...,
- **Nœuds** et **Interfaces Réseaux** - postes informatiques, équipements en réseau ....

Un **Nœud** est une extrémité de connexion qui peut être une intersection de plusieurs connexions ou de plusieurs **Equipements**.

Une **Interface Réseau** est une prise ou élément d'un **Equipement Actif** faisant la connexion vers d'autres **Equipements** réseaux et qui reçoit et émet des données.

Dans le cas d'un mélange d'**Equipements** non-homogènes en termes de performances au sein du même réseau, c'est la loi du plus faible qui emporte.

Tous les **Equipements** connectés au même support doivent respecter un ensemble de règles appelé une **Protocole de Communication**.

Les **Protocoles de Communication** définissent de façon formelle et interopérable la manière dont les informations sont échangées entre les **Equipements**.

Des **Logiciels**, dédiés à la gestion de ces **Protocoles de Communication**, sont installés sur des **Equipements d'Interconnexion** afin de fournir des fonctions de contrôle permettant une communication entre les **Equipements**.

Se basant sur des **Protocoles de Communication**, des **Services** fournissent des fonctionnalités accessibles aux utilisateurs ou d'autres programmes.

L'ensemble des **Equipements**, **Logiciels** et **Protocoles de Communication** constitue l'**Architecture Réseau**.

Classification des Réseaux

Les réseaux peuvent être classifiés de trois façon différentes :

- par **Mode de Transmission**,
- par **Topologie**,
- par **Étendue**.

Classification par Mode de Transmission

Il existe deux **Classes** de réseaux dans cette classification :

- les **Réseaux en Mode de Diffusion**,
  - utilise un seul support de transmission,
  - le message est envoyé sur tout le réseau à l'adresse d'**un** destinataire,
- les **Réseaux en Mode Point à Point**,
  - une seule liaison entre deux équipements,
  - les nœuds permettent de choisir la route en fonction de l'adresse du destinataire,
  - quand deux nœuds non directement connectés entre eux veulent communiquer ils le font par l'intermédiaire des autres noeuds du réseau.

Classification par Topologie

La **Topologie Physique** d'un réseau décrit l'organisation de ce dernier en termes de câblage. La **Topologie Logique** d'un réseau décrit comment les données circulent sur le réseau. En effet c'est le choix des concentrateurs ainsi que les connections des câbles qui déterminent la topologie logique.

La Topologie Physique

Il existe 6 topologies physiques de réseau :

- La Topologie en Ligne,
- La Topologie en Bus,

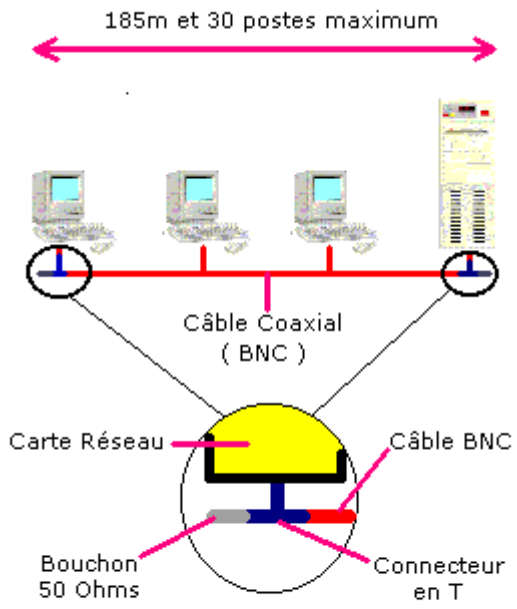
- La Topologie en Etoile,
- La Topologie en Anneau,
- La Topologie en Arbre,
- La Topologie Maillée.

La Topologie en Ligne

Tous les nœuds sont connectés à un seul support. L'inconvénient de cette topologie est que dans le cas d'une défaillance d'une station, le réseau se trouve coupé en deux sous-réseaux.

La Topologie en Bus

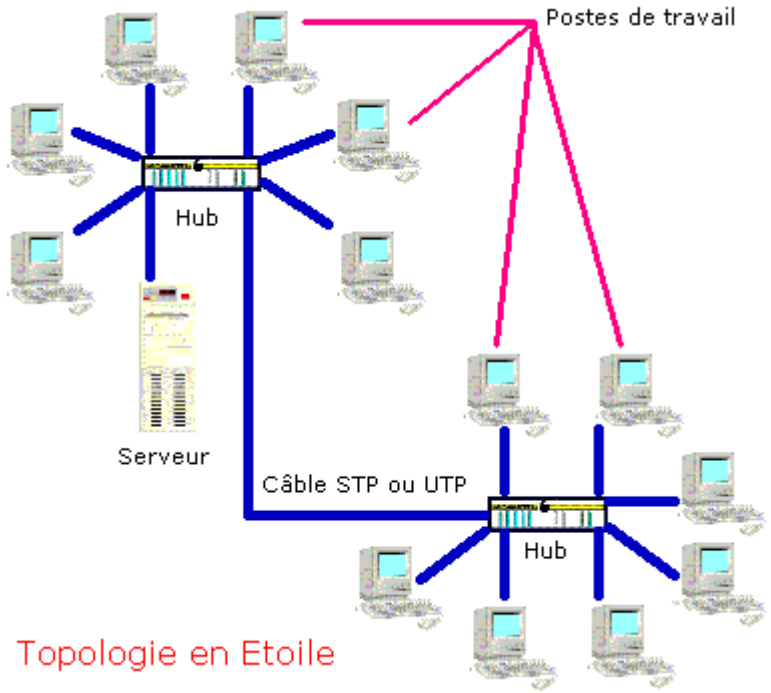
Tous les nœuds sont connectés à un seul support (un câble BNC en T) avec des bouchons à chaque extrémité. La longueur du bus est limitée à **185m**. Le nombre de stations de travail est limité à **30**. Les Stations sont reliées au Bus par des 'T'. Les bouchons sont des terminateurs qui sont des résistances de **50 Ohms**. Quand le support tombe en panne, le réseau ne fonctionne plus. Quand une station tombe en panne, elle ne perturbe pas le fonctionnement de l'ensemble du réseau. Les Stations étant reliés à un suel support, ce type de topologie necessite un **Protocole d'Accès** pour gérer le tour de parole des Stations afin d'éviter des conflits.



Topologie en Bus

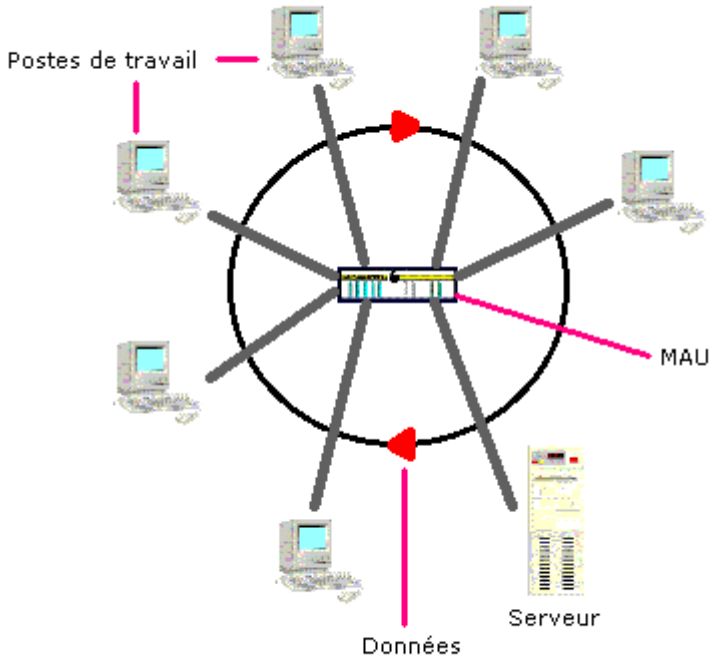
La Topologie en Étoile

Chaque nœud est connecté à un périphérique central appelé un **Hub** (**Concentrateur**) ou un **Switch** (**Commutateur**). Un Hub ou un Switch est prévu pour 4, 8, 16, 32 ... stations. En cas d'un réseau d'un plus grand nombre de stations, plusieurs Hubs ou Switches sont connectés ensemble. Quand une station tombe en panne, elle ne perturbe pas le fonctionnement de l'ensemble du réseau. Le point faible de cette topologie est l'équipement central.



La Topologie en Anneau

Chaque nœud est relié directement à ses deux voisins dans une topologie logique de cercle ininterrompu et une topologie physique en étoile car les stations sont reliées à un type de hub spécial, appelé un **Multistation Access Unit** (MAU).



Topologie en Anneau

Les stations sont reliées à la MAU par un câble 'IBM' munie d'une prise **AUI** du côté de la carte et une prise **Hermaphrodite** du coté de la MAU. Les données sont échangées dans un sens unidirectionnel. Une trame, appelée un **jeton**, circule en permanence. Si l'anneau est brisé, l'ensemble du réseau s'arrête. Pour cette raison, il est courant de voir deux anneaux contre-rotatifs.

La Topologie en Arbre

La Topologie en Arbre est utilisée dans un réseau hierarchique où le sommet, aussi appelé la **racine**, est connecté à plusieurs noeuds de niveau inférieur. Ces neouds peuvent à leur tour être connectés à d'autres noeuds inférieurs. L'ensemble forme une arborescence. Le point faible de cette topologie est sa racine. En cas de défaillance, le réseau est coupé en deux.

La Topologie Maillée

Cette Topologie est utilisée pour des grands réseaux de distribution tels Internet ou le WIFI. Chaque noeud à tous les autres via des liaisons point à point. Le nombre de liaisons devient très rapidement important en cas d'un grand nombre de noeuds. Par exemple dans le cas de 100 Stations (N), le nombre de liaisons est obtenu par la formule suivante :

```
N(N-1)/2 = 100(100-1)/2 = 4 950
```

La **Topologie Physique** la plus répandue est la **Topologie en Etoile**.

Classification par Etendue

La classification par étendue nous fournit 4 réseaux principaux :

Nom	Description	Traduction	Taille Approximative (M)
PAN	Personal Area Network	Réseau Personnel	1 -10
LAN	Local Area Network	Réseau Local Entreprise (RLE)	5 - 1 200
MAN	Métropolitain Area Network	Réseau Urbain	900 - 100 000
WAN	Wide Area Network	Réseau Long Distance (RLD)	50 000 et au delà

Cependant, d'autres classification existent :

CAN	Campus Area Network	Réseau de Campus
GAN	Global Area Network	Réseau Global
TAN	Tiny Area Network	Réseau Minuscule
FAN	Family Area Network	Réseau Familial
SAN	Storage Area Network	Réseau de Stockage

Etant donné que les WANs sont gérés par des opérateurs de télécommunications qui doivent demander une licence à l'état mais que les LANs ont été historiquement mis en oeuvre dans les entreprises, ces derniers sont en majorité issus du monde informatique.

Les Types de LAN

Il existe deux types de LAN :

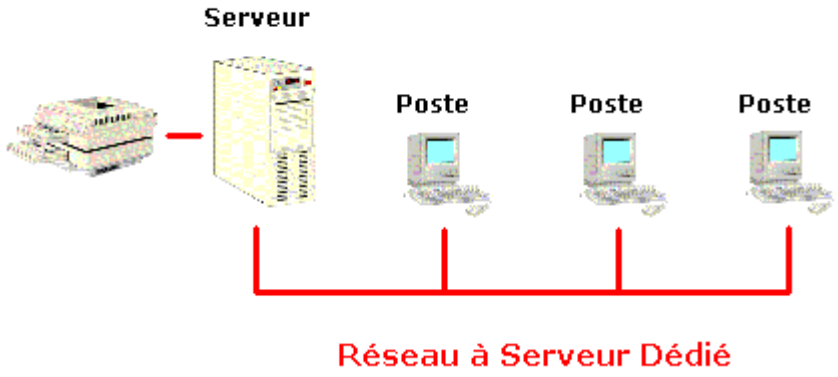
- le réseau à serveur dédié,
- le réseau poste à poste.

Réseau à Serveur Dédié

Le réseau à serveur dédié est caractérisé par le fait que toutes les ressources (imprimantes, applications, lecteurs etc.) sont gérées par le serveur. Les autres micro-ordinateurs ne jouent le rôle de client.

Des exemples des systèmes d'exploitation du réseau à serveur dédié sont :

- Windows NT Server,
- Windows 2000 Server,
- Windows 2003 Server,
- Windows 2008 Server,
- Linux,
- Unix.

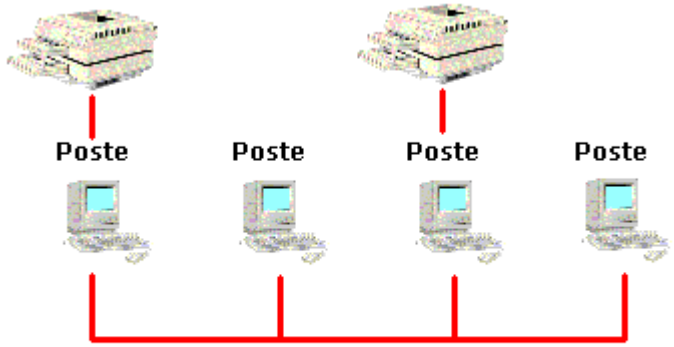


Réseau à Serveur Dédié

Réseau Poste-à-Poste

Le réseau poste à poste est caractérisé par le fait que tous les ordinateurs peuvent jouer le rôle de client et de serveur :

- Windows 95,
- Windows 98,
- Windows NT Workstation.



Réseau Poste à Poste

Le Modèle Client/Serveur

Le modèle Client/Serveur est une des modalités des architectures informatiques distribuées. Dans ce modèle un serveur est tout **Logiciel** fournissant un **Service**.

Le serveur est aussi :

- passif, c'est-à-dire en attente permenante d'une demande, appelée une requête d'un client,
- capable de traiter plusieurs requêtes simultanément en utilisant le **multi-threading**,
- garant de l'intégrité globale.

Le client est, par contre **actif**, étant à l'origine des requêtes.

Il existe trois types de modèle client/serveur :

- **Plat** - tous les clients communiques avec un seul serveur,
- **Hiérarchique** - les clients n'ont de contact qu'avec les serveurs de plus haut niveau qu'eux,
- **Peer-to-Peer** - les équipements sont à la fois client **et** serveur en même temps.

Modèles de Communication

Les réseaux sont bâtis sur des technologies et des modèles. Le modèle **théorique** le plus important est le modèle **O**pen **S**ystem **Interconnection** créé par l'**I**nternational **Organization** for **S**tandardization tandis que le modèle pratique le plus important est le modèle **TCP/IP**.

Le modèle OSI

Le modèle OSI qui a été proposé par l'ISO est devenu le standard en termes de modèle pour décrire l'échange de données entre ordinateurs. Cette norme se repose sur sept couches, de la une - la Couche Physique, à la sept - la Couche d'Application, appelés des services. La communication entre les différentes couches est synchronisée entre le poste émetteur et le poste récepteur grâce à ce que l'on appelle un protocole.

Ce modèle repose sur trois termes :

- Les **Couches**,
- Les **Protocoles**,
- Les **Interfaces**.

Les Couches

Des sept couches :

- Les couches 1 à 3 sont les **Couches Basses** orientées **Transmission**,
- La couche 4 est la **Couche Charnière** entre les **Couches Basses** et les **Couches Hautes**,
- Les couches 5 à 7 sont les **Couches Hautes** orientées **Traitement**.

La couche du même niveau du système **A** parle avec son homologue du système **B**.

- **La Couche Physique** (Couche 1) est responsable :
  - du transfert de données binaires sur le câble physique ou virtuel
  - de la définition de tout aspect physique allant du connecteur jusqu'au câble en passant par la carte réseau, y compris l'organisation même du réseau
  - de la définition des tensions électriques sur le câble pour obtenir le 0 et le 1 binaires
- **La Couche de Liaison** (Couche 2) est responsable :
  - de la réception des données de la couche physique
  - de l'organisation des données en fragments, appelés des trames qui ont un format différent selon s'il s'agit d'un réseau basé sur la technologie Ethernet ou la technologie Token-Ring
  - de la préparation, émission et réception des trames
  - de la gestion de l'accès au réseau
  - de la communication nœud à nœud
  - de la gestion des erreurs
    - avant la transmission, le nœud émetteur calcule un code appelé un CRC et l'incorpore dans les données envoyées
    - le nœud récepteur recalcule un CRC en fonction du contenu de la trame reçue et le compare à celui incorporé avec l'envoi
    - en cas de deux CRC identique, le nœud récepteur envoie un accusé de réception au nœud émetteur
  - de la réception de l'accusé de réception
  - éventuellement de le ré-émission des données
  - En prenant ce modèle, l'IEEE (Institute of Electrical and Eletronics Engineers) l'a étendu avec le Modèle IEEE (802).

        - Dans ce modèle la Couche de Liaison est divisée en deux sous-couches importantes :
            - La **Sous-Couche LLC** (Logical Link Control) qui :
                - gère les accusés de réception
                - gère le flux de trames
            - La **Sous-Couche MAC** (Media Access Control) qui :
                - gère la méthode d'accès au réseau
                - le CSMA/CD dans un réseau basé sur la technologie Ethernet
                - l'accès au jeton dans un réseau basé sur la technologie Token-Ring
                - gère les erreurs

- **La Couche de Réseau** (Couche 3) est responsable de la gestion de la bonne distribution des différentes informations aux bonnes adresses en :
    - identifiant le chemin à emprunter d'un nœud donné à un autre
    - appliquant une conversion des adresses logiques (des noms) en adresses physiques
    - ajoutant des information adressage aux envois
    - détectant des paquets trop volumineux avant l'envoi et en les divisant en trames de données de tailles autorisées

- **La Couche de Transport** (Couche 4) est responsable de veiller à ce que les données soient envoyées correctement en :
    - constituant des paquets de données corrects
    - les envoyant dans le bon ordre
    - vérifiant que les données sont traités dans le même ordre que l'ordre d'émission
    - permettant à un processus sur un nœud de communiquer avec un autre nœud et d'échanger des messages avec lui

- **La Couche de Session** (Couche 5) est responsable :
    - de l'établissement, du maintien, et de la mise à fin de la communication entre deux noeuds distants, c'est-à-dire, de la session
    - de la conversation entre deux processus de vérification de la réception des messages envoyés en séquences, c'est-à-dire, le point de contrôle

- de la sécurité lors de l'ouverture de la session, c'est-à-dire, les droits d'utilisateurs etc.

- **La Couche de Présentation** (Couche 6) est responsable :
    - du formatage et de la mise en forme des données
    - des conversions de données telles le cryptage/décryptage

- **La Couche d'Application** (Couche 7) est responsable :
    - du dialogue homme/machine via des messages affichés
    - du partage des ressources

- de la messagerie

Les Protocoles

Un **protocole** est un langage commun utilisé par dexu entités en communication pour pouvoir se comprendre. La nature du Protocole dépends directement de la nature de la communication. Cette bature dépend du **paradigme** de communication que l'application nécessite. Le paradigme est un modèle abstrait d'un problème ou d'une situation. Dans le paradigme de la diffusion, l'émetteur envoie dans informations au récepteur sans se soucier de ce que le récepteur va en faire. C'est la responsabilité du récepteur de comprendre et d'utiliser les informations.

Les Interfaces

Chaque couche rend des **services** à la couche immédiatement supérieure et utilise les services de la couche immédiatement inférieure. L'ensemble des services s'appelle une **Interface**. Les services sont composés de **S**ervice **D**ata **U**nits et sont disponibles par un **S**service **A**ccess **P**oint.

Protocol Data Units

L'**Unité de Données** ou *Protocol Data Unit* pour chaque couche comporte un nom spécifique :

- **Application Protocol Data Units** pour la couche **Application**,
- **Présentation Protocol Data Units** pour la couche **Présentation**,
- **Session Protocol Data Units** pour la couche **Session**,
- **Transport Protocol Data Units** pour la couche **Transport**.

Or, pour les **Couches Basses** on parle de :

- **Paquets** pour la couche **Réseau**,
- **Trames** pour la couche **Liaison**,
- **Bits** pouyr la couche **Physique**.

Encapsulation et Désencapsulation

Lorque les données sont communiqueés par le système A au système B, celles-ci commencent au niveau de la couche d'Application. Le couche d'Application ajoute une en-tête à l'unité de données qui contient des **informations de contrôle du protocole**. Au passage de chaque couche, celle-ci ajoute sa propre en-tête. De cette façon, lors de sa descente vers la couche physique, les données et l'entête de la couche supérieure sont encapulsulées :

Couche Système A	**Encapsulation**
Application	Application Header (AH) + Unité de Données (UD)
Présentation	Présentation Header (PH) + AH + UD
Session	Session Header (SH) + PH + AH + UD
Transport	Transport Header (TH) + SH + PH + AH + UD
Réseau	Network Header (NH) + TH + SH + PH + AH + UD
Liaison	Liaison Header (DH) + NH + TH + SH + PH + AH + UD

Lors de son voyage de la couche Physique vers la couche Application dans le système B, les en-têtes sont supprimées par chaque couche correspondante. On parle alors de **désencapsulation** :

Couche Système B	**Encapsulation**
Liaison	Liaison Header (DH) + NH + TH + SH + PH + AH + UD
Réseau	Network Header (NH) + TH + SH + PH + AH + UD
Transport	Transport Header (TH) + SH + PH + AH + UD
Session	Session Header (SH) + PH + AH + UD
Présentation	Présentation Header (PH) + AH + UD
Application	Application Header (AH) + Unité de Données (UD)

Spécification NDIS et le Modèle ODI

<note tip> **Cliquez ici pour ouvrir le schéma Simplifié du Modèle OSI incluant la spécification NDIS** </note>

La spécification NDIS (Network Driver Interface Specification) a été introduite conjointement par les sociétés Microsoft et 3Com. Cette spécification ainsi que son homologue, le modèle ODI (Open Datalink Interface) introduit conjointement par les sociétés Novell et Apple à la même époque, définit

des standards pour les pilotes de cartes réseau afin qu'ils puissent être indépendants des protocoles utilisées et les systèmes d'exploitation sur les machines. Des deux 'standards', la spécification NDIS est le plus répandu, intervenant a niveau de la sous-couche MAC et l a couche de liaison. Elle spécifie :

- l'interface pilote-matériel
- l'interface pilote-protocole
- l'interface pilote - système d'exploitation

Le modèle TCP/IP

<note tip> **Cliquez ici pour voir le modèle OSI incluant la suite des protocoles et services TCP/IP** </note>

La suite des protocoles TCP/IP (Transmission Control Protocol / Internet Protocol) est issu de la DOD (Dept. Américain de la Défense) et le travail de l'ARPA (Advanced Research Project Agency).

- La suite des protocoles TCP/IP
  - a été introduite en 1974
  - a été utilisée dans l'ARPAnet en 1975
  - permet la communication entre des réseaux à base de systèmes d'exploitation, architectures et technologies différents
  - est très proche du modèle OSI en termes d'architecture et se place au niveau de la couche d'Application jusqu'à la couche Réseau.
  - est, en réalité, une suite de protocoles et de services :
    - **IP** (Internet Protocol)
      - le protocole IP s'intègre dans la couche Réseau du modèle OSI en assurant la communication entre les systèmes. Bien qu'il puisse découper des messages en fragments ou datagrammes et les reconstituer dans le bon ordre à l'arrivée, il ne garantit pas la réception.
    - **ICMP** (Internet Control Message Protocol)
      - le protocole ICMP produit des messages de contrôle aidant à synchroniser le réseau. Un exemple de ceci est la commande ping.
    - **TCP** (Transmission Control Protocol)
      - le protocole TCP se trouve au niveau de la couche de Transport du modèle OSI et s'occupe de la transmission des données entre noeuds.
    - **UDP** (User Datagram Protocol)
      - le protocole UDP n'est pas orienté connexion. Il est utilisé pour la transmission rapide de messages entre nœuds sans garantir

leur acheminement.

- **Telnet**
  - le protocole Telnet est utilisé pour établir une connexion de terminal à distance. Il se trouve dans la couche d'Application du modèle OSI.
- **Ftp** (File Transfer Protocol)
  - le protocole ftp est utilisé pour le transfert de fichiers. Il se trouve dans la couche d'Application du modèle OSI.
- **SMTP** (Simple Message Transfer Protocol)
  - le service SMTP est utilisé pour le transfert de courrier électronique. Il se trouve dans la couche d'Application du modèle OSI.
- **DNS** (Domain Name Service)
  - le service DNS est utilisé pour le résolution de noms en adresses IP. Il se trouve dans la couche d'Application du modèle OSI.
- **SNMP** (Simple Network Management Protocol)
  - le protocole SNMP est composé d'un agent et un gestionnaire. L'agent SNMP collecte des informations sur les périphériques, les configurations et les performances tandis que le gestionnaire SNMP reçois ses informations et réagit en conséquence.
- **NFS** (Network File System)
  - le NFS a été mis au point par Sun Microsystems
  - le NFS génère un lien virtuel entre les lecteurs et les disques durs permettant de monter dans un disque virtuel local un disque distant
- et aussi POP3, NNTP, IMAP etc ...

<note tip> **Cliquez ici pour voir les modèles TCP/IP et OSI** </note>

Le modèle TCP/IP est composé de 4 couches :

- La couche d'Accès Réseau
  - Cette couche spécifie la forme sous laquelle les données doivent être acheminées, quelque soit le type de réseau utilisé.
- La couche Internet
  - Cette couche est chargée de fournir le paquet de données.
- La couche de Transport
  - Cette couche assure l'acheminement des données et se charge des mécanismes permettant de connaître l'état de la transmission.
- La couche d'Application
  - Cette couche englobe les applications standards de réseau telles ftp, telnet, ssh, etc..

Les noms des Unités de Données sont différents selon le protocole utilisé et la couche du modèle TCP/IP :

Couche	TCP	UDP
Application	Stream	Message
Transport	Segment	Packet
Internet	Datagram	Datagram
Réseau	Frame	Frame

Les Raccordements

Les Modes de Transmission

On peut distinguer 3 modes de transmission :

- La **Liaison Simplex**,
  - Les données ne circulent que dans un **seul** sens de l'émetteur ver le récepteur,
  - La liaison nécessite deux canaux de transmissions,
- La **Liaison Half-Duplex** aussi appelée la **Liaison à l'Alternat** ou encore la **Liaison Semi-Duplex**,
  - Les données circulent dans un sens ou l'autre mais jamais dans les deux sens en même temps. Chaque extrémité émet donc à son tour,
  - La liaison permet d'avoir une liaison bi-directionnelle qui utilise la totalité de la banse passante,
- La **Liaison Full-Duplex** dans les deux sens en **même** temps. Chaque extrémité peut émettre et recevoir simultanément,
  - La liaison est caractérisée par une bande passante divisée par deux pour chaque sens des émissions.

Les Câbles

Le Câble Coaxial

En partant de l'extérieur, le câble coaxial est composé :

- d'une **Gaine** en caoutchouc, PVC ou Téflon pour protéger le câble,
- d'un **Blindage** en métal pour diminuer le bruit du aux parasites,
- d'un **Isolant** (diélectrique) pour éviter le contact entre le blindage et l'âme et ainsi éviter des courts-circuits,
- d'un **Âme** en cuivre ou torsadés pour transporter les données.

Avantages :

- **Peux coûteux**,
- Facilement **manipulable**,
- Peut être utilisé pour de **longues distances**,
- A un débit de 10 Mbit/s dans un LAN et 100 Mbit/s dans un WAN.

Inconvénients :

- Fragile,
- Instable,
- Vulnérable aux interférences,
- Half-Duplex.

Le Câble Paire Torsadée

Ce câble existe sous deux formes selon son utilisation :

- **Monobrin** pour du câblage **horizontal** (**Capillaire**),
  - chaque fil est composé d'un seul conducteur en cuivre,
  - la distance ne doit pas dépassée 90m.
- **Multibrin** pour des **cordons de brassage** :
  - chaque fil est composé de plusieurs brins en cuivre,
  - câble souple.

Avantages :

- Un débit de 10 Mbit/s à 10 GBit/s,
- A une bande passante plus large,
- Pas d'interruption par coupure du câble,
- Permet le **câblage universel** (téléphonie, fax, données ...),
- Full-Duplex.

Inconvénients :

- Nombre de câbles > câble coaxial,
- Plus cher,
- Plus encombrant dans les gaines techniques.

Catagories de Blindage

Il existe trois catagories de blindage :

- **Twisted** ou Torsadé,
- **Foiled** ou Entouré,
- **Shielded** ou Avec Ecran.

De ce fait, il existe 5 catagories de câbles Paire Torsadée :

Nom anglais ^ Appelation Ancienne ^ Nouvelle Appelation ^

Unshielded Twisted Pair	UTP	U/UTP
Foiled Twisted Pair	FTP	F/UTP
Shield Twisted Pair	STP	S/UTP
Shield Foiled Twisted Pair	SFTP	SF/UTP
Shield Shield Twisted Pair	S/STP	SS/STP3

Ces catégories donnent lieu à des **Classes** :

Classe	**Débit**	**Nombre de Paires Torsadées**	**Connecteur**	**Commentaires**
3	10 Mbit/s	4	RJ11	
4	16 Mbit/s	4	S/O	Non-utilisée de nos jours
5	100 Mbit/s	4	RJ45	Obsolète
5e/D	1 Gbit/s sur 100m	4	RJ45	S/O

Classe	Débit	Nombre de Paires Torsadées	Connecteur	Commentaires
6/E	2.5 Gbit/s sur 100m ou 10 Gbit/s sur 25m à 55m	4	Idéal pour PoE	
7/F	10 Gbit/s sur 100m	4	GG45 ou Tera	Paires individuellement et collectivement blindées. Problème de compatibilité avec les classes precédentes due au connecteur.

La Prise RJ45

Une prise RJ45 comporte 8 broches. Un câble peut être **droit** quand la broche 1 d'une extremité est connectée à la broche 1 de la prise RJ45 à l'autre extrémité, la broche 2 d'une extremité est connectée à la broche 2 de la prise RJ45 à l'autre extrémité et ainsi de suite ou bien **croisé** quand le brochage est inversé.

Les câbles croisés sont utilisés lors du branchement de deux équipements identiques (PC à PC, Hub à Hub, Routeur à Routeur).

Channel Link et Basic Link

Le **Channel Link** ou **Canal** est l'ensemble du **Basic Link** ou **Lien** de base et les cordons de brassage et de raccordement des équipements qui sont limités en distance à 10m.

Le **Basic Link** est le lien entre la prise RJ45 murale et la baie de brassage. Il est limité à 90m en classe 5D.

La Fibre Optique

La **Fibre Optique** est un fil de **Silice** permettant le transfert de la lumière. De ce fait elle est caractérisée par :

- des meilleures performances que le cuivre,
- de plus de communications simultanément,
- de la capacité de relier de plus grandes distances,
- une insensibilité aux perturbations,
- une résistance à la corrosion.

Qui plus est, elle ne produit aucune perturbation.

Elle est composée :

- d'un coeur de 10, de 50/125 ou de 62.50 micron,
- d'une gaine de 125 micron,
- d'une protection de 230 micron.

Il existe deux types de fibres, la **Fibre Monomode** et la **Fibre Multimodes**.

La Fibre Monomode :

- a un coeur de 8 à 10 Microns,
- est divisée en sous-catégories de distance,
    - 10 Km,
    - 15 Km,
    - 20 Km,
    - 50 Km,
    - 80 Km,
    - 100 Km.

La Fibre Multimode :

- a un coeur de 62,50 micron ou de 50/125 micron avec une gaine orange,
- permet plusieurs trajets lumineux appelés **modes** en même temps en Full Duplex,
- est utilisée pour de bas débits ou de courtes distances,
    - 2 Km pour 100 Mbit/s,
    - 500 m pour 1 Gbit/s.

Les Réseaux sans Fils

Les réseaux sans fils sans basés sur une liaison qui utilise des ondes radio-électriques (radio et infra-rouges).

Il existe des technologies différentes en fonction de la fréquence utilisée et de la portée des transmissions :

- Réseaux Personnels sans Fils - Bluetooth, HomeRF,
- Réseaux Locaux sans Fils - LiFI, WiFI,
- Réseaux Métropolitains sans Fil - wImax,
- Réseaux Etendus sans Fils - GSM, GPRS, UMTS.

Les principales ondes utilisées pour la transmission des données sont :

- Ondes GSM - Ondes Hertziennes repeosant sur des micro-ondes à basse fréquence avec une portée d'une dizaine de kilomètres,
- Ondes Wi-Fi - Ondes Hertziennes reposant sur des micro-ondes à haute fréquence avec une portée de 20 à 50 mètres,
- Ondes Satellitaires - Ondes Hertziennes longues portées.

Le Courant Porteur en Ligne

Le CPL utilise le réseau électrique domestique, le réseau moyenne et basse tension pour transmettre des informations numériques.

Le CPL superpose un signal à plus haute fréquence au signal électrique.

Seuls donc, les fils conducteurs transportent les signaux CPL.

Le coupleur intégré en entrée des boîtiers CPL élimine les composants basses fréquences pour isoler le signal CPL.

Le CPL utilise la phase électrique et le neutre. De ce fait, une installation triphasée fournit 3 réseaux CPL différents.

Le signal CPL ne s'arrête pas necéssairement aux limites de l'installation électrique. En effet en cas de compteurs non-numériques le signal les traversent.

Les normes CPL sont :

Norme	Débit Théorique	Débit Pratique	Temps pour copier 1 Go
Homeplug 1.01	14 Mbps	5.4 Mbps	25m 20s
Homeplug 1.1	85 Mbps	12 Mbps	11m 20s
PréUPA 200	200 Mbps	30 Mbps	4m 30s

Technologies

Il existe plusieurs technologies de réseau :

- Ethernet,
- Token-Ring,
- ARCnet,
- etc..

Nous détaillerons ici les deux technologies les plus répandues, à savoir Ethernet et Token-Ring.

Ethernet

La technologie Ethernet se repose sur :

- une topologie logique de bus,
- une topologie physique de bus ou étoile.

L'accès au bus utilise le **CSMA/CD**, Carrier Sense Multiple Access / Collision Detection (Accès Multiple à Détection de Porteuse / Détection de Collisions).

Il faut noter que :

- les données sont transmises à chaque nœud - c'est la méthode d'**accès multiple**,
- chaque nœud qui veut émettre écoute le réseau - c'est la **détection de porteuse**,
- quand le réseau est silencieux une trame est émise dans laquelle se trouvent les données ainsi que l'adresse du destinataire,
- le système est dit donc **aléatoire** ou **non-déterministe**,
- quand deux nœuds émettent en même temps, il y a **collision de données**,
- les deux nœuds vont donc cesser d'émettre, se mettant en attente jusqu'à ce qu'ils commencent à émettre de nouveau.

Token-Ring

La technologie Token-Ring se repose sur :

- une topologie logique en anneau,
- une topologie physique en étoile.

Token-Ring se traduit par **Anneau à Jeton**. Il n'est pas aussi répandu que l'Ethernet pour des raisons de coûts. En effet le rajout d'un nœud en Token-Ring peut coûter jusqu'à **4 fois plus cher qu'en Ethernet**.

Il faut noter que :

- les données sont transmises dans le réseau par un système appelé **méthode de passage de jeton**,
- le jeton est une **trame numérique vide** de données qui tourne en permanence dans l'anneau,
- quand un nœud souhaite émettre, il saisit le jeton, y dépose des données avec l'adresse du destinataire et ensuite laisse poursuivre son chemin jusqu'à sa destination,
- pendant son voyage, aucun autre nœud ne peut émettre,
- une fois arrivé à sa destination, le jeton dépose ses données et retourne à l'émetteur pour confirmer la livraison,
- ce système est appelé **déterministe**.

L'intérêt de la technologie Token-Ring se trouve dans le fait :

- qu'il **évite des collisions**,
- qu'il est **possible de déterminer avec exactitude le temps que prenne l'acheminement des données**.

La technologie Token-Ring est donc idéale, voire obligatoire, dans des installations où chaque nœud doit disposer d'une opportunité à intervalle fixe d'émettre des données.

Périphériques Réseaux Spéciaux

En plus du câblage, les périphériques de réseau spéciaux sont des éléments primordiaux tant au niveau de la topologie physique que la topologie logique.

Les périphériques de réseau spéciaux sont :

- les Concentrateurs ou *Hubs*,

- les Répéteurs ou *Repeaters*,
- les Ponts ou *Bridges*,
- les Commutateurs ou *Switches*,
- les Routeurs ou *Routers*,
- les Passerelles ou *Gateways*.

L'objectif ici est de vous permettre de comprendre le rôle de chaque périphérique.

Les Concentrateurs

Les Concentrateurs permettent une connectivité entre les nœuds en topologie en étoile. Selon leur configuration, la topologie logique peut être en étoile, en bus ou en anneau. Il existe de multiples types de Concentrateurs allant du plus simple au Concentrateur intelligent.

- **Le Concentrateur Simple**
  - est une boîte de raccordement centrale,
  - joue le rôle de récepteur et du réémetteur des signaux sans accélération ni gestion de ceux-ci,
  - est un périphérique utilisé pour des groupes de travail.
- **Le Concentrateur Évolué**
  - est un Concentrateur simple qui offre en plus l'amplification des signaux, la gestion du type de topologie logique grâce à des capacités d'être configurés à l'aide d'un logiciel ainsi que l'homogénéisation du réseau en offrant des ports pour un câblage différent. Par exemple, 8 ports en paire torsadée non-blindée et un port BNC.
- **Le Concentrateur Intelligent**
  - est un Concentrateur évolué qui offre en plus la détection automatique des pannes, la connectique avec un Pont ou un Routeur ainsi que le diagnostic et la génération de rapports.

Les Répéteurs

Un Répéteur est un périphérique réseau simple. Il est utilisé pour amplifier le signal quand :

- la longueur du câble dépasse la limite autorisée,
- le câble passe par une zone ou les interférences sont importantes.

Éventuellement, et uniquement dans le cas où le Répéteur serait muni d'une telle fonction, celui-ci peut être utiliser pour connecter deux réseaux ayant un câblage différent.

Les Ponts

Un Pont est **Répéteur intelligent**. Outre sa capacité d'amplifier les signaux, le Pont analyse le trafic qui passe par lui et met à jour une liste d'adresses des cartes réseau, appelée **une table de routage**, n'autorisant que les transmissions destinées à d'autres segments du réseau.

Les **diffusions** sont néanmoins autorisées.

Comme un Pont doit être intelligent, on utilise souvent un micro-ordinateur comme Pont. Forcément équipé de 2 cartes réseau, le Pont peut également jouer le rôle de serveur de fichiers.

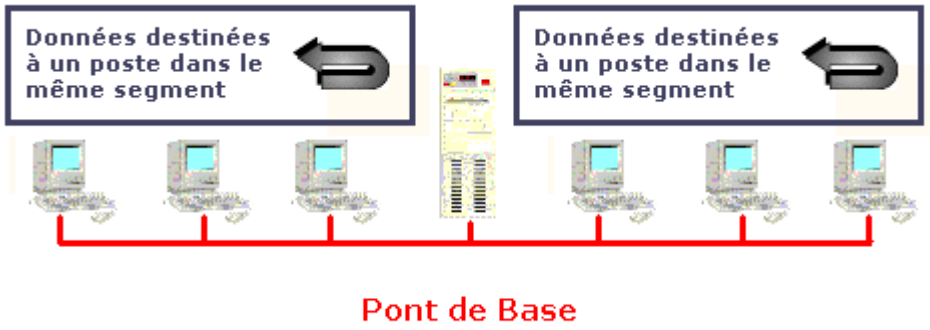
Le Pont sert donc à isoler des segments du réseau pour des raisons de :

- **sécurité** afin d'éviter à ce que des données sensibles soient propagées sur tout le réseau,
- **performance** afin qu'une partie du réseau trop chargée ralentisse le réseau entier,
- **fiabilité** afin par exemple qu'une carte en panne ne gène pas le reste du réseau avec une diffusion.

Il existe trois types de configuration de Ponts

Le Pont de Base

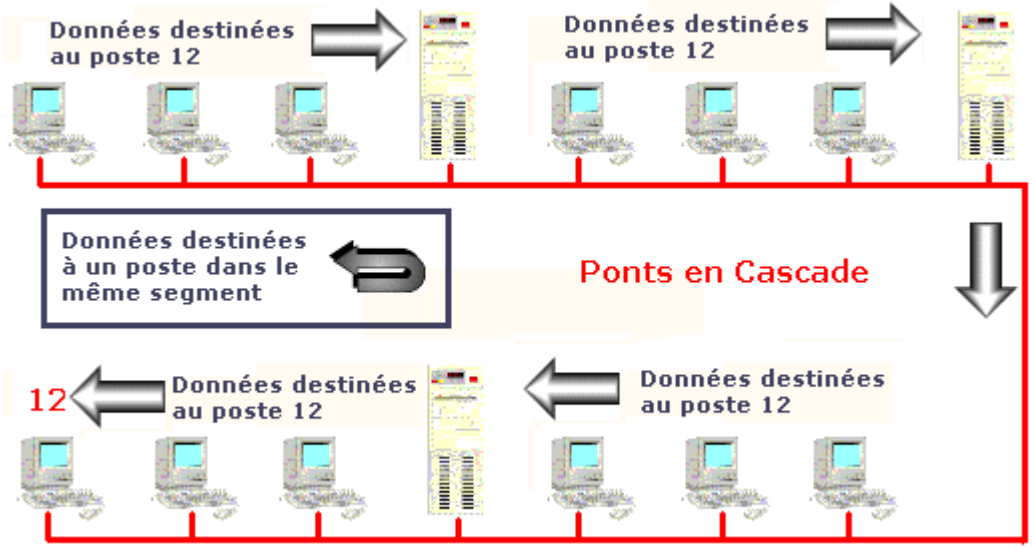
Le Pont de Base est utilisé très rarement pour isoler deux segments.



;#; ;#;

Le Pont en Cascade

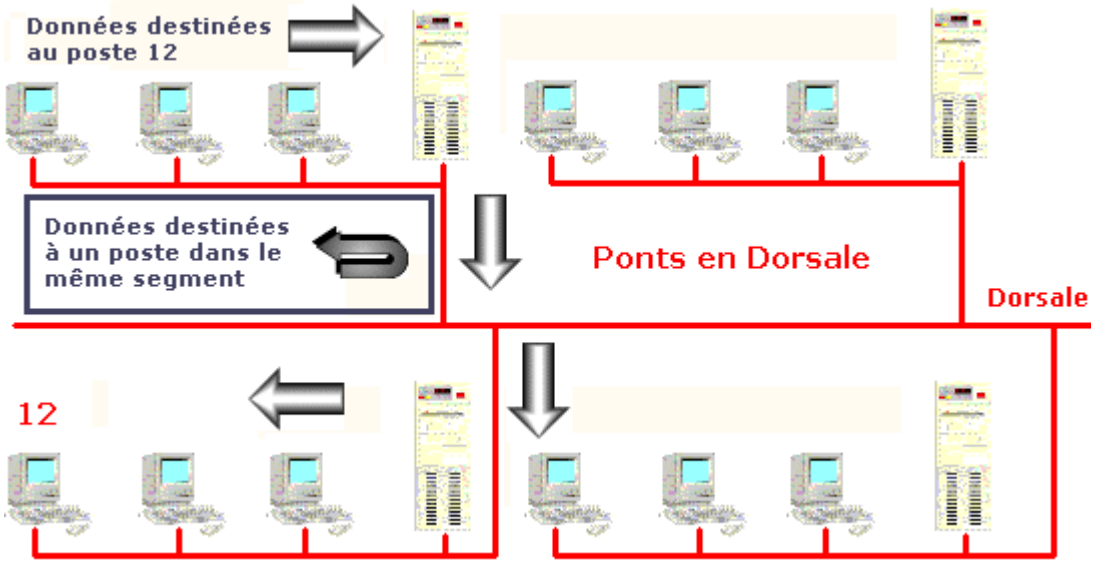
Le Pont en Cascade est à éviter car les données en provenance d'un segment doivent passer par plusieurs Ponts. Ceci a pour conséquence de ralentir la transmission des données, voire même de créer un trafic superflu en cas de rémission par le nœud



;#; ;#;

Le Pont en Dorsale

Le Pont en Dorsale coûte plus chère que la configuration précédente car il faut un nombre de Ponts équivalent au nombre de segments + 1. Par contre elle réduit les problèmes précédemment cités puisque les données ne transitent que par deux Ponts.



;#; ;#;

Les Commutateurs

Un Commutateur peut être considéré comme un Concentrateur intelligent et un Pont. Ils sont gérés souvent par des logiciels. La topologie physique d'un réseau commuté est en étoile. Par contre la topologie logique est spéciale, elle s'appelle une topologie commutée.

Lors de la communication de données entre deux nœuds, le Commutateur ouvre une connexion temporaire virtuelle en fermant les autres ports. De cette façon la bande passante totale est disponible pour cette transmission et les risques de collision sont minimisés.

Certains Commutateurs haut de gamme sont équipés d'un système anti-catastrophe qui leur permet d'isoler une partie d'un réseau en panne afin que les autres parties puissent continuer à fonctionner sans problème.

Les Routeurs

Un Routeur est un Pont sophistiqué capable :

- d'assurer l'interconnexion entre des segments,
- de filtrer le trafic,
- d’isoler une partie du réseau,
- d’ explorer les informations d'adressage pour trouver le chemin le plus approprié et le plus rentable pour la transmission des données.

Les Routeurs utilisent une table de routage pour stocker les informations sur :

- les adresses du réseau,
- les solutions de connexion vers d'autres réseaux,
- l'efficacité des différentes routes.

Il existe deux types de Routeur :

- le **Routeur Statique**
  - la table de routage est éditer manuellement,
  - les routes empruntées pour la transmission des données sont toujours les mêmes,
  - il n'y a pas de recherche d'efficacité.
- le **Routeur Dynamique**
  - découvre automatiquement les routes à emprunter dans un réseau.

Les Passerelles

Ce périphérique, souvent un logiciel, sert à faire une conversion de données :

- entre deux technologies différentes (Ethernet - Token-Ring),
- entre deux protocoles différents,
- entre des formats de données différents.

Comprendre TCP Version 4

En-tête TCP

L'en-tête TCP est codée sur 4 octets soit 32 bits :

<table>
<tr><th>1er octet</th><th>2ème octet</th><th>3ème octet</th><th>4 ème octet</th></tr>
<tr><td colspan="2">Port source</td><td colspan="2">Port destination</td></tr>
<tr><td colspan="4">Numéro de séquence</td></tr>
<tr><td colspan="4">Numéro d'acquittement</td></tr>
<tr><td>Offset</td><td>Flags</td><td colspan="2">Fenêtre</td></tr>
<tr><td colspan="2">Checksum</td><td colspan="2">Pointeur Urgent</td></tr>
<tr><td colspan="3">Options</td><td>Padding</td></tr>
<tr><td colspan="4">Données</td></tr>
</table>

Vous noterez que les numéros de ports sont codés sur 16 bits. Cette information nous permet de calculer le nombres de ports maximum en IPv4, soit 2^{16} ports ou 65 535.

L'**Offset** contient la taille de l'en-tête.

Les **Flags** sont :

- URG - Si la valeur est 1 le pointeur urgent est utilisé. Le numéro de séquence et le pointeur urgent indique un octet spécifique.
- ACK - Si la valeur est 1, le paquet est un accusé de réception
- PSH - Si la valeur est 1, les données sont immédiatement présentées à l'application
- RST - Si la valeur est 1, la communication comporte un problème et la connexion est réinitialisée
- SYN - Si la valeur est 1, le paquet est un paquet de synchronisation
- FIN - Si la valeur est 1, le paquet indique la fin de la connexion

La **Fenêtre** est codée sur 16 bits. La Fenêtre est une donnée liée au fonctionnement d'expédition de données appelé le **sliding window** ou la **fenêtre glissante**. Puisque il serait impossible, pour des raisons de performance, d'attendre l'accusé de réception de chaque paquet envoyé, l'expéditeur envoie des paquets par groupe. La taille de cette groupe s'appelle la Fenêtre. Dans le cas d'un problème de réception d'une partie de la Fenêtre, toute

la Fenêtre est ré-expédiée.

Le **Checksum** est une façon de calculer si le paquet est complet.

Le **Padding** est un champ pouvant être rempli de valeurs nulles de façon à ce que la taille de l'en-tête soit un multiple de 32

En-tête UDP

L'en-tête UDP est codée sur 4 octets soit 32 bits :

1er octet	**2ème octet**	**3ème octet**	**4 ème octet**
Port source		Port destination	
longueur		Checksum	
Données			

L'en-tête UDP a une longueur de 8 octets.

Fragmentation et Ré-encapsulation

La taille limite d'un paquet TCP, l'en-tête comprise, ne peut pas dépasser **65 535 octets**. Cependant chaque réseau est qualifié par son MTU (Maximum Tranfer Unit). Cette valeur est la taille maximum d'un paquet autorisée. L'unité est en **octets**. Pour un réseau Ethernet sa valeur est de 1 500. Quand un paquet doit être expédié sur un réseau ayant un MTU inférieur à sa propre taille, le paquet doit être **fractionné**. A la sortie du réseau, le paquet est reconstitué. Cette reconstitution s'appelle **ré-encapsulation**.

Adressage

L'adressage IP requière que chaque périphérique sur le réseau possède une adresse IP unique de 4 octets, soit 32 bits au format XXX.XXX.XXX.XXX De cette façon le nombre total d'adresses est de 2^{32} = 4.3 Milliards.

Les adresses IP sont divisées en 5 classes, de A à E. Les 4 octets des classes A à C sont divisés en deux, une partie qui s'appelle le **Net ID** qui identifie

le réseau et une partie qui s'appelle le **Host ID** qui identifie le hôte :

	1er octet	2ème octet	3ème octet	4 ème octet
A	Net ID	Host ID		
B	Net ID		Host ID	
C	Net ID			Host ID
D	Multicast			
E	Réservé			

L'attribution d'une classe dépend du nombre de hôtes à connecter. Chaque classe est identifié par un **Class ID** composé de 1 à 3 bits :

Classe	Bits ID Classe	Valeur ID Classe	Bits ID Réseau	Nb. de Réseaux	Bits ID hôtes	Nb. d'adresses	Octet de Départ
A	1	0	7	2^7=128	24	2^{24}=16 777 216	1 - 126
B	2	10	14	2^{14}=16 834	16	2^{16}=65 535	128 - 191
C	3	110	21	2^{21}=2 097 152	8	2^8=256	192 - 223

Dans chaque classe, certaines adresses sont réservées pour un usage privé :

Classe	IP de Départ	IP de Fin
A	10.0.0.0	10.255.255.255
B	172.16.0.0	172.31.255.255
C	192.168.0.0	192.168.255.255

Il existe des adresses particulières ne pouvant pas être utilisées pour identifier un hôte :

Adresse Particulière	Description
169.254.0.0 à 169.254.255.255	Automatic Private IP Addressing de Microsoft
Hôte du réseau courant	Tous les bits du Net ID sont à 0
Adresse de réseau	Tous les bits du Host ID sont à 0
Adresse de diffusion	Tous les bits du Host ID sont à 1

L'adresse de réseau identifie le **segment** du réseau entier tandis que l'adresse de diffusion identifie tous les hôtes sur le segment de réseau.

Afin de mieux comprendre l'adresse de réseau et l'adresse de diffusion, prenons le cas de l'adresse 192.168.10.1 en classe C :

	1er octet	**2ème octet**	**3ème octet**	**4 ème octet**
	Net ID			**Host ID**
Adresse IP	192	168	10	1
Binaire	11000000	10101000	000001010	00000001
Calcul de l'adresse de réseau				
Binaire	11000000	10101000	000001010	**00000000**
Adresse réseau	192	168	10	0
Calcul de l'adresse de diffusion				
Binaire	11000000	10101000	000001010	**11111111**
Adresse de diffusion	192	168	10	255

Masques de sous-réseaux

Tout comme l'adresse IP, le masque de sous-réseau compte 4 octets ou 32 bits. Les masques de sous-réseaux permettent d'identifer le Net ID et le Host ID :

Classe	**Masque**	**Notation CIDR**
A	255.0.0.0	/8
B	255.255.0.0	/16
C	255.255.255.0	/24

Le terme **CIDR** veut dire **Classless InterDomain Routing**. Le terme Notation CIDR correspond au nombre de bits d'une valeur de 1 dans le masque de sous-réseau.

Quand un hôte souhaite émettre il procède d'abord à l'identification de sa propre adresse réseau par un calcul AND (ET) appliqué à sa propre adresse et son masque de sous-réseau qui stipule :

- 1 x 1 = 1
- 0 x 1 = 0
- 1 x 0 = 0
- 0 x 0 = 0

Prenons le cas de l'adresse IP 192.168.10.1 ayant un masque de 255.255.255.0 :

	1er octet	**2ème octet**	**3ème octet**	**4 ème octet**
Adresse IP	192	168	10	1
Binaire	11000000	10101000	00001010	00000001
Masque de sous-réseau				
Binaire	11111111	11111111	11111111	00000000
Calcul AND	11000000	10101000	00001010	00000000
Adresse réseau	192	168	10	0

Cet hôte essaie de communiquer avec un hôte ayant une adresse IP de 192.168.10.10. Il procède donc au même calcul en appliquant **son propre masque de sous-réseau** à l'adresse IP de l'hôte destinataire :

	1er octet	**2ème octet**	**3ème octet**	**4 ème octet**
Adresse IP	192	168	10	10
Binaire	11000000	10101000	00001010	00001010
Masque de sous-réseau				
Binaire	11111111	11111111	11111111	00000000
Calcul AND	11000000	10101000	00001010	00000000
Adresse réseau	192	168	10	0

Puisque l'adresse réseau est identique dans les deux cas, l'hôte émetteur présume que l'hôte de destination se trouve sur son réseau et envoie les paquets directement sur le réseau sans s'adresser à sa passerelle par défaut.

L'hôte émetteur essaie maintenant de communiquer avec avec un hôte ayant une adresse IP de 192.168.2.1. Il procède donc au même calcul en appliquant **son propre masque de sous-réseau** à l'adresse IP de l'hôte destinataire :

	1er octet	**2ème octet**	**3ème octet**	**4 ème octet**
Adresse IP	192	168	2	1
Binaire	11000000	10101000	00000010	00000001
Masque de sous-réseau				
Binaire	11111111	11111111	11111111	00000000
Calcul AND	11000000	10101000	00000010	00000000

	1er octet	2ème octet	3ème octet	4 ème octet
Adresse réseau	192	168	2	0

Dans ce cas, l'hôte émetteur constate que le réseau de destination 192.168.2.0 n'est pas identique à son propre réseau 192.168.10.0. Il adresse donc les paquets à la passerelle par défaut.

VLSM

Puisque le stock de réseaux disponibles sous IPv4 est presque épuisé, une solution a du être trouvée pour créer des sous-réseaux en attendant l'introduction de l'IPv6. Cette solution s'appelle le VLSM ou Variable Length Subnet Masks. Le VLSM exprime les masques de sous-réseaux au format CIDR.

Son principe est simple. Afin de créer des réseaux différents à partir d'une adresse réseau d'une classe donnée, il convient de réduire le nombre d'hôtes. De cette façon les bits 'libérés' du Host ID peuvent être utilisés pour identifier les sous-réseaux.

Pour illustrer ceci, prenons l'exemple d'un réseau 192.168.1.0. Sur ce réseau, nous pouvons mettre 2^8-2 soit 254 hôtes entre 192.168.1.1 au 192.168.1.254.

Supposons que nous souhaiterions diviser notre réseau en 2 sous-réseaux. Pour coder 2 sous-réseaux, il faut que l'on libère 2 bits du Host ID. Les deux bits libérés auront les valeurs binaires suivantes :

- 00
- 01
- 10
- 11

Les valeurs binaires du quatrième octet de nos adresses de sous-réseaux seront donc :

- 192.168.1.00XXXXXX
- 192.168.1.01XXXXXX
- 192.168.1.10XXXXXX
- 192.168.1.11XXXXXX

où les XXXXXX représentent les bits que nous réservons pour décrire les hôtes dans chacun des sous-réseaux.

Nous ne pouvons pas utiliser les deux sous-réseaux suivants :

- 192.168.1.00XXXXXX
- 192.168.1.11XXXXXX

car ceux-ci correspondent aux débuts de l'adresse réseau 192.168.1.0 et de l'adresse de diffusion 192.168.1.255.

Nous pouvons utiliser les deux sous-réseaux suivants :

- 192.168.1.01XXXXXX
- 192.168.1.10XXXXXX

Pour le premier sous-réseau l'adresse réseau et l'adresse de diffusion sont :

Sous-réseau #1	192	168	1	01XXXXXX
Calcul de l'adresse de réseau				
Binaire	11000000	10101000	00000001	01**000000**
Adresse réseau	192	168	1	**64**
Calcul de l'adresse de diffusion				
Binaire	11000000	10101000	00000001	01**111111**
Adresse de diffusion	192	168	1	**127**

- L'adresse CIDR du réseau est donc 192.168.1.64/26 car le Net ID est codé sur 24+2 bits.
- Le masque de sous-réseau est donc le 11111111.11111111.11111111.11000000 ou le 255.255.255.192
- Nous pouvons avoir 2^6-2 soit 62 hôtes.
- La plage valide d'adresses IP est de 192.168.1.65 à 192.168.1.126

Pour le deuxième sous-réseau l'adresse réseau et l'adresse de diffusion sont :

Sous-réseau #2	192	168	1	10XXXXXX
Calcul de l'adresse de réseau				
Binaire	11000000	10101000	00000001	10**000000**
Adresse réseau	192	168	1	**128**

Calcul de l'adresse de diffusion				
Binaire	11000000	10101000	00000001	10**111111**
Adresse de diffusion	192	168	1	**191**

- L'adresse CIDR du réseau est donc 192.168.1.128/26 car le Net ID est codé sur 24+2 bits.
- Le masque de sous-réseau est donc le 11111111.11111111.11111111.11000000 ou le 255.255.255.192
- Nous pouvons avoir 2^6-2 soit 62 hôtes.
- La plage valide d'adresses IP est de 192.168.1.129 à 192.168.1.190

La valeur qui sépare les sous-réseaux est 64. Cette valeur comporte le nom **incrément**.

Ports et sockets

Afin que les données arrivent aux applications que les attendent, TCP utilise des numéros de ports sur la couche transport. Le numéros de ports sont divisés en trois groupes :

- **Well Known Ports**
  - De 1 à 1023
- **Registered Ports**
  - De 1024 à 49151
- **Dynamic** et/ou **Private Ports**
  - De 49152 à 65535

Le couple **numéro IP:numéro de port** s'appelle un **socket**.

/etc/services

Les ports les plus utilisés sont détaillés dans le fichier **/etc/services** :

```
SLES12SP1:~ # tail /etc/services
```

```
# The mrt entry is temporary until a official number is registerd
mrt                50000/tcp    # telnet interface of mrt programs
mrt                50000/udp    # telnet interface of mrt programs
#
# make apple talk more friendly
#
rtmp               1/ddp        # Routing Table Maintenance Protocol
nbp                2/ddp        # Name Binding Protocol
echo               4/ddp        # AppleTalk Echo Protocol
zip                6/ddp        # Zone Information Protocol
```

Notez que les ports sont listés par deux :

- le port TCP
- le port UDP

La liste la plus complète peut être consultée sur le site Internet **www.iana.org**.

Options de la commande

Les options de cette commande sont :

```
SLES12SP1:~ # arp --help
Usage:
  arp [-vn]  [<HW>] [-i <if>] [-a] [<hostname>]             <-Display ARP cache
  arp [-v]          [-i <if>] -d  <hostname> [pub][nopub]    <-Delete ARP entry
  arp [-vnD] [<HW>] [-i <if>] -f  [<filename>]              <-Add entry from file
  arp [-v]   [<HW>] [-i <if>] -s  <hostname> <hwaddr> [temp][nopub] <-Add entry
  arp [-v]   [<HW>] [-i <if>] -s  <hostname> <hwaddr> [netmask <nm>] pub  <-''-
  arp [-v]   [<HW>] [-i <if>] -Ds <hostname> <if> [netmask <nm>] pub      <-''-

        -a                      display (all) hosts in alternative (BSD) style
        -s, --set               set a new ARP entry
```

```
        -d, --delete            delete a specified entry
        -v, --verbose           be verbose
        -n, --numeric           don't resolve names
        -i, --device            specify network interface (e.g. eth0)
        -D, --use-device        read <hwaddr> from given device
        -A, -p, --protocol      specify protocol family
        -f, --file              read new entries from file or from /etc/ethers

  <HW>=Use '-H <hw>' to specify hardware address type. Default: ether
  List of possible hardware types (which support ARP):
    ether (Ethernet) tr (16/4 Mbps Token Ring) tr (16/4 Mbps Token Ring (New))
    ax25 (AMPR AX.25) netrom (AMPR NET/ROM) arcnet (ARCnet)
    dlci (Frame Relay DLCI) fddi (Fiber Distributed Data Interface) hippi (HIPPI)
    irda (IrLAP) x25 (generic X.25) infiniband (InfiniBand)
```

Notez que les ports sont listés par deux :

- le port TCP
- le port UDP

La liste la plus complète peut être consultée à l'adresse suivante **https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml**.

Pour connaitre la liste des sockets ouverts sur l'ordinateur, saisissez la commande suivante :

```
SLES12SP1:~ # netstat -an | more
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:23              0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:25            0.0.0.0:*               LISTEN
tcp        0      0 10.0.2.15:22            10.0.2.2:42320          ESTABLISHED
tcp        0      0 :::22                   :::*                    LISTEN
tcp        0      0 ::1:25                  :::*                    LISTEN
```

```
udp        0      0 0.0.0.0:68              0.0.0.0:*
Active UNIX domain sockets (servers and established)
Proto RefCnt Flags       Type       State         I-Node Path
unix  2      [ ]         DGRAM                    15886  @/org/freedesktop/syste
md1/notify/c692b8aa8fa3c4bb
unix  2      [ ACC ]     STREAM     LISTENING     15887  /run/user/1000/systemd/
private
unix  2      [ ACC ]     STREAM     LISTENING     5392   /run/systemd/journal/st
dout
unix  6      [ ]         DGRAM                    5395   /run/systemd/journal/so
cket
unix  16     [ ]         DGRAM                    5397   /dev/log
unix  2      [ ACC ]     STREAM     LISTENING     8508   /run/lvm/lvmetad.socket
unix  2      [ ACC ]     STREAM     LISTENING     8260   /run/systemd/private
unix  2      [ ]         DGRAM                    8284   /run/systemd/shutdownd
--More--
```

Pour connaitre la liste des applications ayant ouvert un port sur l'ordinateur, saisissez la commande suivante :

```
SLES12SP1:~ # netstat -anp | more
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      1416/sshd
tcp        0      0 0.0.0.0:23              0.0.0.0:*               LISTEN      1399/xinetd
tcp        0      0 127.0.0.1:25            0.0.0.0:*               LISTEN      1808/master
tcp        0      0 10.0.2.15:22            10.0.2.2:42320          ESTABLISHED 1895/sshd: trainee
tcp        0      0 :::22                   :::*                    LISTEN      1416/sshd
tcp        0      0 ::1:25                  :::*                    LISTEN      1808/master
udp        0      0 0.0.0.0:68              0.0.0.0:*                           937/wickedd-dhcp4
Active UNIX domain sockets (servers and established)
Proto RefCnt Flags       Type       State         I-Node PID/Program name    Path
unix  2      [ ]         DGRAM                    15886  1898/systemd
@/org/freedesktop/systemd1/notify/c692b8aa8fa3c4bb
unix  2      [ ACC ]     STREAM     LISTENING     15887  1898/systemd        /run/user/1000/systemd/private
```

```
unix  2      [ ACC ]     STREAM     LISTENING     5392   1/systemd           /run/systemd/journal/stdout
unix  6      [ ]         DGRAM                    5395   1/systemd           /run/systemd/journal/socket
unix  16     [ ]         DGRAM                    5397   1/systemd           /dev/log
unix  2      [ ACC ]     STREAM     LISTENING     8508   1/systemd           /run/lvm/lvmetad.socket
unix  2      [ ACC ]     STREAM     LISTENING     8260   1/systemd           /run/systemd/private
unix  2      [ ]         DGRAM                    8284   1/systemd           /run/systemd/shutdownd
unix  2      [ ACC ]     SEQPACKET  LISTENING     8288   1/systemd           /run/udev/control
unix  2      [ ]         DGRAM                    5381   1/systemd           @/org/freedesktop/systemd1/notify
unix  2      [ ACC ]     STREAM     LISTENING     15230  1808/master         public/cleanup
unix  2      [ ACC ]     STREAM     LISTENING     15235  1808/master         private/rewrite
unix  2      [ ACC ]     STREAM     LISTENING     15238  1808/master         private/bounce
unix  2      [ ACC ]     STREAM     LISTENING     15241  1808/master         private/defer
unix  2      [ ACC ]     STREAM     LISTENING     15244  1808/master         private/trace
unix  2      [ ACC ]     STREAM     LISTENING     15247  1808/master         private/verify
unix  2      [ ACC ]     STREAM     LISTENING     15250  1808/master         public/flush
unix  2      [ ACC ]     STREAM     LISTENING     15253  1808/master         private/proxymap
unix  2      [ ACC ]     STREAM     LISTENING     15256  1808/master         private/proxywrite
unix  2      [ ACC ]     STREAM     LISTENING     15259  1808/master         private/smtp
unix  2      [ ACC ]     STREAM     LISTENING     15262  1808/master         private/relay
unix  2      [ ACC ]     STREAM     LISTENING     15265  1808/master         public/showq
unix  2      [ ACC ]     STREAM     LISTENING     15268  1808/master         private/error
unix  2      [ ACC ]     STREAM     LISTENING     15271  1808/master         private/retry
--More--
```

Résolution d'adresses Ethernet

Chaque protocole peut être encapsulé dans une **trame** Ethernet. Lorsque la trame doit être transportée de l'expéditeur au destinataire, ce premier doit connaitre l'adresse Ethernet du dernier. L'adresse Ethernet est aussi appelée l'adresse **Physique** ou l'adresse **MAC**.

Pour connaître l'adresse Ethernet du destinataire, l'expéditeur fait appel au protocol **ARP**. Les informations reçues sont stockées dans une table. Pour visualiser ces informations, il convient d'utiliser la commande suivante :

```
SLES12SP1:~ # arp -a
```

```
? (10.0.2.2) at 52:54:00:12:35:02 [ether] on eth0
```

Options de la commande

Les options de cette commande sont :

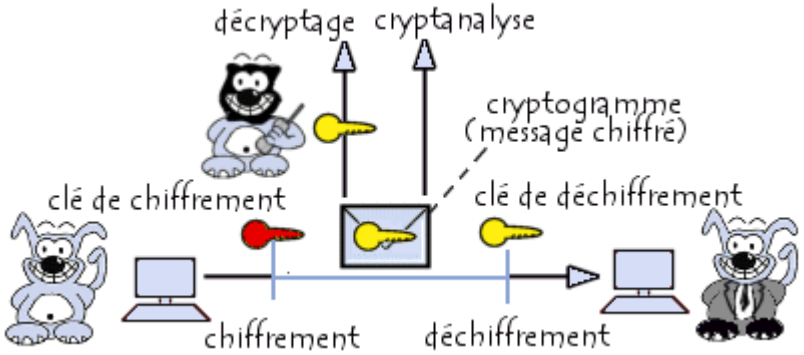
```
SLES12SP1:~ # arp --help
Usage:
  arp [-vn]  [<HW>] [-i <if>] [-a] [<hostname>]             <-Display ARP cache
  arp [-v]          [-i <if>] -d  <hostname> [pub][nopub]    <-Delete ARP entry
  arp [-vnD] [<HW>] [-i <if>] -f  [<filename>]              <-Add entry from file
  arp [-v]   [<HW>] [-i <if>] -s  <hostname> <hwaddr> [temp][nopub] <-Add entry
  arp [-v]   [<HW>] [-i <if>] -s  <hostname> <hwaddr> [netmask <nm>] pub  <-''-
  arp [-v]   [<HW>] [-i <if>] -Ds <hostname> <if> [netmask <nm>] pub      <-''-

        -a                      display (all) hosts in alternative (BSD) style
        -s, --set               set a new ARP entry
        -d, --delete            delete a specified entry
        -v, --verbose           be verbose
        -n, --numeric           don't resolve names
        -i, --device            specify network interface (e.g. eth0)
        -D, --use-device        read <hwaddr> from given device
        -A, -p, --protocol      specify protocol family
        -f, --file              read new entries from file or from /etc/ethers

  <HW>=Use '-H <hw>' to specify hardware address type. Default: ether
  List of possible hardware types (which support ARP):
    ether (Ethernet) tr (16/4 Mbps Token Ring) tr (16/4 Mbps Token Ring (New))
    ax25 (AMPR AX.25) netrom (AMPR NET/ROM) arcnet (ARCnet)
    dlci (Frame Relay DLCI) fddi (Fiber Distributed Data Interface) hippi (HIPPI)
    irda (IrLAP) x25 (generic X.25) infiniband (InfiniBand)
```

Comprendre le Chiffrement

Introduction à la cryptologie

Définitions

- **La Cryptologie**
  - La science qui étudie les aspects scientifiques de ces techniques, c'est-à-dire qu'elle englobe la cryptographie et la cryptanalyse.
- **La Cryptanalyse**
  - Lorsque la clef de déchiffrement n'est pas connue de l'attaquant on parle alors de cryptanalyse ou cryptoanalyse (on entend souvent aussi le terme plus familier de cassage).
- **La Cryptographie**
  - Un terme générique désignant l'ensemble des techniques permettant de chiffrer des messages, c'est-à-dire permettant de les rendre inintelligibles sans une action spécifique. Les verbes crypter et chiffrer sont utilisés.
- **Le Décryptement ou Décryptage**
  - Est le fait d'essayer de déchiffrer illégitimement le message (que la clé de déchiffrement soit connue ou non de l'attaquant).



La Cryptographie

La cryptographie apporte quatre points clefs:

- La confidentialité
  - consiste à rendre l'information inintelligible à d'autres personnes que les acteurs de la transaction.
- L'intégrité
  - consiste à déterminer si les données n'ont pas été altérées durant la communication (de manière fortuite ou intentionnelle).
- L'authentification
  - consiste à assurer l'identité d'un utilisateur.
- La non-répudiation
  - est la garantie qu'aucun des correspondants ne pourra nier la transaction.

La cryptographie est basée sur l'arithmétique. Il s'agit, dans le cas d'un texte, de transformer les lettres qui composent le message en une succession de chiffres (sous forme de bits dans le cas de l'informatique), puis ensuite de faire des calculs sur ces chiffres pour:

- Procéder au chiffrement
  - Le résultat de cette modification (le message chiffré) est appelé cryptogramme (Ciphertext) par opposition au message initial, appelé message en clair (Plaintext)
- Procéder au déchiffrement

Le chiffrement se fait à l'aide d'une clef de chiffrement. Le déchiffrement nécessite une clef de déchiffrement.

On distingue deux types de clefs:

- Les clés symétriques:
  - des clés utilisées pour le chiffrement ainsi que pour le déchiffrement. On parle alors de chiffrement symétrique ou de chiffrement à clé secrète.
- Les clés asymétriques:
  - des clés utilisées dans le cas du chiffrement asymétrique (aussi appelé chiffrement à clé publique). Dans ce cas, une clé différente est utilisée pour le chiffrement et pour le déchiffrement.

Le Chiffrement par Substitution

Le chiffrement par substitution consiste à remplacer dans un message une ou plusieurs entités (généralement des lettres) par une ou plusieurs autres entités. On distingue généralement plusieurs types de cryptosystèmes par substitution :

- La substitution **monoalphabétique**

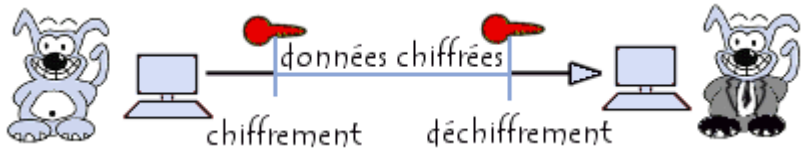
    - consiste à remplacer chaque lettre du message par une autre lettre de l'alphabet
- La substitution **polyalphabétique**
    - consiste à utiliser une suite de chiffres monoalphabétique réutilisée périodiquement
- La substitution **homophonique**
    - permet de faire correspondre à chaque lettre du message en clair un ensemble possible d'autres caractères
- La substitution de **polygrammes**
    - consiste à substituer un groupe de caractères (polygramme) dans le message par un autre groupe de caractères

Algorithmes à clé secrète

Le Chiffrement Symétrique

Ce système est aussi appelé le système à **Clef Secrète** ou à **clef privée**.

Ce système consiste à effectuer une opération de chiffrement par algorithme mais comporte un inconvénient, à savoir qu'il nécessite un canal sécurisé pour la transmission de la clef de chiffrement/déchiffrement.



Le système de Méthode du Masque Jetable (One Time Pad) fût mis au point dans les années 1920. Il utilisait une clef générée aléatoirement à usage unique.

Les algorithmes de chiffrement symétrique couramment utilisés en informatique sont:

- **Data Encryption Standard** (DES),
- **Triple DES** (3DES),
- **RC2**,

- **Blowfish**,
- **International Data Encryption Algorithm** (IDEA),
- **Advanced Encryption Standard** (AES).

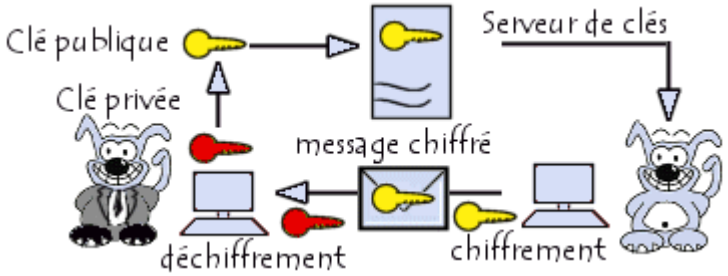
Algorithmes à clef publique

Le Chiffrement Asymétrique

Ce système est aussi appelé **Système à Clef Publique**.

Ce système consiste à avoir deux clefs appelées des **bi-clefs**:

- Une clef **publique** pour le chiffrement
- Une clef **secrète** ou **privée** pour le déchiffrement



- L'utilisateur A (celui qui déchiffre) choisit une clef privée.
- A partir de cette clef il génère plusieurs clefs publiques grâce à un algorithme.
- L'utilisateur B (celui qui chiffre) choisit une des clefs publiques à travers un canal non-sécurisé pour chiffrer les données à l'attention de l'utilisateur A.

Ce système est basé sur ce que l'on appelle une **fonction à trappe à sens unique** ou **one-way trap door**.

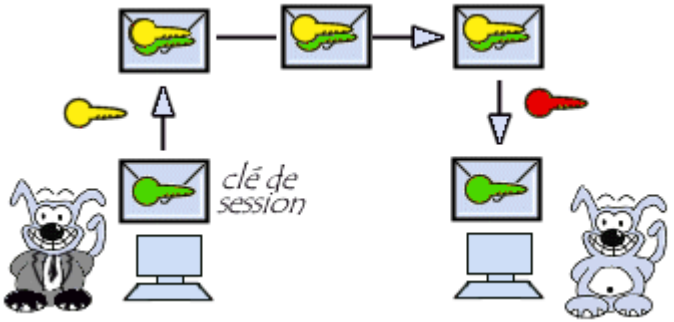
Il existe toutefois un problème – s'assurer que la clef publique récupérée est bien celle qui correspond au destinataire !

Les algorithmes de chiffrement asymétrique couramment utilisés en informatique sont:

- **Digital Signature Algorithm** (DSA)
- **Rivest, Shamir, Adleman** (RSA)

La Clef de Session

Ce système est un compromis entre le système symétrique et le système asymétrique. Il permet l'envoie de données chiffrées à l'aide d'un algorithme de chiffrement symétrique par un canal non-sécurisé et a été mis au point pour palier au problème de lenteur de déchiffrement du système asymétrique.

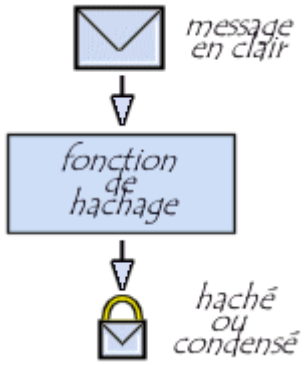


Ce système fonctionne de la façon suivante :

- L'utilisateur A chiffre une clef privée générée aléatoirement, appelée une « clef de session », en utilisant une des clefs publiques de l'utilisateur B.
- L'utilisateur A chiffre les données avec la clef de session.
- L'utilisateur B déchiffre la clef de session en utilisant sa propre clef privée.
- L'utilisateur B déchiffre les données en utilisant la clef de session.

Fonctions de Hachage

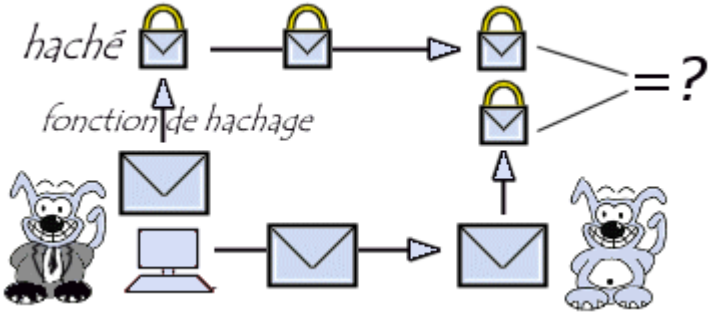
La fonction de **hachage**, aussi appelée une fonction de **condensation**, est à **sens unique** (one way function). Il « condense » un message en clair et produit un haché unique.



Les deux algorithmes de hachage utilisés sont:

- **Message Digest 5** (MD5)
- **Secure Hash Algorithm** (SHA)

Lors de son envoie, le message est accompagné de son haché et il est donc possible de garantir son intégrité:

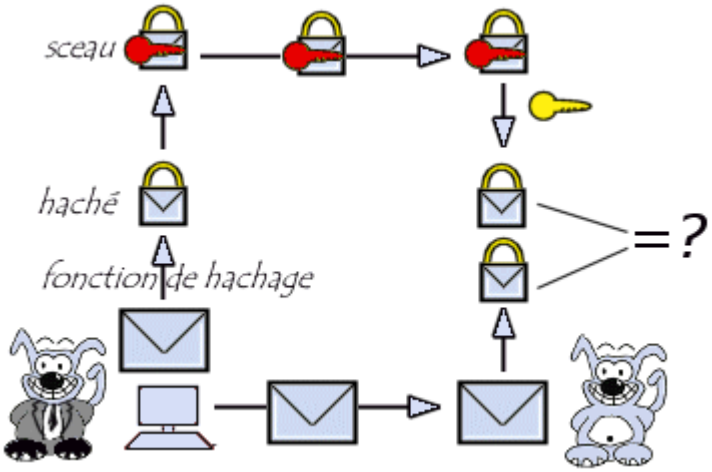


- A la réception du message, le destinataire ou l'utilisateur B calcule le haché du message reçu et le compare avec le haché accompagnant le document.
- Si le message ou le haché a été falsifié durant la communication, les deux empreintes ne correspondront pas.

Ce système permet de vérifier que l'empreinte correspond bien au message reçu, mais ne permet pas de prouver que le message a bien été envoyé par l'utilisateur A.

Signature Numérique

Pour garantir l'authentification du message l'utilisateur A va chiffrer ou **signer** le haché à l'aide de sa clé privée. Le haché signé est appelé un **sceau**.



- L'utilisateur A envoie le sceau au destinataire.
- A la réception du message L'utilisateur B déchiffre le sceau avec la clé publique de l'utilisateur A.
- Il compare le haché obtenu au haché reçu en pièce jointe.

Ce mécanisme de création de sceau est appelé **scellement**.

Ce mécanisme est identique au procédé utilisé par SSH lors d'une connexion

Utilisation de GnuPG

Présentation

GNU Privacy Guard permet aux utilisateurs de transférer des messages chiffrés et/ou signés.

Installation

Sous SLES 12, le paquet gnupg est installé par défaut :

```
SLES12SP1:~ # whereis gpg
gpg: /usr/bin/gpg /usr/share/man/man1/gpg.1.gz
```

Configuration

Pour initialiser GnuPG, saisissez la commande suivante :

```
trainee@SLES12SP1:~> gpg
gpg: répertoire « /home/trainee/.gnupg » créé
gpg: nouveau fichier de configuration « /home/trainee/.gnupg/gpg.conf » créé
gpg: Attention : les options de « /home/trainee/.gnupg/gpg.conf » ne sont pas encore actives cette fois
gpg: le porte-clefs « /home/trainee/.gnupg/secring.gpg » a été créé
gpg: le porte-clefs « /home/trainee/.gnupg/pubring.gpg » a été créé
gpg: Vous pouvez taper votre message…
^C
gpg: signal Interrupt caught ... exiting

trainee@SLES12SP1:~>
```

Pour générer les clefs, saisissez la commande suivante :

```
trainee@SLES12SP1:~> gpg --gen-key
gpg (GnuPG) 2.0.24; Copyright (C) 2013 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Sélectionnez le type de clef désiré :
   (1) RSA et RSA (par défaut)
```

```
   (2) DSA et Elgamal
   (3) DSA (signature seule)
   (4) RSA (signature seule)
Quel est votre choix ? 1
les clefs RSA peuvent faire entre 1024 et 4096 bits de longueur.
Quelle taille de clef désirez-vous ? (2048)
La taille demandée est 2048 bits
Veuillez indiquer le temps pendant lequel cette clef devrait être valable.
         0 = la clef n'expire pas
      <n>  = la clef expire dans n jours
      <n>w = la clef expire dans n semaines
      <n>m = la clef expire dans n mois
      <n>y = la clef expire dans n ans
Pendant combien de temps la clef est-elle valable ? (0) 0
La clef n'expire pas du tout
Est-ce correct ? (o/N) o

GnuPG doit construire une identité pour identifier la clef.

Nom réel : I2TCH
Adresse électronique : infos@i2tch.eu
Commentaire : test
Vous avez sélectionné cette identité :
    « I2TCH (test) <infos@i2tch.eu> »

Faut-il modifier le (N)om, le (C)ommentaire, l'(A)dresse électronique
ou (O)ui/(Q)uitter ? O
Une phrase de passe est nécessaire pour protéger votre clef secrète.
fenestros
De nombreux octets aléatoires doivent être générés. Vous devriez faire
autre chose (taper au clavier, déplacer la souris, utiliser les disques)
pendant la génération de nombres premiers ; cela donne au générateur de
nombres aléatoires une meilleure chance d'obtenir suffisamment d'entropie.
De nombreux octets aléatoires doivent être générés. Vous devriez faire
```

```
autre chose (taper au clavier, déplacer la souris, utiliser les disques)
pendant la génération de nombres premiers ; cela donne au générateur de
nombres aléatoires une meilleure chance d'obtenir suffisamment d'entropie.
gpg: /home/trainee/.gnupg/trustdb.gpg : base de confiance créée
gpg: clef 64FF9193 marquée de confiance ultime.
les clefs publique et secrète ont été créées et signées.

gpg: vérification de la base de confiance
gpg: 3 marginale(s) nécessaire(s), 1 complète(s) nécessaire(s),
     modèle de confiance PGP
gpg: profondeur : 0  valables :   1  signées :   0
     confiance : 0 i., 0 n.d., 0 j., 0 m., 0 t., 1 u.
pub   2048R/64FF9193 2017-11-14
 Empreinte de la clef = 8C24 F85A D0E9 0679 935E  41D6 3B7B 9077 64FF 9193
uid      [  ultime ] I2TCH (test) <infos@i2tch.eu>
sub   2048R/A227C3A1 2017-11-14
```

La liste de clefs peut être visualisée avec la commande suivante :

```
trainee@SLES12SP1:~> gpg --list-keys
/home/trainee/.gnupg/pubring.gpg
--------------------------------
pub   2048R/64FF9193 2017-11-14
uid      [  ultime ] I2TCH (test) <infos@i2tch.eu>
sub   2048R/A227C3A1 2017-11-14
```

Pour importer la clef d'un correspondant dans sa trousse de clefs il convient d'utiliser la commande suivante :

```
# gpg --import la-clef.asc
```

Pour exporter sa clef publique, il convient d'utiliser la commande suivante :

```
trainee@SLES12SP1:~> gpg --export --armor I2TCH > ~/I2TCH.asc
trainee@SLES12SP1:~> cat I2TCH.asc
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v2

mQENBFoK9xYBCADChnxR2MJmQ4tYgclogJl+NJgmaP70uHCNjYdab+g6tFFWjq/M
4M5m68FuBvhRWzbsLrWLr+wZNYOmpX7rXIeVprcF8TRhe6aW+tEgKxlQoI6Wizyh
/VqlfCRVKiB5CS2Zr88ZSVqxkcEPYDa3tLRHnuRHIIuyP02tsM/szTJ8TT9eIoaL
iwrOj9EOyOkhIZ1x7y/9bKIifTFtF1H1ya5+YsCMaxLOAw2ViBJ3ni7Shr5MQL0Q
KrjJkhex9LYDhhL9HsccFxBBKjVySDb4mUjcguZx6nvsOoV7JaAJUn3gwVBRf3RP
K3VKqo06bbLTEfzvazZJeSWubcZU6KMoMZtdABEBAAG0HUkyVENIICh0ZXN0KSA8
aW5mb3NAaTJ0Y2guZXU+iQE5BBMBAgAjBQJaCvcWAhsDBwsJCAcDAgEGFQgCCQoL
BBYCAwECHgECF4AACgkQO3uQd2T/kZPwxQf/TZpOqwZYWFGwTE9PkU5FHMgY4+hb
rDv/jkSyPuPxw6SUIBBm11Kpxdd0e6Ed1Lo/j4mnWVfJ15QZvTw4dy+lXjRkHTAt
MTdzkX5JifFprog563JgqaV1U7ep4xX1cg3iHHRzUgFTiPN/ex8/vOsNB1jTvh6Q
HdiKZLQ6UxZf9ME13A3VB0qzfbndcHWgh8wsjl3Vh2gzSDCeJI0T+PaDWLvTkQE5
qIAbD+k4+rrE5VaJxJeJ0Q+77hIK4/c2hAvYnqswqs+sI0p8kzdYkD/56YiJqADB
JzKG/fFfJ3jLlr9JnjqyrvfixnL49eGmDvp6C6O3FC+NzhsZAuMLnyt4gLkBDQRa
CvcWAQgAuwgkNpK+1dp1PabtOMPYLnc8y4pKHmwxueEH4wjJIFp2BPshxNHtgECV
DML0jpOIc5jgBJIoqNa747XF3D4c5LdTRjSUrLsdl94dMNQgo2zuVwWbwzZlUuch
QnGe3KuBEm6LB1icYOh7WqCqLL7CZUHh++/0spNSqdOii2WbbZqTdLFM8ON2Em68
m8g5anDSPuoSR8mDx75ByJ5knkkvXJMVKDTTL31DDFez9DWR1uu5j+pylrB4bfnj
MEBmMk5T3/VgjO6ee+GvpssSTQx+ZvG8NLHDTlS4Dtr0ERtIcGXk2SgfoKY/fDYg
66641Bmafl+k1tzgvt2G6Q6qpUuKUwARAQABiQEfBBgBAgAJBQJaCvcWAhsMAAoJ
EDt7kHdk/5GTntYH/1UAPnpPIDff/tNPqJqwJBHES/NOL5qJRm+rDUWKy7XXmxQX
PEGfShaN4COT3/4IQbTpmjae7LFNItO6gkDOQxgP0S1eChKpKNtguCjpj0w59F4Q
1gPA/vKXLibqx/RtrKONFGHAMHQXZazPBFsNXsc8IkV7akhzsJGk9yvksLM3yQzs
Xrh7GUfQ+tF0E9QuXULoXy07FPX7+J2+JoY2YM+of46w2TRP5CpwqO+TTTw9qgWo
dhFyxj3n53LxBHwGGHMENpTndpnqg2ZKXRIoBPEJNc4k+GFe3WUXV5rNOFTunOAh
cFyQ3GE9bDDZiHpUvmJyvCPTGrNuu3U+YDDEH04=
=gB61
-----END PGP PUBLIC KEY BLOCK-----
```

Cette clef peut ensuite être jointe à des messages électroniques ou bien déposée sur un serveur de clefs tel http://www.keyserver.net.

Signer un message

Créez maintenant un message à signer :

```
trainee@SLES12SP1:~> vi ~/message.txt
trainee@SLES12SP1:~> cat ~/message.txt
This is a test message for gpg
```

Pour signer ce message en format binaire, il convient d'utiliser la commande suivante :

```
trainee@SLES12SP1:~> vi ~/message.txt
trainee@SLES12SP1:~> cat ~/message.txt
This is a test message for gpg
trainee@SLES12SP1:~> gpg --default-key I2TCH --detach-sign message.txt

Une phrase de passe est nécessaire pour déverrouiller la clef secrète de
l'utilisateur : « I2TCH (test) <infos@i2tch.eu> »
clef RSA de 2048 bits, identifiant 64FF9193, créée le 2017-11-14

trainee@SLES12SP1:~> ls -l | grep message
-rw-r--r-- 1 trainee users   31 14 nov.  15:05 message.txt
-rw-r--r-- 1 trainee users  287 14 nov.  15:06 message.txt.sig
trainee@SLES12SP1:~> cat message.txt.sig
□□□□□□□□Z
□□
    □;{□wd□□□□M□+1□PQ@□□□Sp"S□□□Ų□□□□□(N    □□□E٩◈3□□?{□□□□+>□$□□ n□w2□□O□{+Z}□□□□□Q□U    □-X=8□X□O□□□□f
□(_Q<G□□i□|□□Re□□□□□□□□2i4|□□□□□6□□□[QX84□□3□□□□]□□tV□w□u□⁎□H□□1□L□□□□□□BK□@□K*□?□□⁝Y]D-□□□□□□#□z □Qq□□zu2D□E
□□□`
□□trainee@SLES12SP1:~> □□Ů□□}□e□□O□□□N□□
```

Pour signer ce message en format ascii, il convient d'utiliser la commande suivante :

```
trainee@SLES12SP1:~> gpg --default-key I2TCH --armor --detach-sign message.txt
```

```
Une phrase de passe est nécessaire pour déverrouiller la clef secrète de
l'utilisateur : « I2TCH (test) <infos@i2tch.eu> »
clef RSA de 2048 bits, identifiant 64FF9193, créée le 2017-11-14

trainee@SLES12SP1:~> ls -l | grep message
-rw-r--r-- 1 trainee users   31 14 nov.  15:05 message.txt
-rw-r--r-- 1 trainee users  473 14 nov.  15:08 message.txt.asc
-rw-r--r-- 1 trainee users  287 14 nov.  15:06 message.txt.sig
trainee@SLES12SP1:~> cat message.txt.asc
-----BEGIN PGP SIGNATURE-----
Version: GnuPG v2

iQEcBAABAgAGBQJaCvjtAAoJEDt7kHdk/5GTlrwIAI5yoWPlX12VW3Ve/P4ZlTEC
vg6C2YfFpQDZmqt7VaiqvXSE0nzoerHjt+GCjSa5nEjSCk3M8X8BZiJmzX58o2VU
nJ+ZbHDqa9yIG1bgJQnBPwnVYF4GqlrWqHJMRf+NKU6/2fRDUF9h1HmgSTtHg7Q/
gwZ9dy1e4GZws8oCYQgJGdTcSUcA0hiYjFk7Ixj0Ck/nCPJzy7jEIlpCext1oRLZ
mXi64DC9eUbJ40DSjzIMMFXBfjtzfR4op8zXoo+Wj0/6jSoFnkcupMKZfuQJL3y3
seaHP1HS5KNhYoCe8jODAytbnOxJVbXLnC6iuFyp8y2cUzCmARrGKBx2eSb9DcA=
=4B3p
-----END PGP SIGNATURE-----
```

Pour vérifier la signature d'un message signé en mode ascii, il convient d'utiliser la commande :

```
trainee@SLES12SP1:~> gpg --verify message.txt.asc
gpg: Signature faite le mar. 14 nov. 2017 15:08:45 CET avec la clef RSA d'identifiant 64FF9193
gpg: Bonne signature de « I2TCH (test) <infos@i2tch.eu> » [ultime]
```

Important - Pour vérifier la signature d'un message signé en mode ascii et produit en dehors du message lui-même, il convient d'utiliser la commande :

```
# gpg --verify message.txt.asc message.txt
```

Pour signer ce message **dans le message lui-même** en format ascii, il convient d'utiliser la commande suivante :

```
trainee@SLES12SP1:~> gpg --default-key I2TCH --clearsign message.txt

Une phrase de passe est nécessaire pour déverrouiller la clef secrète de
l'utilisateur : « I2TCH (test) <infos@i2tch.eu> »
clef RSA de 2048 bits, identifiant 64FF9193, créée le 2017-11-14

Le fichier « message.txt.asc » existe. Faut-il réécrire par-dessus ? (o/N) o
trainee@SLES12SP1:~> ls -l | grep message
-rw-r--r-- 1 trainee users   31 14 nov.  15:05 message.txt
-rw-r--r-- 1 trainee users  551 14 nov.  15:11 message.txt.asc
-rw-r--r-- 1 trainee users  287 14 nov.  15:06 message.txt.sig
trainee@SLES12SP1:~> cat message.txt.asc
-----BEGIN PGP SIGNED MESSAGE-----
Hash: SHA1

This is a test message for gpg
-----BEGIN PGP SIGNATURE-----
Version: GnuPG v2

iQEcBAEBAgAGBQJaCvl/AAoJEDt7kHdk/5GTubkIAJy4+RtpYLWwH2in5n1bDsd+
qrdml9/d0SXZ3OcUaxQXu0Vqozl364sWSy1PtK5wUmrDTlYS9TQrRGEyhVtILmBZ
6G8TSaSVymrlX5vkEIgM+GatNzvWa943fJLXKcb0+MLykbZTsToBp9+3z/tLRReX
IAJkx3beHStz22b6ZBxsfpcuRIG+54uRTzjaqXZrPI5UPxxzvMrRaFn2rg0eLCwn
C4v8mKQg/6LuiEqPd1h+MTYwL/OnSWM5IS7IBeayRgsjhusEh4skQToJPGp1EQSe
LfL8BhMKe1HsQvo2qUMV7Hcb3p0q56E7LZqbHk0YZbyAzERsaYUWFpp8CtW/S/Y=
=sPxS
-----END PGP SIGNATURE-----
```

Chiffrer un message

Pour chiffrer un message, il faut disposer de la clef publique du destinataire du message. Ce dernier utilisera ensuite sa clef privée pour déchiffrer le

message. Il convient de préciser le destinataire du message, ou plus précisément la clef publique à utiliser, lors d'un chiffrement :

```
  gpg --recipient <destinataire> --encrypt <message>
```

- <*destinataire*> représente toute information permettant de distinguer sans ambigüité une clef publique dans votre trousseau. Cette information peut-être le nom ou l'adresse email associé à la clef publique que vous voulez utiliser,
- <*message*> représente le message à chiffrer.

Par exemple pour chiffrer un message en mode binaire, il convient de saisir la commande suivante :

```
trainee@SLES12SP1:~> gpg --recipient I2TCH --encrypt message.txt
trainee@SLES12SP1:~> ls -l | grep message
-rw-r--r-- 1 trainee users   31 14 nov.  15:05 message.txt
-rw-r--r-- 1 trainee users  551 14 nov.  15:11 message.txt.asc
-rw-r--r-- 1 trainee users  367 14 nov.  15:12 message.txt.gpg
-rw-r--r-- 1 trainee users  287 14 nov.  15:06 message.txt.sig
trainee@SLES12SP1:~> cat message.txt.gpg
��
  ?"x??&?'á??{~?u??QJ?P?,o??B??I1v2?n?K)L j?[
                                                ??e??PZ???8??;?????????? ??V:LĞ??V!?????X?lN,?J>??I??##?|????d?R?
|N????????e?.?k??1??rzm>68????????$:????`?(JO?^?????C????p???????"??>????dk???|?s[?5?C????H??w??t7?mW??f<zOL??g?
._??o3Ī?????M???7f?7H??_?#?^?????#??3q?????}????^UK2L?1~$
??/}N/&jR??|?????.??~zg??p??`s?_??&???Zr?????0s?K.?????F??cȝ?+%trainee@SLES12SP1:~>
trainee@15:~>
```

Et pour chiffrer un message en mode ascii, il convient de saisir la commande suivante :

```
trainee@SLES12SP1:~> gpg --recipient I2TCH --armor --encrypt message.txt
Le fichier « message.txt.asc » existe. Faut-il réécrire par-dessus ? (o/N) o
trainee@SLES12SP1:~> ls -l | grep message
-rw-r--r-- 1 trainee users   31 14 nov.  15:05 message.txt
-rw-r--r-- 1 trainee users  579 14 nov.  15:14 message.txt.asc
-rw-r--r-- 1 trainee users  367 14 nov.  15:12 message.txt.gpg
-rw-r--r-- 1 trainee users  287 14 nov.  15:06 message.txt.sig
```

```
trainee@SLES12SP1:~> cat message.txt.asc
-----BEGIN PGP MESSAGE-----
Version: GnuPG v2

hQEMAyJ44CaiJ8OhAQf/T+KQWmorQ0ceFZKDm5GQafuqWiU773f5l2Q2gJCkL+vJ
eaudPXBlBHxynzdT25oIzxND9OInN4ahVUkU7lVOMQ0lA5FGmWzwhOttkH2fHzD4
rXoAEQTVaKS6fuXOiSChLg8Wg5nzudnYAEPL46owQLuXLvbqRe+Jjb+I+LdEt3bM
uZX5XWYPBpHqDcE0n8jGypoLYvHhyWDPlHsONL7pbyFDIqcZf6gCLq9tBfYVEuxL
Mhxaq9rj2c9lKzLvc8LHee12YKurh8nQuueFtEx4E+dTjr34P4AOTgEOGQ39h+aN
awrsBfMZAn/pPqJimG0rySQAGWxD9sPZOD84cWU4U9JeAU3PjTUxlR+i5G8g0+r6
8dWz5tRmk3p2WX2qlGe6SvbBATG284wGCuU2QkmYNnaeLTbkVSedL9dWLipYSmTB
d/AQOl0WYF+3TnYaHsU7E1axpJFt+v7s8eMkm85kSQ==
=xFf0
-----END PGP MESSAGE-----
```

Pour décrypter un message il convient d'utiliser la commande suivante :

```
trainee@SLES12SP1:~> gpg --decrypt message.txt.asc

Une phrase de passe est nécessaire pour déverrouiller la clef secrète de
l'utilisateur : « I2TCH (test) <infos@i2tch.eu> »
clef RSA de 2048 bits, identifiant A227C3A1, créée le 2017-11-14 (identifiant de clef principale 64FF9193)

gpg: chiffré avec une clef RSA de 2048 bits, identifiant A227C3A1, créée le 2017-11-14
      « I2TCH (test) <infos@i2tch.eu> »
This is a test message for gpg
```

PKI

On appelle **PKI** (Public Key Infrastucture, ou en français **infrastructure à clé publique (ICP)**, parfois **infrastructure de gestion de clés (IGC)**) l’ensemble des solutions techniques basées sur la cryptographie à clé publique.

Les cryptosystèmes à clés publiques permettent de s'affranchir de la nécessité d'avoir recours systématiquement à un canal sécurisé pour s'échanger

les clés. En revanche, la publication de la clé publique à grande échelle doit se faire en toute confiance pour assurer que :

- La clé publique est bien celle de son propriétaire ;
- Le propriétaire de la clé est digne de confiance ;
- La clé est toujours valide.

Ainsi, il est nécessaire d'associer au bi-clé (ensemble clé publique / clé privée) un certificat délivré par un **tiers de confiance** : l'infrastructure de gestion de clés.

Le tiers de confiance est une entité appelée communément autorité de certification (ou en anglais Certification authority, abrégé CA) chargée d'assurer la véracité des informations contenues dans le certificat de clé publique et de sa validité.

Pour ce faire, l'autorité signe le certificat de clé publique à l'aide de sa propre clé en utilisant le principe de signature numérique.

Le rôle de l'infrastructure de clés publiques est multiple et couvre notamment les champs suivants :

- enregistrer des demandes de clés en vérifiant l'identité des demandeurs ;
- générer les paires de clés (clé privée / clé publique) ;
- garantir la confidentialité des clés privées correspondant aux clés publiques ;
- certifier l'association entre chaque utilisateurs et sa clé publique ;
- révoquer des clés (en cas de perte par son propriétaire, d'expiration de sa date de validité ou de compromission).

Une infrastructure à clé publique est en règle générale composée de trois entités distinctes :

- L'autorité d'enregistrement (AE ou RA pour Recording authority), chargée des formalité administratives telles que la vérification de l'identité des demandeurs, le suivi et la gestion des demandes, etc.) ;
- L'autorité de certification (AC ou CA pour Certification Authority), chargée des tâches techniques de création de certificats. L'autorité de certification est ainsi chargée de la signature des demandes de certificat (CSR pour Certificate Signing Request, parfois appelées PKCS#10, nom du format correspondant). L'autorité de certification a également pour mission la signature des listes de révocations (CRL pour Certificate Revocation List) ;
- L'Autorité de dépôt (Repository) dont la mission est de conserver en sécurité les certificats.

Certificats X509

Pour palier aux problèmes liés à des clefs publiques piratées, un système de certificats a été mis en place.

Le certificat permet d'associer la clef publique à une entité ou une personne. Les certificats sont délivrés par des Organismes de Certification.

Les certificats sont des fichiers divisés en deux parties :

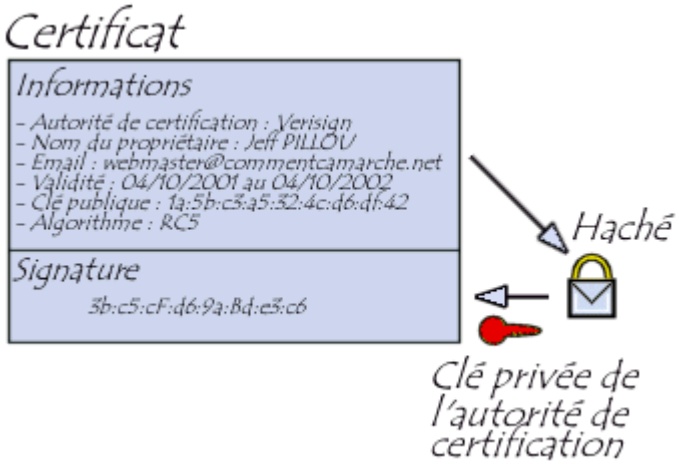
- La partie contenant les informations
- La partie contenant la signature de l'autorité de certification

La structure des certificats est normalisée par le standard **X.509** de l'**Union internationale des télécommunications**.

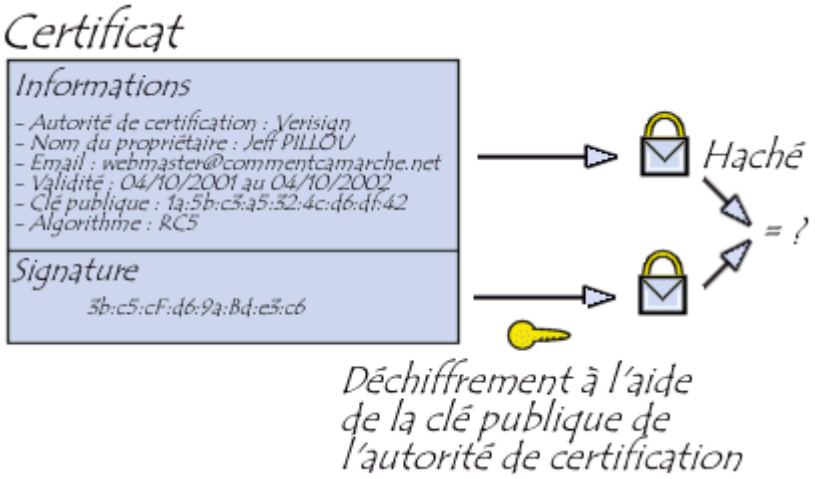
Elle contient :

- Le nom de l'autorité de certification
- Le nom du propriétaire du certificat
- La date de validité du certificat
- L'algorithme de chiffrement utilisé
- La clé publique du propriétaire

Le Certificat est signé par l'autorité de certification:



La vérification se passe ainsi:



Configuration du Réseau

Configuration de TCP/IP

La configuration TCP/IP se trouve dans le répertoire **/etc/sysconfig/network**. Les fichiers importants sont :

Le Fichier /etc/sysconfig/network/config

Ce fichier contient des directives applicables à toutes les interfaces réseau :

```
SLES12SP1:~ # cat /etc/sysconfig/network/config
## Type:        integer
## Default:     ""
#
# How log to wait for IPv6 autoconfig in ifup when requested with
# the auto6 or +auto6 tag in BOOTPROTO variable.
```

```
# When unset, a wicked built-in default defer time (10sec) is used.
#
AUTO6_WAIT_AT_BOOT=""

## Type:        list(all,dns,none,"")
## Default:     ""
#
# Whether to update system (DNS) settings from IPv6 RA when requested
# with the auto6 or +auto6 tag in BOOTPROTO variable.
# Defaults to update if autoconf sysctl (address autoconf) is enabled.
#
AUTO6_UPDATE=""

## Type:        list(auto,yes,no)
## Default:     "auto"
#
# Permits to specify/modify a global ifcfg default. Use with care!
#
# This settings breaks rules for many things, which require carrier
# before they can start, e.g. L2 link protocols, link authentication,
# ipv4 duplicate address detection, ipv6 duplicate detection will
# happen "post-mortem" and maybe even cause to disable ipv6 at all.
# See also "man ifcfg" for further informations.
#
LINK_REQUIRED="auto"

## Type:        string
## Default:     ""
#
# Allows to specify a comma separated list of debug facilities used
# by wicked. Negated facility names can be prepended by a "-", e.g.:
#   "all,-events,-socket,-objectmodel,xpath,xml,dbus"
#
# When set, wicked debug level is automatically enabled.
```

```
# For a complete list of facility names, see: "wicked --debug help".
#
WICKED_DEBUG=""

## Type:        list("",error,warning,notice,info,debug,debug1,debug2,debug3)
## Default:     ""
#
# Allows to specify wicked debug level. Default level is "notice".
#
WICKED_LOG_LEVEL=""
## Path:    Network/General
## Description: Global network configuration
#
# Note:
# Most of the options can and should be overridden by per-interface
# settings in the ifcfg-* files.
#
# Note: The ISC dhclient started by the NetworkManager is not using any
# of these options -- NetworkManager is not using any sysconfig settings.
#

## Type:        yesno
## Default:     yes
# If ifup should check if an IPv4 address is already in use, set this to yes.
#
# Make sure that packet sockets (CONFIG_PACKET) are supported in the kernel,
# since this feature uses arp, which depends on that.
# Also be aware that this takes one second per interface; consider that when
# setting up a lot of interfaces.
CHECK_DUPLICATE_IP="yes"

## Type:        list(auto,yes,no)
## Default:     auto
# If ifup should send a gratuitous ARP to inform the receivers about its
```

```
# IPv4 addresses. Default is to send gratuitous ARP, when duplicate IPv4
# address check is enabled and the check were sucessful.
#
# Make sure that packet sockets (CONFIG_PACKET) are supported in the kernel,
# since this feature uses arp, which depends on that.
SEND_GRATUITOUS_ARP="auto"

## Type:        yesno
## Default:     no
# Switch on/off debug messages for all network configuration stuff. If set to no
# most scripts can enable it locally with "-o debug".
DEBUG="no"

## Type:    integer
## Default: 30
#
# Some interfaces need some time to come up or come asynchronously via hotplug.
# WAIT_FOR_INTERFACES is a global wait for all mandatory interfaces in
# seconds. If empty no wait occurs.
#
WAIT_FOR_INTERFACES="30"

## Type:    yesno
## Default: yes
#
# With this variable you can determine if the SuSEfirewall when enabled
# should get started when network interfaces are started.
FIREWALL="yes"

## Type:    int
## Default: 30
#
# When using NetworkManager you may define a timeout to wait for NetworkManager
# to connect in NetworkManager-wait-online.service.  Other network services
```

```
# may require the system to have a valid network setup in order to succeed.
#
# This variable has no effect if NetworkManager is disabled.
#
NM_ONLINE_TIMEOUT="30"

## Type:        string
## Default:     "dns-resolver dns-bind ntp-runtime nis"
#
# This variable defines the start order of netconfig modules installed
# in the /etc/netconfig.d/ directory.
#
# To disable the execution of a module, don't remove it from the list
# but prepend it with a minus sign, "-ntp-runtime".
#
NETCONFIG_MODULES_ORDER="dns-resolver dns-bind dns-dnsmasq nis ntp-runtime"

## Type:        yesno
## Default:     no
#
# Enable netconfig verbose reporting.
#
NETCONFIG_VERBOSE="no"

## Type:    yesno
## Default: no
#
# This variable enables netconfig to always force a replace of modified
# files and automatically enables the -f | --force-replace parameter.
#
# The purpose is to use it as workaround, when some other tool trashes
# the files, e.g. /etc/resolv.conf and you observe messages like this
# in your logs on in "netconfig update" output:
# ATTENTION: You have modified /etc/resolv.conf. Leaving it untouched.
```

```
#
# Please do not forget to also report a bug as we have a system policy
# to use netconfig.
#
NETCONFIG_FORCE_REPLACE="no"

## Type:        string
## Default:     "auto"
#
# Defines the DNS merge policy as documented in netconfig(8) manual page.
# Set to "" to disable DNS configuration.
#
NETCONFIG_DNS_POLICY="auto"

## Type:        string(resolver,bind,dnsmasq,)
## Default:     "resolver"
#
# Defines the name of the DNS forwarder that has to be configured.
# Currently implemented are "bind", "dnsmasq" and "resolver", that
# causes to write the name server IP addresses to /etc/resolv.conf
# only (no forwarder). Empty string defaults to "resolver".
#
NETCONFIG_DNS_FORWARDER="resolver"

## Type:        yesno
## Default:     yes
#
# When enabled (default) in forwarder mode ("bind", "dnsmasq"),
# netconfig writes an explicit localhost nameserver address to the
# /etc/resolv.conf, followed by the policy resolved name server list
# as fallback for the moments, when the local forwarder is stopped.
#
NETCONFIG_DNS_FORWARDER_FALLBACK="yes"
```

```
## Type:        string
## Default:     ""
#
# List of DNS domain names used for host-name lookup.
# It is written as search list into the /etc/resolv.conf file.
#
NETCONFIG_DNS_STATIC_SEARCHLIST=""

## Type:        string
## Default:     ""
#
# List of DNS nameserver IP addresses to use for host-name lookup.
# When the NETCONFIG_DNS_FORWARDER variable is set to "resolver",
# the name servers are written directly to /etc/resolv.conf.
# Otherwise, the nameserver are written into a forwarder specific
# configuration file and the /etc/resolv.conf does not contain any
# nameservers causing the glibc to use the name server on the local
# machine (the forwarder). See also netconfig(8) manual page.
#
NETCONFIG_DNS_STATIC_SERVERS=""

## Type:        string
## Default:     "auto"
#
# Allows to specify a custom DNS service ranking list, that is which
# services provide preferred (e.g. vpn services), and which services
# fallback settings (e.g. avahi).
# Preferred service names have to be prepended with a "+", fallback
# service names with a "-" character. The special default value
# "auto" enables the current build-in service ranking list -- see the
# netconfig(8) manual page -- "none" or "" disables the ranking.
#
NETCONFIG_DNS_RANKING="auto"
```

```
## Type:        string
## Default:     ""
#
# Allows to specify options to use when writting the /etc/resolv.conf,
# for example:
#   "debug attempts:1 timeout:10"
# See resolv.conf(5) manual page for details.
#
NETCONFIG_DNS_RESOLVER_OPTIONS=""

## Type:        string
## Default:     ""
#
# Allows to specify a sortlist to use when writting the /etc/resolv.conf,
# for example:
#   130.155.160.0/255.255.240.0 130.155.0.0"
# See resolv.conf(5) manual page for details.
#
NETCONFIG_DNS_RESOLVER_SORTLIST=""

## Type:        string
## Default:     "auto"
#
# Defines the NTP merge policy as documented in netconfig(8) manual page.
# Set to "" to disable NTP configuration.
#
NETCONFIG_NTP_POLICY="auto"

## Type:        string
## Default:     ""
#
# List of NTP servers.
#
NETCONFIG_NTP_STATIC_SERVERS=""
```

```
## Type:        string
## Default:     "auto"
#
# Defines the NIS merge policy as documented in netconfig(8) manual page.
# Set to "" to disable NIS configuration.
#
NETCONFIG_NIS_POLICY="auto"

## Type:        string(yes,no,)
## Default:     "yes"
#
# Defines whether to set the default NIS domain. When enabled and no domain
# is provided dynamically or in static settings, /etc/defaultdomain is used.
# Valid values are:
#  - "no" or ""         netconfig does not set the domainname
#  - "yes"              netconfig sets the domainname according to the
#                       NIS policy using settings provided by the first
#                       iterface and service that provided it.
#  - "<interface name>" as yes, but only using settings from interface.
#
NETCONFIG_NIS_SETDOMAINNAME="yes"

## Type:        string
## Default:     ""
#
# Defines a default NIS domain.
#
# Further domain can be specified by adding a "_<number>" suffix to
# the NETCONFIG_NIS_STATIC_DOMAIN and NETCONFIG_NIS_STATIC_SERVERS
# variables, e.g.: NETCONFIG_NIS_STATIC_DOMAIN_1="second".
#
NETCONFIG_NIS_STATIC_DOMAIN=""

## Type:        string
```

```
## Default:     ""
#
# Defines a list of NIS servers for the default NIS domain or the
# domain specified with same "_<number>" suffix.
#
NETCONFIG_NIS_STATIC_SERVERS=""

## Type:    string
## Default: ''
#
# Set this variable global variable to the ISO / IEC 3166 alpha2
# country code specifying the wireless regulatory domain to set.
# When not empty, ifup-wireless will be set in the wpa_supplicant
# config or via 'iw reg set' command.
#
# Note: This option requires a wpa driver supporting it, like
# the 'nl80211' driver used by default since openSUSE 11.3.
# When you notice problems with your hardware, please file a
# bug report and set e.g. WIRELESS_WPA_DRIVER='wext' (the old
# default driver) in the ifcfg file.
# See also "/usr/sbin/wpa_supplicant --help" for the list of
# available wpa drivers.
#
WIRELESS_REGULATORY_DOMAIN=''
```

Les directives activées de ce fichier sont :

```
SLES12SP1:~ # egrep -v '^(#|$)' /etc/sysconfig/network/config > /tmp/config
SLES12SP1:~ # cat /tmp/config
AUTO6_WAIT_AT_BOOT=""
AUTO6_UPDATE=""
LINK_REQUIRED="auto"
WICKED_DEBUG=""
WICKED_LOG_LEVEL=""
```

```
CHECK_DUPLICATE_IP="yes"
SEND_GRATUITOUS_ARP="auto"
DEBUG="no"
WAIT_FOR_INTERFACES="30"
FIREWALL="yes"
NM_ONLINE_TIMEOUT="30"
NETCONFIG_MODULES_ORDER="dns-resolver dns-bind dns-dnsmasq nis ntp-runtime"
NETCONFIG_VERBOSE="no"
NETCONFIG_FORCE_REPLACE="no"
NETCONFIG_DNS_POLICY="auto"
NETCONFIG_DNS_FORWARDER="resolver"
NETCONFIG_DNS_FORWARDER_FALLBACK="yes"
NETCONFIG_DNS_STATIC_SEARCHLIST=""
NETCONFIG_DNS_STATIC_SERVERS=""
NETCONFIG_DNS_RANKING="auto"
NETCONFIG_DNS_RESOLVER_OPTIONS=""
NETCONFIG_DNS_RESOLVER_SORTLIST=""
NETCONFIG_NTP_POLICY="auto"
NETCONFIG_NTP_STATIC_SERVERS=""
NETCONFIG_NIS_POLICY="auto"
NETCONFIG_NIS_SETDOMAINNAME="yes"
NETCONFIG_NIS_STATIC_DOMAIN=""
NETCONFIG_NIS_STATIC_SERVERS=""
WIRELESS_REGULATORY_DOMAIN=''
```

Important - Veuillez noter que chaque directive est détaillée dans le fichier lui-même. Notez aussi qu'à partir du SP1 de SUSE celui-ci n'utilise pas NetworkManager par défaut ni le système de scripts **if***, à savoir **ifup** et **ifdown** pour contrôler l'interface réseau. Un nouveau système a été introduit, appelé **wicked**.

Le Fichier /etc/sysconfig/network/dhcp

Ce fichier spécifie les valeurs des directives utilisées par **dhcpcd** quand l'interface réseau est configurée en mode **dhcp** :

```
SLES12SP1:~ # cat /etc/sysconfig/network/dhcp
## Type:        list(enabled,disabled,default,)
## Default:     ""
#
# Default is to use the FQDN option, when the DHCLIENT_HOSTNAME_OPTION
# variable is set to a full hostname, that is, when it contains a dot.
# When DHCLIENT_HOSTNAME_OPTION is set to AUTO, short hostname from
# /etc/hostname is send via hostname option 12 (same as SLES-11).
#
DHCLIENT_FQDN_ENABLED=""

## Type:        list(both,ptr,none,)
## Default:     ""
#
# Request to update A and PTR or only the PTR DNS records using the
# hostname specified in DHCLIENT_HOSTNAME_OPTION variable.
# Default is to update 'both' when hostname is set or 'none' when
# the hostname is empty and DHCLIENT_FQDN_ENABLED is set to enabled.
#
DHCLIENT_FQDN_UPDATE=""

## Type:        yesno
## Default:     yes
#
# Qualify relative sub-domains/hostname in the DHCLIENT_HOSTNAME_OPTION
# variable adding a final dot ('foo.bar' -> 'foo.bar.').
# When disabled, the DHCP server may append it's update domain to the
# hostname (e.g. 'foo.bar' -> 'foo.bar.example.net').
#
```

```
DHCLIENT_FQDN_QUALIFY="yes"

## Type:        yesno
## Default:     yes
#
# The FQDN option is encoding hostnames using canonical DNS wire format
# by default. This flag permits to enable use of the deprecated ascii
# format limited to a single label (host hostname) for compatibility
# purposes with draft implementation, which may be unsupported and cause
# that a DHCP server ignores the fqdn option request completely.
#
DHCLIENT_FQDN_ENCODE="yes"

## Type:        list(enabled,disabled,default,)
## Default:     ""
#
# Default is to use the FQDN option, when the DHCLIENT6_HOSTNAME_OPTION
# variable provides a hostname.
# When DHCLIENT6_HOSTNAME_OPTION is set to AUTO, short hostname from the
# /etc/hostname file is send (same to SLES-11).
#
DHCLIENT6_FQDN_ENABLED=""

## Type:        list(both,ptr,none,)
## Default:     ""
#
# Request to update AAAA and PTR or only the PTR DNS records using the
# hostname specified in DHCLIENT6_HOSTNAME_OPTION variable.
# Default is to update \fIboth\fR when hostname is given or \fInone\fR
# when hostname is empty and DHCLIENT6_FQDN_ENABLED is set to enabled.
#
DHCLIENT6_FQDN_UPDATE=""

## Type:        yesno
```

```
## Default:     yes
#
# Qualify relative sub-domains/hostname in the DHCLIENT6_HOSTNAME_OPTION
# variable adding a final dot ('foo.bar' -> 'foo.bar.').
# When disabled, the DHCP server may append it's update domain to the
# hostname (e.g. 'foo.bar' -> 'foo.bar.example.net').
#
DHCLIENT6_FQDN_QUALIFY="yes"
## Path:    Network/DHCP/DHCP client
## Description: DHCPv4 client configuration variables
#
# Note:
# To configure one or more interfaces for DHCP configuration, you have to
# change the BOOTPROTO variable in /etc/sysconfig/network/ifcfg-<interface>
# to 'dhcp' (and possibly set STARTMODE='onboot').
#
# Most of the options can and should be overridden by per-interface
# settings in the ifcfg-* files.
#
# Note: NetworkManager is not using any sysconfig settings.
#

## Type:    yesno
## Default: no
#
# Should the DHCPv4 client set the hostname? (yes|no)
#
# When it is likely that this would occur during a running X session,
# your DISPLAY variable could be screwed up and you won't be able to open
# new windows anymore, then this should be "no".
#
# If it happens during booting it won't be a problem and you can
# safely say "yes" here. For a roaming notebook with X kept running, "no"
# makes more sense.
```

```
#
DHCLIENT_SET_HOSTNAME="yes"

## Type:    string
## Default: AUTO
#
# Specifies the hostname option field when DHCPv4 client sends messages.
# Some DHCP servers will update nameserver entries (dynamic DNS) to it.
# Also, some DHCP servers, notably those used by @Home Networks, require
# the hostname option field containing a specific string in the DHCP
# messages from clients.
#
# When set to "AUTO", the current hostname from /etc/hostname is sent.
# Use this variable to override it with another hostname, or leave it
# empty to not send any hostname.
#
DHCLIENT_HOSTNAME_OPTION="AUTO"

## Type:    yesno
## Default: yes
#
# Should the DHCP client set a default route (default Gateway) (yes|no)
#
# When multiple copies of dhcp client run, it would make sense that only
# one of them does it.
#
DHCLIENT_SET_DEFAULT_ROUTE="yes"

## Type:        integer
## Default:     "0"
#
# This option allows to set a metrics/priority for DHCPv4 routes.
#
DHCLIENT_ROUTE_PRIORITY="0"
```

```
## Type:    string
## Default: ""
#
# specify a client ID
#
# Specifies a client identifier string. By default an id derived from the
# hardware address of the network interface is sent as client identifier.
#
DHCLIENT_CLIENT_ID=""

## Type:    string
## Default: ""
#
# Specifies the vendor class identifier string. The default is dhcp client
# specific.
#
DHCLIENT_VENDOR_CLASS_ID=""

## Type:        list<rfc3004,string>
## Default:     string
#
# Specifies the format of the DHCLIENT_USER_CLASS_ID variable.
#
# The DHCPv4 option and it's format is specified by RFC3004 as an array
# of class identifiers, but most DHCP clients/servers aren't compliant
# with the specification and send/expect a single string without proper
# RFC3004 length-value tuple format instead.
#
# When set to "rfc3004" DHCLIENT_USER_CLASS_ID[SUFFIX] permit an RFC
# compliant array, otherwise DHCLIENT_USER_CLASS_ID is used as string.
#
DHCLIENT_USER_CLASS_FORMAT=""

## Type:        string
```

```
## Default:     ""
## Suffix:      yes
#
# Specifies the user class identifier (array) to send in dhcp requests.
# The DHCLIENT_USER_CLASS_FORMAT variable specified how to interpret it.
#
DHCLIENT_USER_CLASS_ID=""

## Type:    integer
## Default: ""
#
# Specifies the lease time (in seconds), that is suggested to the
# server. Default is to use the lease time offered by the server.
#
DHCLIENT_LEASE_TIME=""

## Type:        yesno
## Default:     yes
#
# This setting controls whether dhcp client should try to use DHCP settings
# provided in its last lease when the dhcp-server is not reachable and
# the lease hasn't expired yet.
# Set this variable to "no" to disable the fallback to the last lease.
#
DHCLIENT_USE_LAST_LEASE="yes"

## Type:    yesno
## Default: no
#
# Send a DHCPRELEASE to the server (sign off the address)? (yes|no)
# This may lead to getting a different address/hostname next time an address
# is requested. But some servers require it.
#
DHCLIENT_RELEASE_BEFORE_QUIT="no"
```

```
## Type:    integer
## Default: 0
#
# Some interfaces need time to initialize and/or do not report correct status.
# Add the latency time in seconds so these can be handled properly. Should
# probably set per interface rather than here.
# This setting causes a sleep time before dhcp clients are started regardless
# of the link status.
#
# Note: RFC 2131 specifies, that the dhcp client should wait a random time
# between one and ten seconds to desynchronize the use of DHCP at startup.
# We do not use this initial delay to not slow down start/boot time.
#
DHCLIENT_SLEEP="0"

## Type:    integer
## Default: 15
#
# The DHCPv4 client will try to get a lease for DHCLIENT_WAIT_AT_BOOT seconds,
# then inform ifup waiting for it, that it continues in background.
# When you increase this time, increase also the WAIT_FOR_INTERFACES variable
# e.g. to a value twice as high as the time specified here.
#
DHCLIENT_WAIT_AT_BOOT="15"

## Type:    integer
## Default: "0"
#
# The DHCPv4 client will stop processing / fail after this time when it does
# not get a reply from the dhcp server. Before you set this variable, take a
# look at DHCLIENT_WAIT_AT_BOOT allowing to continue in background instead.
#
DHCLIENT_TIMEOUT="0"
```

```
## Path:    Network/DHCP/DHCPv6 client
## Description: Global DHCPv6 client configuration

## Type:        list(auto,managed,info)
## Default:     auto
#
# This option allows to specify the request mode used by the DHCPv6
# client when the BOOTPROTO is set to dhcp or dhcp6, and overrides
# the "Managed Address Configuration" and the "Other Configuration"
# flags provided by the IPv6 router its Router Advertisement (RA)
# for the network connected to this interface.
#
#   auto:    follow RA flags, remain silent when no RA flag is set
#   info:    request other configuration (dns,ntp) only, no IP address
#   managed: request IP address as well as other configuration
#
DHCLIENT6_MODE="auto"

## Type:        yesno
### Default:     yes
#
# This option allows the DHCPv6 client to indicate its desire to accept
# rapid commit leases using two-packet exchange (solicitation, lease ack)
# instead of the four packet (solicitation, offer, request, lease ack).
#
DHCLIENT6_RAPID_COMMIT="yes"

## Type:    yesno
## Default: no
#
# Should the DHCPv6 client set the hostname? (yes|no)
#
# When it is likely that this would occur during a running X session,
# your DISPLAY variable could be screwed up and you won't be able to
```

```
# open new windows anymore, then this should be "no".
#
# If it happens during booting it won't be a problem and you can
# safely say "yes" here. For a roaming notebook with X kept running,
# "no" makes more sense.
#
DHCLIENT6_SET_HOSTNAME="no"

## Type:        string
### Default:     AUTO
#
# Specifies the hostname option field when DHCPv6 client sends messages.
# Some DHCP servers will update nameserver entries (dynamic DNS) to it.
#
# When set to "AUTO", the current hostname from /etc/hostname is sent.
# Use this variable to override it with another hostname, or leave it
# empty to not send any hostname.
#
DHCLIENT6_HOSTNAME_OPTION="AUTO"

## Type:        integer
### Default:     ""
#
# Specifies the preferred lifetime (in seconds) used as T1/renewal
# (1/2 of it) and T1/rebind (4/5 of it) in DHCPv6 IA NA requests.
#
# Default is to not propose anything but use the times as offered
# by the DHCPv6 server.
#
DHCLIENT6_LEASE_TIME=""

## Type:        yesno
## Default:     yes
#
```

```
# This setting controls whether DHCPv6 client should try to use settings
# provided in its last lease when the DHCPv6-server is not reachable and
# the lease hasn't expired yet.
# Set this variable to "no" to disable the fallback to the last lease.
#
DHCLIENT6_USE_LAST_LEASE="yes"

## Type:    yesno
## Default: no
#
# Send a DHCPv6 RELEASE to the server (sign off the address)? (yes|no)
# This may lead to getting a different address/hostname next time an address
# is requested. But some servers require it.
#
DHCLIENT6_RELEASE_BEFORE_QUIT="no"

## Type:    integer
## Default: 0
#
# Some interfaces need time to initialize and/or do not report correct status.
# By default, DHCPv6 waits until the link-local address (fe80::) is available
# and then ~1 second as specified by RFC3315.
# This setting allows override to use a non-standsrd initial delay.
#
DHCLIENT6_SLEEP="0"

## Type:    integer
## Default: 15
#
# The DHCPv6 client will try to get a lease for DHCLIENT6_WAIT_AT_BOOT seconds,
# then inform ifup waiting for it, that it continues in background.
# When you increase this time, increase also the WAIT_FOR_INTERFACES variable
# e.g. to a value twice as high as the time specified here.
#
```

```
DHCLIENT6_WAIT_AT_BOOT="15"

## Type:    integer
## Default: "0"
#
# The dhcpv6 client will stop processing / fail after this time when it does
# not get a reply from the dhcp server. Before you set this variable, take a
# look at DHCLIENT6_WAIT_AT_BOOT allowing to continue in background instead.
#
DHCLIENT6_TIMEOUT="0"
## Path:    Network/DHCP/DHCP client
## Description: DHCP client configuration
## Type:    yesno
## Default: yes
#
# Should the DHCP client modify /etc/samba/dhcp.conf?
#
DHCLIENT_MODIFY_SMB_CONF="yes"
WRITE_HOSTNAME_TO_HOSTS="no"
```

Les directives activées de ce fichier sont :

```
SLES12SP1:~ # egrep -v '^(#|$)' /etc/sysconfig/network/dhcp > /tmp/dhcp
SLES12SP1:~ # cat /tmp/dhcp
DHCLIENT_FQDN_ENABLED=""
DHCLIENT_FQDN_UPDATE=""
DHCLIENT_FQDN_QUALIFY="yes"
DHCLIENT_FQDN_ENCODE="yes"
DHCLIENT6_FQDN_ENABLED=""
DHCLIENT6_FQDN_UPDATE=""
DHCLIENT6_FQDN_QUALIFY="yes"
DHCLIENT_SET_HOSTNAME="yes"
DHCLIENT_HOSTNAME_OPTION="AUTO"
DHCLIENT_SET_DEFAULT_ROUTE="yes"
```

```
DHCLIENT_ROUTE_PRIORITY="0"
DHCLIENT_CLIENT_ID=""
DHCLIENT_VENDOR_CLASS_ID=""
DHCLIENT_USER_CLASS_FORMAT=""
DHCLIENT_USER_CLASS_ID=""
DHCLIENT_LEASE_TIME=""
DHCLIENT_USE_LAST_LEASE="yes"
DHCLIENT_RELEASE_BEFORE_QUIT="no"
DHCLIENT_SLEEP="0"
DHCLIENT_WAIT_AT_BOOT="15"
DHCLIENT_TIMEOUT="0"
DHCLIENT6_MODE="auto"
DHCLIENT6_RAPID_COMMIT="yes"
DHCLIENT6_SET_HOSTNAME="no"
DHCLIENT6_HOSTNAME_OPTION="AUTO"
DHCLIENT6_LEASE_TIME=""
DHCLIENT6_USE_LAST_LEASE="yes"
DHCLIENT6_RELEASE_BEFORE_QUIT="no"
DHCLIENT6_SLEEP="0"
DHCLIENT6_WAIT_AT_BOOT="15"
DHCLIENT6_TIMEOUT="0"
DHCLIENT_MODIFY_SMB_CONF="yes"
WRITE_HOSTNAME_TO_HOSTS="no"
```

Important - Les valeurs des directives ne sont ni utilisées par **NetworkManager**, ni utilisées par **dhclient**.

Les Fichiers /etc/sysconfig/network/ifcfg-ethX

Chaque interface réseau a son fichier de configuration propre. Dans le cas de l'interface **eth0**, le fichier est **/etc/sysconfig/networkifcfg-eth0** :

```
SLES12SP1:~ # cat /etc/sysconfig/network/ifcfg-eth0
BOOTPROTO='dhcp'
BROADCAST=''
ETHTOOL_OPTIONS=''
IPADDR=''
MTU=''
NAME=''
NETMASK=''
NETWORK=''
REMOTE_IPADDR=''
STARTMODE='auto'
DHCLIENT_SET_DEFAULT_ROUTE='yes'
```

Important - Toutes les variables se trouvant dans les fichiers **/etc/sysconfig/network/config** et **/etc/sysconfig/network/dhcp** peuvent être utilisées dans ce fichier au quel cas elles sont prioritaires par rapport à celles dans les deux fichiers de configuration générique.

Dans le cas de l'exemple ci dessus, l'interface est en mode **dhcp**. Pour configurer l'interface en IP fixe, modifiez le fichier ainsi :

```
SLES12SP1:~ # vi /etc/sysconfig/network/ifcfg-eth0
SLES12SP1:~ # cat /etc/sysconfig/network/ifcfg-eth0
BOOTPROTO='static'
BROADCAST=''
ETHTOOL_OPTIONS=''
IPADDR='10.0.2.15/24'
MTU='1500'
NAME='82540EM Gigabit Ethernet Controller'
NETMASK='255.255.255.0'
NETWORK=''
REMOTE_IPADDR=''
```

```
STARTMODE='auto'
USERCONTROL='yes'
```

Important - La configuration n'est pas encore complète car il faut configurer manuellement les serveurs DNS et la passerelle !

Pour modifier la valeur des adresses IP des serveurs DNS, il est nécessaire de modifier les fichiers **/etc/sysconfig/network/config** et **/etc/sysconfig/network/dhcp**.

Modifiez donc le fichier **/etc/sysconfig/network/config** ainsi :

```
...
# List of DNS domain names used for host-name lookup.
# It is written as search list into the /etc/resolv.conf file.
#
NETCONFIG_DNS_STATIC_SEARCHLIST="fenestros.loc"

## Type:        string
## Default:     ""
#
# List of DNS nameserver IP addresses to use for host-name lookup.
# When the NETCONFIG_DNS_FORWARDER variable is set to "resolver",
# the name servers are written directly to /etc/resolv.conf.
# Otherwise, the nameserver are written into a forwarder specific
# configuration file and the /etc/resolv.conf does not contain any
# nameservers causing the glibc to use the name server on the local
# machine (the forwarder). See also netconfig(8) manual page.
#
NETCONFIG_DNS_STATIC_SERVERS="8.8.8.8 8.8.4.4"
...
```

Puis modifiez le fichier **/etc/sysconfig/network/dhcp** ainsi :

```
...
# Should the DHCPv4 client set the hostname? (yes|no)
#
# When it is likely that this would occur during a running X session,
# your DISPLAY variable could be screwed up and you won't be able to open
# new windows anymore, then this should be "no".
#
# If it happens during booting it won't be a problem and you can
# safely say "yes" here. For a roaming notebook with X kept running, "no"
# makes more sense.
#
# DHCLIENT_SET_HOSTNAME="yes"
DHCLIENT_SET_HOSTNAME="no"
...
```

Redémarrez maintenant le service réseau afin que les modifications soient prises en compte :

```
SLES12SP1:~ # systemctl restart network.service
```

Vérifiez la mise en place des serveurs DNS :

```
SLES12SP1:~ # cat /etc/resolv.conf
### /etc/resolv.conf file autogenerated by netconfig!
#
# Before you change this file manually, consider to define the
# static DNS configuration using the following variables in the
# /etc/sysconfig/network/config file:
#     NETCONFIG_DNS_STATIC_SEARCHLIST
#     NETCONFIG_DNS_STATIC_SERVERS
#     NETCONFIG_DNS_FORWARDER
# or disable DNS configuration updates via netconfig by setting:
#     NETCONFIG_DNS_POLICY=''
```

```
#
# See also the netconfig(8) manual page and other documentation.
#
# Note: Manual change of this file disables netconfig too, but
# may get lost when this file contains comments or empty lines
# only, the netconfig settings are same with settings in this
# file and in case of a "netconfig update -f" call.
#
### Please remove (at least) this line when you modify the file!
search fenestros.loc
nameserver 8.8.8.8
nameserver 8.8.4.4
```

Notez qu'à ce stade, il n'existe toujours pas de résolution de noms :

```
SLES12SP1:~ # ping www.free.fr
ping: unknown host www.free.fr
```

La raison est le manque d'une route par défault dans la table de routage statique du noyau :

```
SLES12SP1:~ # route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
10.0.2.0        *               255.255.255.0   U     0      0        0 eth0
```

Tester la Configuration d'IP Fixe

Utilisez maintenant la commande **wicked** et la sous-commande **ifstatus** pour vérifier la configuration d'eth0 :

```
SLES12SP1:~ # wicked ifstatus eth0
eth0            up
      link:     #2, state up, mtu 1500
```

```
      type:     ethernet, hwaddr 08:00:27:7d:da:95
      config:   compat:suse:/etc/sysconfig/network/ifcfg-eth0
      leases:   ipv4 static granted
      addr:     ipv4 10.0.2.15/24 [static]
```

VirtualBox fournit une passerelle par défaut à l'adresse IP 10.0.2.2. Insérez donc cette information dans la table de routage du noyau avec la commande suivante :

```
SLES12SP1:~ # route add default gw 10.0.2.2 eth0
SLES12SP1:~ # route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
default         10.0.2.2        0.0.0.0         UG    0      0        0 eth0
10.0.2.0        *               255.255.255.0   U     0      0        0 eth0
```

Pour rendre cette route persistante, créez le fichier **/etc/sysconfig/network/routes** avec comme contenu **default 10.0.2.2** :

```
SLES12SP1:~ # vi /etc/sysconfig/network/routes
SLES12SP1:~ # cat /etc/sysconfig/network/routes
default 10.0.2.2
```

Utilisez de nouveau la commande **wicked ifstatus** pour vérifier la configuration d'eth0 :

```
SLES12SP1:~ # wicked ifstatus eth0
eth0            up
      link:     #2, state up, mtu 1500
      type:     ethernet, hwaddr 08:00:27:7d:da:95
      config:   compat:suse:/etc/sysconfig/network/ifcfg-eth0
      leases:   ipv4 static granted
      addr:     ipv4 10.0.2.15/24 [static]
      route:    ipv4 default via 10.0.2.2 proto boot
```

Important - Notez que la passerelle par défaut a été renseignée.

La Commande hostname

Afin d'informer le système immédiatement de la modification du FQDN (*Fully Qualified Domain Name*), utilisez la commande **hostname** :

```
SLES12SP1:~ # hostname
SLES12SP1.fenestros.loc
SLES12SP1:~ # hostname sles12sp1.fenestros.loc
SLES12SP1:~ # hostname
sles12sp1.fenestros.loc
SLES12SP1:~ # cat /etc/HOSTNAME
SUSE12SP1.fenestros.loc
SLES12SP1:~ # hostname SLES12SP1.fenestros.loc
SLES12SP1:~ # hostname
SLES12SP1.fenestros.loc
```

Pour afficher le FQDN du système vous pouvez également utiliser la commande suivante :

```
SLES12SP1:~ # uname -n
sles12sp1.fenestros.loc
```

Options de la commande hostname

Les options de cette commande sont :

```
SLES12SP1:~ # hostname --help
Usage: hostname [-v] {hostname|-F file}      set hostname (from file)
       domainname [-v] {nisdomain|-F file}   set NIS domainname (from file)
       hostname [-v] [-d|-f|-s|-a|-i|-y|-n]  display formatted name
```

```
       hostname [-v]                           display hostname

       hostname -V|--version|-h|--help         print info and exit

    dnsdomainname=hostname -d, {yp,nis,}domainname=hostname -y

    -s, --short           short host name
    -a, --alias           alias names
    -i, --ip-address      addresses for the hostname
    -f, --fqdn, --long    long host name (FQDN)
    -d, --domain          DNS domain name
    -y, --yp, --nis       NIS/YP domainname
    -F, --file            read hostname or NIS domainname from given file

   This command can read or set the hostname or the NIS domainname. You can
   also read the DNS domain or the FQDN (fully qualified domain name).
   Unless you are using bind or NIS for host lookups you can change the
   FQDN (Fully Qualified Domain Name) and the DNS domain name (which is
   part of the FQDN) in the /etc/hosts file.
```

La Commande ifconfig

Pour afficher la configuration IP de la machine il faut saisir la commande suivante :

```
SLES12SP1:~ # ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:7D:DA:95
          inet addr:10.0.2.15  Bcast:10.0.2.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe7d:da95/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:7741 errors:0 dropped:0 overruns:0 frame:0
          TX packets:6419 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:525771 (513.4 Kb)  TX bytes:6124180 (5.8 Mb)
```

```
lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  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 txqueuelen:0
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
```

La commande ifconfig est également utilisée pour configurer une interface.

Créez maintenant une interface fictive ainsi :

```
SLES12SP1:~ # ifconfig eth0:1 192.168.1.2 netmask 255.255.255.0 broadcast 192.168.1.255
```

Constatez maintenant le résultat :

```
SLES12SP1:~ # ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:7D:DA:95
          inet addr:10.0.2.15  Bcast:10.0.2.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe7d:da95/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:7750 errors:0 dropped:0 overruns:0 frame:0
          TX packets:6424 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:526511 (514.1 Kb)  TX bytes:6125670 (5.8 Mb)

eth0:1    Link encap:Ethernet  HWaddr 08:00:27:7D:DA:95
          inet addr:192.168.1.2  Bcast:192.168.1.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
```

```
          UP LOOPBACK RUNNING  MTU:65536  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 txqueuelen:0
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
```

Options de la commande ifconfig

Les options de cette commande sont :

```
SLES12SP1:~ # ifconfig --help
Usage:
  ifconfig [-a] [-i] [-v] [-s] <interface> [[<AF>] <address>]
  [add <address>[/<prefixlen>]]
  [del <address>[/<prefixlen>]]
  [[-]broadcast [<address>]]  [[-]pointopoint [<address>]]
  [netmask <address>]  [dstaddr <address>]  [tunnel <address>]
  [outfill <NN>] [keepalive <NN>]
  [hw <HW> <address>]  [metric <NN>]  [mtu <NN>]
  [[-]trailers]  [[-]arp]  [[-]allmulti]
  [multicast]  [[-]promisc]
  [mem_start <NN>]  [io_addr <NN>]  [irq <NN>]  [media <type>]
  [txqueuelen <NN>]
  [[-]dynamic]
  [up|down] ...

  <HW>=Hardware Type.
  List of possible hardware types:
    loop (Local Loopback) slip (Serial Line IP) cslip (VJ Serial Line IP)
    slip6 (6-bit Serial Line IP) cslip6 (VJ 6-bit Serial Line IP) adaptive (Adaptive Serial Line IP)
    ether (Ethernet) tr (16/4 Mbps Token Ring) tr (16/4 Mbps Token Ring (New))
    ax25 (AMPR AX.25) netrom (AMPR NET/ROM) tunnel (IPIP Tunnel)
    ppp (Point-to-Point Protocol) arcnet (ARCnet) dlci (Frame Relay DLCI)
```

```
    frad (Frame Relay Access Device) sit (IPv6-in-IPv4) fddi (Fiber Distributed Data Interface)
    hippi (HIPPI) irda (IrLAP) x25 (generic X.25)
    infiniband (InfiniBand)
  <AF>=Address family. Default: inet
  List of possible address families:
    unix (UNIX Domain) inet (DARPA Internet) inet6 (IPv6)
    ax25 (AMPR AX.25) netrom (AMPR NET/ROM) ipx (Novell IPX)
    ddp (Appletalk DDP) x25 (CCITT X.25)
```

La Commande ifstatus

Rappelez-vous que cette commande fournit des informations concernant l'interface réseau passée en argument :

```
SLES12SP1:~ # ifstatus eth0
eth0            up
      link:     #2, state up, mtu 1500
      type:     ethernet, hwaddr 08:00:27:7d:da:95
      config:   compat:suse:/etc/sysconfig/network/ifcfg-eth0
      leases:   ipv4 static granted
      addr:     ipv4 10.0.2.15/24 [static]
      addr:     ipv4 192.168.1.2/24
      route:    ipv4 default via 10.0.2.2 [static]
```

Options de la commande ifstatus

Les options de cette commande sont :

```
SLES12SP1:~ # ifstatus --help
Usage: if{up,down,status,probe} [<config>] <interface> [-o <options>]

Common options are:
```

```
    debug    : be verbose

Options for if{status,probe} are:
    quiet    : Do not print out errors, but just signal the result through exit status

Options for ifdown are:
    force    : Forces persistent (nfsroot type) interface down

Unknown options are simply ignored, so be careful.
```

Le Fichier/etc/HOSTNAME

Ce fichier contient le nom d'hôte de la machine :

```
SLES12SP1:~ # cat /etc/HOSTNAME
SUSE12SP1.fenestros.loc
```

Le Fichier /etc/networks

Ce fichier contient la correspondance entre des noms de réseaux et l'adresse IP du réseau, utilisée notamment lors du démarrage du système en l'absence de serveurs DNS :

```
SLES12SP1:~ # cat /etc/networks
#
# networks  This file describes a number of netname-to-address
#       mappings for the TCP/IP subsystem.  It is mostly
#       used at boot time, when no name servers are running.
#

loopback    127.0.0.0
link-local  169.254.0.0
```

```
# End.
```

Le Fichier /etc/nsswitch.conf

L'ordre de recherche des services de noms est stocké dans le fichier **/etc/nsswitch.conf**. Pour connaître l'ordre, saisissez la commande suivante :

```
SLES12SP1:~ # grep '^hosts:' /etc/nsswitch.conf
hosts:      files dns
```

Le Fichier /etc/hosts

Le mot **files** dans la sortie de la commande précédente fait référence au fichier **/etc/hosts** :

```
SLES12SP1:~ # cat /etc/hosts
#
# hosts         This file describes a number of hostname-to-address
#               mappings for the TCP/IP subsystem.  It is mostly
#               used at boot time, when no name servers are running.
#               On small systems, this file can be used instead of a
#               "named" name server.
# Syntax:
#
# IP-Address  Full-Qualified-Hostname  Short-Hostname
#

127.0.0.1   localhost

# special IPv6 addresses
::1             localhost ipv6-localhost ipv6-loopback

fe00::0         ipv6-localnet
```

```
ff00::0         ipv6-mcastprefix
ff02::1         ipv6-allnodes
ff02::2         ipv6-allrouters
ff02::3         ipv6-allhosts
```

Editez ce fichier en ajoutant une ligne pour l'adresse IP de votre VM :

```
SLES12SP1:~ # vi /etc/hosts
SLES12SP1:~ # cat /etc/hosts
#
# hosts         This file describes a number of hostname-to-address
#               mappings for the TCP/IP subsystem.  It is mostly
#               used at boot time, when no name servers are running.
#               On small systems, this file can be used instead of a
#               "named" name server.
# Syntax:
#
# IP-Address  Full-Qualified-Hostname  Short-Hostname
#

127.0.0.1   localhost
10.0.2.15   SUSE12SP1.fenestros.loc

# special IPv6 addresses
::1             localhost ipv6-localhost ipv6-loopback

fe00::0         ipv6-localnet

ff00::0         ipv6-mcastprefix
ff02::1         ipv6-allnodes
ff02::2         ipv6-allrouters
ff02::3         ipv6-allhosts
```

Le Fichier /etc/resolv.conf

Ce fichier contient une liste de serveurs DNS.

```
SLES12SP1:~ # cat /etc/resolv.conf
### /etc/resolv.conf file autogenerated by netconfig!
#
# Before you change this file manually, consider to define the
# static DNS configuration using the following variables in the
# /etc/sysconfig/network/config file:
#     NETCONFIG_DNS_STATIC_SEARCHLIST
#     NETCONFIG_DNS_STATIC_SERVERS
#     NETCONFIG_DNS_FORWARDER
# or disable DNS configuration updates via netconfig by setting:
#     NETCONFIG_DNS_POLICY=''
#
# See also the netconfig(8) manual page and other documentation.
#
# Note: Manual change of this file disables netconfig too, but
# may get lost when this file contains comments or empty lines
# only, the netconfig settings are same with settings in this
# file and in case of a "netconfig update -f" call.
#
### Please remove (at least) this line when you modify the file!
nameserver 8.8.8.8
nameserver 8.8.4.4
```

Tester la Configuration de la Résolution des Noms

Pour tester la configuration de la résolution des noms, deux commandes sont possibles, **nslookup** et **dig** :

```
SLES12SP1:~ # nslookup www.ittraining.center
```

```
Server:     8.8.8.8
Address:    8.8.8.8#53

Non-authoritative answer:
Name:   www.ittraining.center
Address: 217.160.0.225

SLES12SP1:~ # dig www.ittraining.center

; <<>> DiG 9.9.9-P1 <<>> www.ittraining.center
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 53925
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 512
;; QUESTION SECTION:
;www.ittraining.center.     IN    A

;; ANSWER SECTION:
www.ittraining.center.  3599    IN    A    217.160.0.225

;; Query time: 73 msec
;; SERVER: 8.8.8.8#53(8.8.8.8)
;; WHEN: Fri Oct 27 17:28:50 CEST 2017
;; MSG SIZE  rcvd: 66
```

Services réseaux

Quand un client émet une demande de connexion vers une application réseau sur un serveur, il utilise un socket attaché à un port local **supérieur à 1023**, alloué d'une manière dynamique. La requête contient le port de destination sur le serveur. Certaines applications serveurs se gèrent toutes

seules, ce qui est la cas par exemple d'**httpd**. Par contre d'autres sont gérées par le service **xinetd**.

xinetd

Sous SUSE 12 xinetd est installé par défaut :

Le programme xinetd est configuré via le fichier **/etc/xinetd.conf** :

```
SLES12SP1:~ # cat /etc/xinetd.conf
#
# This is the master xinetd configuration file. Settings in the
# default section will be inherited by all service configurations
# unless explicitly overridden in the service configuration. See
# xinetd.conf in the man pages for a more detailed explanation of
# these attributes.

defaults
{
# The next two items are intended to be a quick access place to
# temporarily enable or disable services.
#
#   enabled        =
#   disabled    =

# Previous default in SUSE - please don't forget to use the logrotate. The
# sample configuration is in /usr/share/packages/doc/xinetd/logrotate
#       log_type        = FILE /var/log/xinetd.log

# Define general logging characteristics.
    log_type    = SYSLOG daemon info
    log_on_failure    = HOST ATTEMPT
    log_on_success    = HOST EXIT DURATION
```

```
# Define access restriction defaults
#
#   no_access    =
#   only_from    = localhost
#   max_load    = 0
    cps        = 50 10
    instances    = 30
    per_source    = 10

#
# The specification of an interface is interesting, if we are on a firewall.
# For example, if you only want to provide services from an internal
# network interface, you may specify your internal interfaces IP-Address.
#
#   bind    = 127.0.0.1

# Address and networking defaults
#
#   bind        =
#   mdns        = yes
    v6only        = no

# setup environmental attributes
#
#   passenv        =
    groups        = yes
    umask        = 002

# Generally, banners are not used. This sets up their global defaults
#
#   banner        =
#   banner_fail    =
#   banner_success    =
}
```

```
includedir /etc/xinetd.d
```

Ce fichier ne définit pas les applications serveurs directement. Il indique plutôt le répertoire qui contient les fichiers de définitions des applications serveurs qui est **/etc/xinetd.d** :

```
SLES12SP1:~ # ls -l /etc/xinetd.d
total 72
-rw-r--r-- 1 root root  313 Sep 23  2014 chargen
-rw-r--r-- 1 root root  333 Sep 23  2014 chargen-udp
-rw-r--r-- 1 root root  256 Jun  7  2016 cups-lpd
-rw-r--r-- 1 root root  313 Sep 23  2014 daytime
-rw-r--r-- 1 root root  333 Sep 23  2014 daytime-udp
-rw-r--r-- 1 root root  313 Sep 23  2014 discard
-rw-r--r-- 1 root root  332 Sep 23  2014 discard-udp
-rw-r--r-- 1 root root  305 Sep 23  2014 echo
-rw-r--r-- 1 root root  324 Sep 23  2014 echo-udp
-rw-r--r-- 1 root root  492 Sep 21  2014 netstat
-rw-r--r-- 1 root root  207 Sep 23  2016 rsync
-rw-r--r-- 1 root root  332 Sep 23  2014 servers
-rw-r--r-- 1 root root  334 Sep 23  2014 services
-rw-r--r-- 1 root root  536 May  3 21:45 systat
-rw-r--r-- 1 root root  339 Sep 23  2014 time
-rw-r--r-- 1 root root  333 Sep 23  2014 time-udp
-rw-r--r-- 1 root root 2685 Oct 21 12:13 vnc
-rw------- 1 root root  916 Apr  7  2017 vsftpd
```

A l'examen de ce répertoire vous noterez que celui-ci contient des fichiers nominatifs par application-serveur, par exemple pour le serveur rsync :

```
SLES12SP1:~ # cat /etc/xinetd.d/vsftpd
# default: off
# description:
#   The vsftpd FTP server serves FTP connections. It uses
#   normal, unencrypted usernames and passwords for authentication.
# vsftpd is designed to be secure.
```

```
#
# NOTE: This file contains the configuration for xinetd to start vsftpd.
#       the configuration file for vsftp itself is in /etc/vsftpd.conf
#
# NOTE: Remember to set both listen and listen_ipv6 to NO in /etc/vsftpd.conf
#       in order to have working xinetd connection.
#
service ftp
{
        socket_type             = stream
        protocol                = tcp
        wait                    = no
        user                    = root
        server                  = /usr/sbin/vsftpd
        server_args             = /etc/vsftpd.conf
#        log_on_success          += DURATION USERID
#        log_on_failure          += USERID
#        nice                    = 10
        disable                 = yes
}
```

Les directives principales de ce fichier sont :

Paramètre	Déscription
disable	**no** : Le service est actif. **yes** : Le service est désactivé
port	Le numéro de port ou, à défaut, le numéro indiqué pour le service dans le fichier /etc/services
socket_type	Nature du socket, soit **stream** pour TCP soit **dgram** pour UDP
protocol	Protocole utilisé soit **TCP** soit **UDP**
wait	**no** : indique si xinetd active un serveur par client. **yes** : indique que xinetd active un seul serveur pour tous les client
user	Indique le compte sous lequel le serveur est exécuté
server	Indique le chemin d'accès de l'application serveur
env	Définit un environnement système
server_args	Donne les arguments transmis à l'application serveur

Afin d'activer une application serveur, il suffit de modifier le paramètre **disable** dans le fichier concerné et de relancer le service xinetd.

TCP Wrapper

TCP Wrapper contrôle l'accès à des services réseaux grâce à des **ACL**.

Quand une requête arrive pour un serveur, xinetd active le wrapper **tcpd** au lieu d'activer le serveur directement.

tcpd met à jour un journal et vérifie si le client a le droit d'utiliser le service concerné. Les ACL se trouvent dans deux fichiers:

- **/etc/hosts.allow**
- **/etc/hosts.deny**

Il faut noter que si ces fichiers n'existent pas ou sont vides, il n'y a pas de contrôle d'accès.

Le format d'une ligne dans un de ces deux fichiers est:

```
démon : liste_de_clients
```

Par exemple dans le cas d'un serveur **démon**, on verrait une ligne dans le fichier **/etc/hosts.allow** similaire à:

```
démon : LOCAL, .fenestros.loc
```

ce qui implique que les machines dont le nom ne comporte pas de point ainsi que les machines du domaine **fenestros.loc** sont autorisées à utiliser le service.

Le mot clef **ALL** peut être utilisé pour indiquer tout. Par exemple, **ALL:ALL** dans le fichier **/etc/host.deny** bloque effectivement toute tentative de connexion à un service xinetd sauf pour les ACL inclus dans le fichier /etc/host.allow.

Routage Statique

La commande route

Pour afficher la table de routage de la machine vous pouvez utiliser la commande **route** :

```
SLES12SP1:~ # route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
default         10.0.2.2        0.0.0.0         UG    0      0        0 eth0
10.0.2.0        *               255.255.255.0   U     0      0        0 eth0
192.168.1.0     *               255.255.255.0   U     0      0        0 eth0
```

La table issue de la commande **route** indique les informations suivantes:

- La destination qui peut être un hôte ou un réseau et est identifiée par les champs **Destination** et **Genmask**
- La route à prendre identifiée par les champs **Gateway** et **Iface**. La valeur d'Iface spécifie la carte à utiliser,
- Le champ **Indic** qui peux prendre un ou plusieurs de svaleurs suivantes:
  - U - **Up** - la route est active
  - H - **Host** - la route conduit à un hôte
  - G - **Gateways** - la route passe par une passerelle
- Le champ **Metric** indique le nombre de sauts (passerelles) pour atteindre la destination,
- Le champ **Ref** indique le nombre de références à cette route. Ce champs est usilisé par le Noyau de Linux,
- Le champ **Use** indique le nombre de recherches associés à cette route.

La commande **route** permet aussi de paramétrer le routage indirect. Par exemple pour supprimer la route vers le réseau 192.168.1.0 :

```
SLES12SP1:~ # route del -net 192.168.1.0 netmask 255.255.255.0
SLES12SP1:~ # route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
default         10.0.2.2        0.0.0.0         UG    0      0        0 eth0
10.0.2.0        *               255.255.255.0   U     0      0        0 eth0
```

Pour ajouter la route vers le réseau 192.168.1.0 :

```
SLES12SP1:~ # route add -net 192.168.1.0 netmask 255.255.255.0 gw 192.168.1.2
SLES12SP1:~ # route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
default         10.0.2.2        0.0.0.0         UG    0      0        0 eth0
10.0.2.0        *               255.255.255.0   U     0      0        0 eth0
192.168.1.0     192.168.1.2     255.255.255.0   UG    0      0        0 eth0
```

Important - La commande utilisée pour ajouter une passerelle par défaut prend la forme suivante **route add default gw *numéro_ip interface***.

Options de la commande route

Les options cette commande sont :

```
SLES12SP1:~ # route --help
Usage: route [-nNvee] [-FC] [<AF>]           List kernel routing tables
       route [-v] [-FC] {add|del|flush} ...  Modify routing table for AF.

       route {-h|--help} [<AF>]              Detailed usage syntax for specified AF.
       route {-V|--version}                  Display version/author and exit.

        -v, --verbose            be verbose
        -n, --numeric            don't resolve names
        -e, --extend             display other/more information
        -F, --fib                display Forwarding Information Base (default)
        -C, --cache              display routing cache instead of FIB

  <AF>=Use '-A <af>' or '--<af>'; default: inet
```

```
  List of possible address families (which support routing):
    inet (DARPA Internet) inet6 (IPv6) ax25 (AMPR AX.25)
    netrom (AMPR NET/ROM) ipx (Novell IPX) ddp (Appletalk DDP)
    x25 (CCITT X.25)
```

Vous pouvez aussi utiliser la commande **netstat** pour afficher la table de routage de la machine :

```
SLES12SP1:~ # netstat -nr
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
0.0.0.0         10.0.2.2        0.0.0.0         UG        0 0          0 eth0
10.0.2.0        0.0.0.0         255.255.255.0   U         0 0          0 eth0
192.168.1.0     0.0.0.0         255.255.255.0   U         0 0          0 eth0
```

La table issue de la commande **netstat -nr** indique les informations suivantes:

- La champ **MSS** indique la taille maximale des segments TCP sur la route,
- Le champ **Window** indique la taille de la fenêtre sur cette route,
- Le champ **irrt** indique le paramètre IRRT pour la route.

Activer/désactiver le routage sur le serveur

Pour activer le routage sur le serveur, il convient d'activer la retransmission des paquets:

```
SLES12SP1:~ # echo 1 > /proc/sys/net/ipv4/ip_forward
SLES12SP1:~ # cat /proc/sys/net/ipv4/ip_forward
1
```

Pour désactiver le routage sur le serveur, il convient de désactiver la retransmission des paquets:

```
SLES12SP1:~ # echo 0 > /proc/sys/net/ipv4/ip_forward
SLES12SP1:~ # cat /proc/sys/net/ipv4/ip_forward
```

```
0
```

Activer/Désactiver une Interface Manuellement

Deux commandes existent pour activer et désactiver manuellement une interface réseau :

```
SLES12SP1:~ # ifdown eth0
eth0        device-ready
SLES12SP1:~ # ifconfig
lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  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 txqueuelen:0
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)

SLES12SP1:~ # ifup eth0
eth0        up
SLES12SP1:~ # ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:7D:DA:95
          inet addr:10.0.2.15  Bcast:10.0.2.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe7d:da95/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:8072 errors:0 dropped:0 overruns:0 frame:0
          TX packets:6735 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:550591 (537.6 Kb)  TX bytes:6168958 (5.8 Mb)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
```

```
          UP LOOPBACK RUNNING  MTU:65536  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 txqueuelen:0
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
```

Diagnostique du Réseau

ping

Pour tester l'accessibilité d'une machine, vous devez utiliser la commande **ping** :

```
SLES12SP1:~ # ping 10.0.2.2
PING 10.0.2.2 (10.0.2.2) 56(84) bytes of data.
64 bytes from 10.0.2.2: icmp_seq=1 ttl=64 time=0.794 ms
64 bytes from 10.0.2.2: icmp_seq=2 ttl=64 time=0.857 ms
64 bytes from 10.0.2.2: icmp_seq=3 ttl=64 time=1.06 ms
64 bytes from 10.0.2.2: icmp_seq=4 ttl=64 time=0.285 ms
^C
--- 10.0.2.2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3030ms
rtt min/avg/max/mdev = 0.285/0.749/1.062/0.287 ms
```

Options de la commande ping

Les options de cette commande sont :

```
SLES12SP1:~ # ping --help
ping: invalid option -- '-'
Usage: ping [-aAbBdDfhLnOqrRUvV] [-c count] [-i interval] [-I interface]
```

```
            [-m mark] [-M pmtudisc_option] [-l preload] [-p pattern] [-Q tos]
            [-s packetsize] [-S sndbuf] [-t ttl] [-T timestamp_option]
            [-w deadline] [-W timeout] [hop1 ...] destination
```

netstat -i

Pour visualiser les statistiques réseaux, vous disposez de la commande **netstat** :

```
SLES12SP1:~ # netstat -i
Kernel Interface table
Iface   MTU Met    RX-OK RX-ERR RX-DRP RX-OVR    TX-OK TX-ERR TX-DRP TX-OVR Flg
eth0    1500   0     7834      0      0      0     6485      0      0      0 BMRU
eth0:1  1500   0     - no statistics available -                        BMRU
lo     65536   0        0      0      0      0        0      0      0      0 LRU
```

Options de la commande netstat

Les options de cette commande sont :

```
SLES12SP1:~ # netstat --help
usage: netstat [-veenNcCF] [<Af>] -r         netstat {-V|--version|-h|--help}
       netstat [-vnNcaeol] [<Socket> ...]
       netstat { [-veenNac] -i | [-cnNe] -M | -s }

        -r, --route              display routing table
        -i, --interfaces         display interface table
        -g, --groups             display multicast group memberships
        -s, --statistics         display networking statistics (like SNMP)
        -M, --masquerade         display masqueraded connections

        -v, --verbose            be verbose
```

```
       -n, --numeric            don't resolve names
       --numeric-hosts          don't resolve host names
       --numeric-ports          don't resolve port names
       --numeric-users          don't resolve user names
       -N, --symbolic           resolve hardware names
       -e, --extend             display other/more information
       -p, --programs           display PID/Program name for sockets
       -c, --continuous         continuous listing

       -l, --listening          display listening server sockets
       -a, --all, --listening   display all sockets (default: connected)
       -o, --timers             display timers
       -F, --fib                display Forwarding Information Base (default)
       -C, --cache              display routing cache instead of FIB

       -T, --notrim             dont't trim address information
 <Socket>={-t|--tcp} {-u|--udp} {-w|--raw} {-x|--unix} --ax25 --ipx --netrom --sctp
 <AF>=Use '-A <af>' or '--<af>'; default: inet
 List of possible address families (which support routing):
   inet (DARPA Internet) inet6 (IPv6) ax25 (AMPR AX.25)
   netrom (AMPR NET/ROM) ipx (Novell IPX) ddp (Appletalk DDP)
   x25 (CCITT X.25)
```

La commande traceroute

La commande ping est à la base de la commande **traceroute**. Cette commande sert à découvrir la route empruntée pour accéder à un site donné :

```
SLES12SP1:~ # traceroute ittraining.center
traceroute to ittraining.center (217.160.0.225), 30 hops max, 60 byte packets
 1  10.0.2.2 (10.0.2.2)  0.341 ms  0.277 ms  0.393 ms
 2  192.168.1.1 (192.168.1.1)  5.704 ms  6.554 ms  7.139 ms
 3  * * *
 4  10.4.2.5 (10.4.2.5)  45.439 ms 10.4.0.5 (10.4.0.5)  57.022 ms 10.4.1.5 (10.4.1.5)  57.520 ms
```

```
 5  10.187.154.251 (10.187.154.251)  59.007 ms  58.326 ms  61.579 ms
 6  1.134.136.77.rev.sfr.net (77.136.134.1)  57.374 ms  44.892 ms  44.671 ms
 7  34.10.136.77.rev.sfr.net (77.136.10.34)  45.004 ms  64.241 ms  51.020 ms
 8  109.132.17.93.rev.sfr.net (93.17.132.109)  47.404 ms  47.281 ms  63.445 ms
 9  linx.bb-d.ba.slo.gb.oneandone.net (195.66.236.98)  63.255 ms  63.649 ms  63.896 ms
10  ae-1.bb-c.thn.lon.gb.oneandone.net (212.227.120.106)  64.907 ms  63.970 ms  65.664 ms
11  ae-6.bb-c.bap.rhr.de.oneandone.net (212.227.120.49)  76.872 ms  90.336 ms  70.017 ms
12  * * *
13  * * *
14  * * *
15  * * *
16  * * *
17  * * *
18  * * *
19  * * *
20  * * *
21  * * *
22  * * *
23  * * *
24  * * *
25  * * *
26  * * *
27  * * *
28  * * *
29  * * *
30  * * *
```

Options de la commande traceroute

Les options de cette commande sont :

```
SLES12SP1:~ # traceroute --help
Usage:
```

```
  traceroute [ -46dFITnreAUDV ] [ -f first_ttl ] [ -g gate,... ] [ -i device ] [ -m max_ttl ] [ -N squeries ] [ -
p port ] [ -t tos ] [ -l flow_label ] [ -w waittime ] [ -q nqueries ] [ -s src_addr ] [ -z sendwait ] [ --
fwmark=num ] host [ packetlen ]
Options:
  -4                          Use IPv4
  -6                          Use IPv6
  -d  --debug                 Enable socket level debugging
  -F  --dont-fragment         Do not fragment packets
  -f first_ttl  --first=first_ttl
                              Start from the first_ttl hop (instead from 1)
  -g gate,...  --gateway=gate,...
                              Route packets through the specified gateway
                              (maximum 8 for IPv4 and 127 for IPv6)
  -I  --icmp                  Use ICMP ECHO for tracerouting
  -T  --tcp                   Use TCP SYN for tracerouting (default port is 80)
  -i device  --interface=device
                              Specify a network interface to operate with
  -m max_ttl  --max-hops=max_ttl
                              Set the max number of hops (max TTL to be
                              reached). Default is 30
  -N squeries  --sim-queries=squeries
                              Set the number of probes to be tried
                              simultaneously (default is 16)
  -n                          Do not resolve IP addresses to their domain names
  -p port  --port=port        Set the destination port to use. It is either
                              initial udp port value for "default" method
                              (incremented by each probe, default is 33434), or
                              initial seq for "icmp" (incremented as well,
                              default from 1), or some constant destination
                              port for other methods (with default of 80 for
                              "tcp", 53 for "udp", etc.)
  -t tos  --tos=tos           Set the TOS (IPv4 type of service) or TC (IPv6
                              traffic class) value for outgoing packets
  -l flow_label  --flowlabel=flow_label
```

```
                              Use specified flow_label for IPv6 packets
-w waittime  --wait=waittime
                              Set the number of seconds to wait for response to
                              a probe (default is 5.0). Non-integer (float
                              point) values allowed too
-q nqueries  --queries=nqueries
                              Set the number of probes per each hop. Default is
                              3
-r                            Bypass the normal routing and send directly to a
                              host on an attached network
-s src_addr  --source=src_addr
                              Use source src_addr for outgoing packets
-z sendwait  --sendwait=sendwait
                              Minimal time interval between probes (default 0).
                              If the value is more than 10, then it specifies a
                              number in milliseconds, else it is a number of
                              seconds (float point values allowed too)
-e  --extensions              Show ICMP extensions (if present), including MPLS
-A  --as-path-lookups         Perform AS path lookups in routing registries and
                              print results directly after the corresponding
                              addresses
-M name  --module=name        Use specified module (either builtin or external)
                              for traceroute operations. Most methods have
                              their shortcuts (`-I' means `-M icmp' etc.)
-O OPTS,...  --options=OPTS,...
                              Use module-specific option OPTS for the
                              traceroute module. Several OPTS allowed,
                              separated by comma. If OPTS is "help", print info
                              about available options
--sport=num                   Use source port num for outgoing packets. Implies
                              `-N 1'
--fwmark=num                  Set firewall mark for outgoing packets
-U  --udp                     Use UDP to particular port for tracerouting
                              (instead of increasing the port per each probe),
```

```
                              default port is 53
  -UL                         Use UDPLITE for tracerouting (default dest port
                              is 53)
  -D  --dccp                  Use DCCP Request for tracerouting (default port
                              is 33434)
  -P prot  --protocol=prot    Use raw packet of protocol prot for tracerouting
  --mtu                       Discover MTU along the path being traced. Implies
                              `-F -N 1'
  --back                      Guess the number of hops in the backward path and
                              print if it differs
  -V  --version               Print version info and exit
  --help                      Read this help and exit

Arguments:
+     host          The host to traceroute to
      packetlen     The full packet length (default is the length of an IP
                    header plus 40). Can be ignored or increased to a minimal
                    allowed value
```

Connexions à Distance

Telnet

La commande **telnet** est utilisée pour établir une connexion à distance avec un serveur telnet :

```
  # telnet numero_ip
```

Important - Le service telnet revient à une redirection des canaux standards d'entrée et de sortie. Notez que la connexion n'est **pas** sécurisée. Pour fermer la connexion, il faut saisir la commande **exit**. La commande telnet n'offre pas de services de transfert de

fichiers. Pour cela, il convient d'utiliser la command **ftp**.

Options de la commande telnet

Les options de cette commande sont :

```
SLES12SP1:~ # telnet --help
telnet: invalid option -- '-'
Usage: telnet [-8] [-E] [-L] [-S tos] [-a] [-c] [-d] [-e char] [-l user]
    [-n tracefile] [-b hostalias ] [-r]
    [host-name [port]]
```

ftp

La commande **ftp** est utilisée pour le transfert de fichiers.

Le client ftp n'est pas installé par défault :

```
SLES12SP1:~ # ftp ftp.microsoft.com
If 'ftp' is not a typo you can use command-not-found to lookup the package that contains it, like this:
    cnf ftp
SLES12SP1:~ # cnf ftp
The program 'ftp' can be found in following packages:
  * lftp [ path: /usr/bin/ftp, repository: zypp (SLES12-SP1-12.1-0) ]
  * lftp [ path: /usr/bin/ftp, repository: zypp (SUSE_Linux_Enterprise_Server_12_SP1_x86_64:SLES12-SP1-Pool) ]
  * lftp [ path: /usr/bin/ftp, repository: zypp (SUSE_Linux_Enterprise_Server_12_SP1_x86_64:SLES12-SP1-Updates) ]

Try installing with:
    zypper install lftp
```

Installez donc ce client :

```
SLES12SP1:~ # zypper install lftp
Refreshing service 'SUSE_Linux_Enterprise_Server_12_SP1_x86_64'.
Loading repository data...
Reading installed packages...
Resolving package dependencies...

The following NEW package is going to be installed:
  lftp

1 new package to install.
Overall download size: 774.6 KiB. Already cached: 0 B. After the operation, additional 3.0 MiB will be used.
Continue? [y/n/? shows all options] (y): y
```

Connectez-vous à ftp.microsoft.com :

```
SLES12SP1:~ # ftp ftp.microsoft.com
Wrapper for lftp to simulate compatibility with lukemftp
Name (trainee): anonymous
Password: hgdfguigioh
```

Une fois connecté, il convient d'utiliser la commande **help** pour afficher la liste des commandes disponibles :

```
lftp anonymous@ftp.microsoft.com:~> help
    !<shell-command>                    (commands)                          alias [<name> [<value>]]
ascii
    attach [PID]                        binary                              bookmark [SUBCMD]
cache [SUBCMD]
    cat [-b] <files>                    cd <rdir>                           chmod [OPTS] mode file...
close [-a]
    [re]cls [opts] [path/][pattern]     debug [OPTS] [<level>|off]          du [options] <dirs>
exit [<code>|bg]
    lftp-get [OPTS] <rfile> [-o <lfile>] glob [OPTS] <cmd> <args>           help [<cmd>]
```

```
    history -w file|-r file|-c|-l [cnt]  jobs [-v] [<job_no...>]             kill all|<job_no>
lcd <ldir>
    lftp [OPTS] <site>                   ln [-s] <file1> <file2>             ls [<args>]
    lftp-mget [OPTS] <files>             mirror [OPTS] [remote [local]]      mkdir [-p] <dirs>
module name [args]
    more <files>                         lftp-mput [OPTS] <files>            mrm <files>
mv <file1> <file2>
    [re]nlist [<args>]                   lftp-open [OPTS] <site>             pget [OPTS] <rfile> [-o <lfile>]
    lftp-put [OPTS] <lfile> [-o <rfile>] pwd [-p]                            queue [OPTS] [<cmd>]
quote <cmd>
    repeat [OPTS] [delay] [command]      rm [-r] [-f] <files>                rmdir [-f] <dirs>
    scache [<session_no>]                set [OPT] [<var> [<val>]]           site <site-cmd>
source <file>
    torrent [-O <dir>] <file|URL>...     lftp-user <user|URL> [<pass>]       type [ascii|binary]
wait [<jobno>]
    zcat <files>                         zmore <files>                       user <username> [<pass>]
    open <site> [<port]
```

Le caractère **!** permet d'exécuter une commande sur la machine cliente

```
lftp anonymous@ftp.microsoft.com:~> !pwd
/root
lftp anonymous@ftp.microsoft.com:~> pwd
ftp://anonymous@ftp.microsoft.com:21
```

```
lftp anonymous@ftp.microsoft.com:~> quit
SLES12SP1:~ #
```

Options de la commande ftp

Les options de cette commande sont :

```
SLES12SP1:~ # ftp --help
ftp: invalid option -- '-'
unknown character ?
usage: ftp [-adefnpR] [-o outfile] [-P port] [-r retry]
           [-T dir,max[,inc][[user@]host [port]]] [host:path[/]]
           [file:///file] [ftp://[user[:pass]@]host[:port]/path[/]]
           [http://[user[:pass]@]host[:port]/path] [...]
       ftp -u url file [...]
```

wget

La commande **wget** est utilisée pour récupérer un fichier via http ou ftp :

```
SLES12SP1:~ # wget wiki.i2tch.co.uk/downloads/file/wget_test
--2017-11-14 16:48:16--  http://www.i2tch.co.uk/files/wget_test
Resolving www.i2tch.co.uk (www.i2tch.co.uk)... 217.160.0.123, 2001:8d8:100f:f000::276
Connecting to www.i2tch.co.uk (www.i2tch.co.uk)|217.160.0.123|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 0 [text/plain]
Saving to: ‘wget_test’

    [ <=>                                                                                 ] 0           -
-.-K/s   in 0s

2017-11-14 16:48:17 (0.00 B/s) - ‘wget_test’ saved [0/0]
```

Options de la commande wget

Les options de cette commande sont :

```
SLES12SP1:~ # wget --help
```

```
GNU Wget 1.14, a non-interactive network retriever.
Usage: wget [OPTION]... [URL]...

Mandatory arguments to long options are mandatory for short options too.

Startup:
  -V,  --version           display the version of Wget and exit.
  -h,  --help              print this help.
  -b,  --background        go to background after startup.
  -e,  --execute=COMMAND   execute a `.wgetrc'-style command.

Logging and input file:
  -o,  --output-file=FILE    log messages to FILE.
  -a,  --append-output=FILE  append messages to FILE.
  -d,  --debug               print lots of debugging information.
  -q,  --quiet               quiet (no output).
  -v,  --verbose             be verbose (this is the default).
  -nv, --no-verbose          turn off verboseness, without being quiet.
       --report-speed=TYPE   Output bandwidth as TYPE.  TYPE can be bits.
  -i,  --input-file=FILE     download URLs found in local or external FILE.
  -F,  --force-html          treat input file as HTML.
  -B,  --base=URL            resolves HTML input-file links (-i -F)
                             relative to URL.
       --config=FILE         Specify config file to use.

Download:
  -t,  --tries=NUMBER            set number of retries to NUMBER (0 unlimits).
       --retry-connrefused       retry even if connection is refused.
  -O,  --output-document=FILE    write documents to FILE.
  -nc, --no-clobber              skip downloads that would download to
                                 existing files (overwriting them).
  -c,  --continue                resume getting a partially-downloaded file.
       --progress=TYPE           select progress gauge type.
  -N,  --timestamping            don't re-retrieve files unless newer than
```

```
                                local.
  --no-use-server-timestamps    don't set the local file's timestamp by
                                the one on the server.
  -S,  --server-response        print server response.
       --spider                 don't download anything.
  -T,  --timeout=SECONDS        set all timeout values to SECONDS.
       --dns-timeout=SECS       set the DNS lookup timeout to SECS.
       --connect-timeout=SECS   set the connect timeout to SECS.
       --read-timeout=SECS      set the read timeout to SECS.
  -w,  --wait=SECONDS           wait SECONDS between retrievals.
       --waitretry=SECONDS      wait 1..SECONDS between retries of a retrieval.
       --random-wait            wait from 0.5*WAIT...1.5*WAIT secs between retrievals.
       --no-proxy               explicitly turn off proxy.
  -Q,  --quota=NUMBER           set retrieval quota to NUMBER.
       --bind-address=ADDRESS   bind to ADDRESS (hostname or IP) on local host.
       --limit-rate=RATE        limit download rate to RATE.
       --no-dns-cache           disable caching DNS lookups.
       --restrict-file-names=OS restrict chars in file names to ones OS allows.
       --ignore-case            ignore case when matching files/directories.
  -4,  --inet4-only             connect only to IPv4 addresses.
  -6,  --inet6-only             connect only to IPv6 addresses.
       --prefer-family=FAMILY   connect first to addresses of specified family,
                                one of IPv6, IPv4, or none.
       --user=USER              set both ftp and http user to USER.
       --password=PASS          set both ftp and http password to PASS.
       --ask-password           prompt for passwords.
       --no-iri                 turn off IRI support.
       --local-encoding=ENC     use ENC as the local encoding for IRIs.
       --remote-encoding=ENC    use ENC as the default remote encoding.
       --unlink                 remove file before clobber.

Directories:
  -nd, --no-directories          don't create directories.
  -x,  --force-directories       force creation of directories.
```

```
  -nH, --no-host-directories      don't create host directories.
       --protocol-directories     use protocol name in directories.
  -P,  --directory-prefix=PREFIX  save files to PREFIX/...
       --cut-dirs=NUMBER          ignore NUMBER remote directory components.

HTTP options:
       --http-user=USER        set http user to USER.
       --http-password=PASS    set http password to PASS.
       --no-cache              disallow server-cached data.
       --default-page=NAME     Change the default page name (normally
                               this is `index.html'.).
  -E,  --adjust-extension      save HTML/CSS documents with proper extensions.
       --ignore-length         ignore `Content-Length' header field.
       --header=STRING         insert STRING among the headers.
       --max-redirect          maximum redirections allowed per page.
       --proxy-user=USER       set USER as proxy username.
       --proxy-password=PASS   set PASS as proxy password.
       --referer=URL           include `Referer: URL' header in HTTP request.
       --save-headers          save the HTTP headers to file.
  -U,  --user-agent=AGENT      identify as AGENT instead of Wget/VERSION.
       --no-http-keep-alive    disable HTTP keep-alive (persistent connections).
       --no-cookies            don't use cookies.
       --load-cookies=FILE     load cookies from FILE before session.
       --save-cookies=FILE     save cookies to FILE after session.
       --keep-session-cookies  load and save session (non-permanent) cookies.
       --post-data=STRING      use the POST method; send STRING as the data.
       --post-file=FILE        use the POST method; send contents of FILE.
       --content-disposition   honor the Content-Disposition header when
                               choosing local file names (EXPERIMENTAL).
       --content-on-error      output the received content on server errors.
       --auth-no-challenge     send Basic HTTP authentication information
                               without first waiting for the server's
                               challenge.
```

```
HTTPS (SSL/TLS) options:
       --secure-protocol=PR     choose secure protocol, one of auto, SSLv2,
                                SSLv3, TLSv1, TLSv1.1, and TLSv1.2.
       --no-check-certificate   don't validate the server's certificate.
       --certificate=FILE       client certificate file.
       --certificate-type=TYPE  client certificate type, PEM or DER.
       --private-key=FILE       private key file.
       --private-key-type=TYPE  private key type, PEM or DER.
       --ca-certificate=FILE    file with the bundle of CA's.
       --ca-directory=DIR       directory where hash list of CA's is stored.
       --random-file=FILE       file with random data for seeding the SSL PRNG.
       --egd-file=FILE          file naming the EGD socket with random data.

FTP options:
       --ftp-user=USER         set ftp user to USER.
       --ftp-password=PASS     set ftp password to PASS.
       --no-remove-listing     don't remove `.listing' files.
       --no-glob               turn off FTP file name globbing.
       --no-passive-ftp        disable the "passive" transfer mode.
       --preserve-permissions  preserve remote file permissions.
       --retr-symlinks         when recursing, get linked-to files (not dir).

WARC options:
       --warc-file=FILENAME      save request/response data to a .warc.gz file.
       --warc-header=STRING      insert STRING into the warcinfo record.
       --warc-max-size=NUMBER    set maximum size of WARC files to NUMBER.
       --warc-cdx                write CDX index files.
       --warc-dedup=FILENAME     do not store records listed in this CDX file.
       --no-warc-compression     do not compress WARC files with GZIP.
       --no-warc-digests         do not calculate SHA1 digests.
       --no-warc-keep-log        do not store the log file in a WARC record.
       --warc-tempdir=DIRECTORY  location for temporary files created by the
                                 WARC writer.
```

```
Recursive download:
  -r,  --recursive          specify recursive download.
  -l,  --level=NUMBER       maximum recursion depth (inf or 0 for infinite).
       --delete-after       delete files locally after downloading them.
  -k,  --convert-links      make links in downloaded HTML or CSS point to
                            local files.
  -K,  --backup-converted   before converting file X, back up as X.orig.
  -m,  --mirror             shortcut for -N -r -l inf --no-remove-listing.
  -p,  --page-requisites    get all images, etc. needed to display HTML page.
       --strict-comments    turn on strict (SGML) handling of HTML comments.

Recursive accept/reject:
  -A,  --accept=LIST               comma-separated list of accepted extensions.
  -R,  --reject=LIST               comma-separated list of rejected extensions.
       --accept-regex=REGEX        regex matching accepted URLs.
       --reject-regex=REGEX        regex matching rejected URLs.
       --regex-type=TYPE           regex type (posix).
  -D,  --domains=LIST              comma-separated list of accepted domains.
       --exclude-domains=LIST      comma-separated list of rejected domains.
       --follow-ftp                follow FTP links from HTML documents.
       --follow-tags=LIST          comma-separated list of followed HTML tags.
       --ignore-tags=LIST          comma-separated list of ignored HTML tags.
  -H,  --span-hosts                go to foreign hosts when recursive.
  -L,  --relative                  follow relative links only.
  -I,  --include-directories=LIST  list of allowed directories.
  --trust-server-names             use the name specified by the redirection
                                   url last component.
  -X,  --exclude-directories=LIST  list of excluded directories.
  -np, --no-parent                 don't ascend to the parent directory.

Mail bug reports and suggestions to <bug-wget@gnu.org>.
```

ssh

Introduction

La commande **ssh** est le successeur et la remplaçante de la commande **rlogin**. La commande ssh permet d'établir des connexions sécurisées avec une machine distante :

```
SLES12SP1:~ # ssh -l trainee localhost
The authenticity of host 'localhost (::1)' can't be established.
ECDSA key fingerprint is 63:f5:33:05:ca:da:d9:bc:43:b4:48:ec:a6:66:25:0c [MD5].
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'localhost' (ECDSA) to the list of known hosts.
Password:
Last login: Fri Oct 27 19:21:44 2017 from 10.0.2.2
trainee@SLES12SP1:~> pwd
/home/trainee
trainee@SLES12SP1:~> whoami
trainee
```

Important - Notez que dans cet exemple vous vous connectez au serveur ssh sur votre propre machine virtuelle en tant que l'utilisateur **trainee**.

Pour fermer la connexion, utilisez la commande **exit** :

```
trainee@SLES12SP1:~> exit
logout
Connection to localhost closed.
```

SSH comporte cinq acteurs :

- Le **serveur SSH**
  - le démon sshd, qui s'occupe des authentifications et autorisations des clients,
- Le **client SSH**
  - ssh ou scp, qui assure la connexion et le dialogue avec le serveur,
- La **session** qui représente la connexion courante et qui commence juste après l'authentification réussie,
- Les **clefs**
  - **Couple de clef utilisateur asymétriques** et persistantes qui assurent l'identité d'un utilisateur et qui sont stockés sur disque dur,
  - **Clef hôte asymétrique et persistante** garantissant l'identité du serveur er qui est conservé sur disque dur
  - **Clef serveur asymétrique et temporaire** utilisée par le protocole SSH1 qui sert au chiffrement de la clé de session,
  - **Clef de session symétrique qui est générée aléatoirement** et qui permet le chiiffrement de la communication entre le client et le serveur. Elle est détruite en fin de session. SSH-1 utilise une seule clef tandis que SSH-2 utilise une clef par direction de la communication,
- La **base de données des hôtes connus** qui stocke les clés des connexions précédentes.

SSH fonctionne de la manière suivante pour la la mise en place d'un canal sécurisé:

- Le client contacte le serveur sur son port 22,
- Les client et le serveur échangent leur version de SSH. En cas de non-compatibilité de versions, l'un des deux met fin au processus,
- Le serveur SSH s'identifie auprès du client en lui fournissant :
  - Sa clé hôte,
  - Sa clé serveur,
  - Une séquence aléatoire de huit octets à inclure dans les futures réponses du client,
  - Une liste de méthodes de chiffrage, compression et authentification,
- Le client et le serveur produisent un identifiant identique, un haché MD5 long de 128 bits contenant la clé hôte, la clé serveur et la séquence aléatoire,
- Le client génère sa clé de session symétrique et la chiffre deux fois de suite, une fois avec la clé hôte du serveur et la deuxième fois avec la clé serveur. Le client envoie cette clé au serveur accompagnée de la séquence aléatoire et un choix d'algorithmes supportés,
- Le serveur déchiffre la clé de session,
- Le client et le serveur mettent en place le canal sécurisé.

SSH-1

SSH-1 utilise une paire de clefs de type RSA1. Il assure l'intégrité des données par une **Contrôle de Redondance Cyclique** (CRC) et est un bloc dit **monolithique**.

Afin de s'identifier, le client essaie chacune des six méthodes suivantes :

- **Kerberos**,
- **Rhosts**,
- **RhostsRSA**,
- Par **clef asymétrique**,
- **TIS**,
- Par **mot de passe**.

SSH-2

SSH-2 utilise **DSA** ou **RSA**. Il assure l'intégrité des données par l'algorithme **HMAC**. SSH-2 est organisé en trois **couches** :

- **SSH-TRANS** – Transport Layer Protocol,
- **SSH-AUTH** – Authentification Protocol,
- **SSH-CONN** – Connection Protocol.

SSH-2 diffère de SSH-1 essentiellement dans la phase authentification.

Trois méthodes d'authentification :

- Par **clef asymétrique**,
  - Identique à SSH-1 sauf avec l'algorithme DSA,
- **RhostsRSA**,
- Par **mot de passe**.

L'authentification par mot de passe

L'utilisateur fournit un mot de passe au client ssh. Le client ssh le transmet de façon sécurisée au serveur ssh puis le serveur vérifie le mot de passe et l'accepte ou non.

Avantage:

- Aucune configuration de clef asymétrique n'est nécessaire.

Inconvénients:

- L'utilisateur doit fournir à chaque connexion un identifiant et un mot de passe,
- Moins sécurisé qu'un système par clef asymétrique.

L'authentification par clef asymétrique

- Le **client** envoie au serveur une requête d'authentification par clé asymétrique qui contient le module de la clé à utiliser,
- Le **serveur** recherche une correspondance pour ce module dans le fichier des clés autorisés **~/.ssh/authorized_keys**,
  - Dans le cas où une correspondance n'est pas trouvée, le serveur met fin à la communication,
  - Dans le cas contraire le serveur génère une chaîne aléatoire de 256 bits appelée un **challenge** et la chiffre avec la **clé publique du client**,
- Le **client** reçoit le challenge et le décrypte avec la partie privée de sa clé. Il combine le challenge avec l'identifiant de session et chiffre le résultat. Ensuite il envoie le résultat chiffré au serveur.
- Le **serveur** génère le même haché et le compare avec celui reçu du client. Si les deux hachés sont identiques, l'authentification est réussie.

Options de la commande ssh

Les options de cette commande sont :

```
SLES12SP1:~ # ssh --help
unknown option -- -
usage: ssh [-1246AaCfgKkMNnqsTtVvXxYy] [-b bind_address] [-c cipher_spec]
           [-D [bind_address:]port] [-E log_file] [-e escape_char]
           [-F configfile] [-I pkcs11] [-i identity_file]
           [-L [bind_address:]port:host:hostport] [-l login_name] [-m mac_spec]
           [-O ctl_cmd] [-o option] [-p port]
           [-Q cipher | cipher-auth | mac | kex | key]
           [-R [bind_address:]port:host:hostport] [-S ctl_path] [-W host:port]
           [-w local_tun[:remote_tun]] [user@]hostname [command]
```

Configuration

La configuration du serveur s'effectue dans le fichier **/etc/ssh/sshd_config** :

```
SLES12SP1:~ # cat /etc/ssh/sshd_config
#   $OpenBSD: sshd_config,v 1.93 2014/01/10 05:59:19 djm Exp $

# This is the sshd server system-wide configuration file.  See
# sshd_config(5) for more information.

# This sshd was compiled with PATH=/usr/bin:/bin:/usr/sbin:/sbin

# The strategy used for options in the default sshd_config shipped with
# OpenSSH is to specify options with their default value where
# possible, but leave them commented.  Uncommented options override the
# default value.

#Port 22
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::

# The default requires explicit activation of protocol 1
#Protocol 2

# HostKey for protocol version 1
#HostKey /etc/ssh/ssh_host_key
# HostKeys for protocol version 2
#HostKey /etc/ssh/ssh_host_rsa_key
#HostKey /etc/ssh/ssh_host_dsa_key
#HostKey /etc/ssh/ssh_host_ecdsa_key
#HostKey /etc/ssh/ssh_host_ed25519_key
```

```
# Minimum accepted size of the DH parameter p. By default this is set to 1024
# to maintain compatibility with RFC4419, but should be set higher.
# Upstream default is identical to setting this to 2048.
#KexDHMin 1024

# Lifetime and size of ephemeral version 1 server key
#KeyRegenerationInterval 1h
#ServerKeyBits 1024

# Ciphers and keying
#RekeyLimit default none

# Logging
# obsoletes QuietMode and FascistLogging
#SyslogFacility AUTH
#LogLevel INFO

# Authentication:

#LoginGraceTime 2m
#PermitRootLogin yes
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10

#RSAAuthentication yes
#PubkeyAuthentication yes

# The default is to check both .ssh/authorized_keys and .ssh/authorized_keys2
# but this is overridden so installations will only check .ssh/authorized_keys
AuthorizedKeysFile  .ssh/authorized_keys

#AuthorizedPrincipalsFile none
```

```
#AuthorizedKeysCommand none
#AuthorizedKeysCommandUser nobody

# For this to work you will also need host keys in /etc/ssh/ssh_known_hosts
#RhostsRSAAuthentication no
# similar for protocol version 2
#HostbasedAuthentication no
# Change to yes if you don't trust ~/.ssh/known_hosts for
# RhostsRSAAuthentication and HostbasedAuthentication
#IgnoreUserKnownHosts no
# Don't read the user's ~/.rhosts and ~/.shosts files
#IgnoreRhosts yes

# To disable tunneled clear text passwords, change to no here!
PasswordAuthentication no
#PermitEmptyPasswords no

# Change to no to disable s/key passwords
#ChallengeResponseAuthentication yes

# Kerberos options
#KerberosAuthentication no
#KerberosOrLocalPasswd yes
#KerberosTicketCleanup yes
#KerberosGetAFSToken no

# GSSAPI options
#GSSAPIAuthentication no
#GSSAPICleanupCredentials yes
#GSSAPIStrictAcceptorCheck yes
#GSSAPIKeyExchange no

# Set this to 'yes' to enable support for the deprecated 'gssapi' authentication
# mechanism to OpenSSH 3.8p1. The newer 'gssapi-with-mic' mechanism is included
```

```
# in this release. The use of 'gssapi' is deprecated due to the presence of
# potential man-in-the-middle attacks, which 'gssapi-with-mic' is not susceptible to.
#GSSAPIEnableMITMAttack no



# Set this to 'yes' to enable PAM authentication, account processing,
# and session processing. If this is enabled, PAM authentication will
# be allowed through the ChallengeResponseAuthentication and
# PasswordAuthentication.  Depending on your PAM configuration,
# PAM authentication via ChallengeResponseAuthentication may bypass
# the setting of "PermitRootLogin without-password".
# If you just want the PAM account and session checks to run without
# PAM authentication, then enable this but set PasswordAuthentication
# and ChallengeResponseAuthentication to 'no'.
UsePAM yes

#AllowAgentForwarding yes
#AllowTcpForwarding yes
#GatewayPorts no
X11Forwarding yes
#X11DisplayOffset 10
#X11UseLocalhost yes
#PermitTTY yes
#PrintMotd yes
#PrintLastLog yes
#TCPKeepAlive yes
#UseLogin no
UsePrivilegeSeparation sandbox      # Default for new installations.
#PermitUserEnvironment no
#Compression delayed
#ClientAliveInterval 0
#ClientAliveCountMax 3
#UseDNS yes
#PidFile /run/sshd.pid
```

```
#MaxStartups 10:30:100
#PermitTunnel no
#ChrootDirectory none
#VersionAddendum none

# no default banner path
#Banner none

# override default of no subsystems
Subsystem   sftp    /usr/lib/ssh/sftp-server

# This enables accepting locale enviroment variables LC_* LANG, see sshd_config(5).
AcceptEnv LANG LC_CTYPE LC_NUMERIC LC_TIME LC_COLLATE LC_MONETARY LC_MESSAGES
AcceptEnv LC_PAPER LC_NAME LC_ADDRESS LC_TELEPHONE LC_MEASUREMENT
AcceptEnv LC_IDENTIFICATION LC_ALL

# Example of overriding settings on a per-user basis
#Match User anoncvs
#   X11Forwarding no
#   AllowTcpForwarding no
#   PermitTTY no
#   ForceCommand cvs server
```

Pour ôter les lignes de commentaires dans ce fichier, utilisez la commande suivante :

```
SLES12SP1:~ # cd /tmp ; grep -E -v '^(#|$)'  /etc/ssh/sshd_config > sshd_config
SLES12SP1:/tmp # cat sshd_config
AuthorizedKeysFile  .ssh/authorized_keys
PasswordAuthentication no

UsePAM yes
X11Forwarding yes
UsePrivilegeSeparation sandbox      # Default for new installations.
Subsystem   sftp    /usr/lib/ssh/sftp-server
```

```
AcceptEnv LANG LC_CTYPE LC_NUMERIC LC_TIME LC_COLLATE LC_MONETARY LC_MESSAGES
AcceptEnv LC_PAPER LC_NAME LC_ADDRESS LC_TELEPHONE LC_MEASUREMENT
AcceptEnv LC_IDENTIFICATION LC_ALL
```

Pour sécuriser le serveur ssh, ajoutez ou modifiez les directives suivantes :

```
PasswordAuthentication yes
AllowGroups adm
Banner /etc/issue.net
HostbasedAuthentication no
IgnoreRhosts yes
LoginGraceTime 60
LogLevel INFO
PermitEmptyPasswords no
PermitRootLogin no
PrintLastLog yes
Protocol 2
StrictModes yes
X11Forwarding no
```

Votre fichier ressemblera à celui-ci :

```
SLES12SP1:/tmp # vi sshd_config
SLES12SP1:/tmp # cat sshd_config
AuthorizedKeysFile  .ssh/authorized_keys
PasswordAuthentication yes
AllowGroups adm
Banner /etc/issue.net
HostbasedAuthentication no
IgnoreRhosts yes
LoginGraceTime 60
LogLevel INFO
PermitEmptyPasswords no
PermitRootLogin no
```

```
PrintLastLog yes
Protocol 2
StrictModes yes
UsePAM yes
X11Forwarding no
UsePrivilegeSeparation sandbox      # Default for new installations.
Subsystem   sftp    /usr/lib/ssh/sftp-server
AcceptEnv LANG LC_CTYPE LC_NUMERIC LC_TIME LC_COLLATE LC_MONETARY LC_MESSAGES
AcceptEnv LC_PAPER LC_NAME LC_ADDRESS LC_TELEPHONE LC_MEASUREMENT
AcceptEnv LC_IDENTIFICATION LC_ALL
```

A Faire - Renommez le fichier **/etc/ssh/sshd_config** en **/etc/ssh/sshd_config.old** puis copiez le fichier **/tmp/sshd_config** vers **/etc/ssh/**.

Ajoutez l'utilisateur trainee au groupe **adm** :

```
SLES12SP1:~ # groupadd adm
SLES12SP1:~ # usermod -G adm trainee
SLES12SP1:~ # groups trainee
trainee : users adm
```

Re-démarrez le service sshd :

```
SLES12SP1:~ # systemctl restart sshd
```

Pour générer les clefs sur le serveur saisissez la commande suivante en tant que **root**:

Lors de la génération des clefs, la passphrase doit être **vide**.

```
SLES12SP1:~ # ssh-keygen -t dsa
Generating public/private dsa key pair.
```

```
Enter file in which to save the key (/root/.ssh/id_dsa): /etc/ssh/ssh_host_dsa_key
/etc/ssh/ssh_host_dsa_key already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /etc/ssh/ssh_host_dsa_key.
Your public key has been saved in /etc/ssh/ssh_host_dsa_key.pub.
The key fingerprint is:
ba:e5:7e:e1:24:10:35:18:ae:12:80:e2:a3:15:7a:19 [MD5] root@SLES12SP1.fenestros.loc
The key's randomart image is:
+--[ DSA 1024]----+
|o     .+o        |
|o.E  .o  .       |
|o..+  ..         |
|.o+. ..          |
|.oo .  .S        |
|.  .   .. o      |
|      . .+ .     |
|       +  o      |
|      ..o.       |
+--[MD5]----------+
```

Important : Le chemin à indiquer pour le fichier est **/etc/ssh/ssh_host_dsa_key**. La passphrase doit être vide.

Générez da la même façon les clefs au format **RSA**, **ECDSA** et **ED25519** :

```
SLES12SP1:~ # ssh-keygen -t rsa
SLES12SP1:~ # ssh-keygen -t ecdsa
SLES12SP1:~ # ssh-keygen -t ed25519
```

Les clefs publiques générées possèdent l'extension **.pub**. Les clefs privées n'ont pas d'extension :

```
SLES12SP1:~ # ls /etc/ssh
ldap.conf   ssh_host_dsa_key      ssh_host_ecdsa_key.pub    ssh_host_key      ssh_host_rsa_key.pub
moduli      ssh_host_dsa_key.pub  ssh_host_ed25519_key      ssh_host_key.pub  sshd_config
ssh_config  ssh_host_ecdsa_key    ssh_host_ed25519_key.pub  ssh_host_rsa_key
```

Re-démarrez ensuite le service sshd :

```
SLES12SP1:~ # systemctl restart sshd.service
```

Saisissez maintenant les commandes suivantes en tant que **trainee** :

Important - Lors de la génération des clefs, la passphrase doit être **vide**.

```
SLES12SP1:~ # exit
trainee@SLES12SP1:~> ssh-keygen -t dsa
Generating public/private dsa key pair.
Enter file in which to save the key (/home/trainee/.ssh/id_dsa):
Created directory '/home/trainee/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/trainee/.ssh/id_dsa.
Your public key has been saved in /home/trainee/.ssh/id_dsa.pub.
The key fingerprint is:
14:dd:b1:94:fa:40:e3:f4:e8:3c:66:3d:01:a6:5f:8b [MD5] trainee@SLES12SP1.fenestros.loc
The key's randomart image is:
+--[ DSA 1024]----+
|        .. .oo   |
|         .B.o.   |
|        .* *.    |
|       .. = +    |
|        S+ * o   |
```

```
|          E =    |
|         o . .   |
|                 |
|                 |
+--[MD5]----------+

trainee@SLES12SP1:~> ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/trainee/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/trainee/.ssh/id_rsa.
Your public key has been saved in /home/trainee/.ssh/id_rsa.pub.
The key fingerprint is:
e3:c9:56:ec:69:89:42:0d:ba:32:bd:e5:b3:18:e3:b4 [MD5] trainee@SLES12SP1.fenestros.loc
The key's randomart image is:
+--[ RSA 2048]----+
|                 |
|                 |
|      .          |
|     . o .       |
|    . . S o      |
|   . o o * o     |
|  o * o * =      |
|   = O.o .       |
|    E oo         |
+--[MD5]----------+

trainee@SLES12SP1:~> ssh-keygen -t ecdsa
Generating public/private ecdsa key pair.
Enter file in which to save the key (/home/trainee/.ssh/id_ecdsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/trainee/.ssh/id_ecdsa.
```

```
Your public key has been saved in /home/trainee/.ssh/id_ecdsa.pub.
The key fingerprint is:
b9:bc:d9:6b:a5:a9:16:3f:f5:87:37:bd:b0:32:8e:77 [MD5] trainee@SLES12SP1.fenestros.loc
The key's randomart image is:
+--[ECDSA  256]---+
|                 |
|                 |
|                 |
|         .       |
|        S        |
|       ...  o    |
|        oo = o ..|
|        .+B+ E=.+|
|       .++=++. +o|
+--[MD5]----------+

trainee@SLES12SP1:~> ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/trainee/.ssh/id_ed25519):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/trainee/.ssh/id_ed25519.
Your public key has been saved in /home/trainee/.ssh/id_ed25519.pub.
The key fingerprint is:
ba:21:89:e9:8b:9a:4e:a6:81:f1:2a:ec:f4:8a:e5:2d [MD5] trainee@SLES12SP1.fenestros.loc
The key's randomart image is:
+--[ED25519  256--+
|                 |
|                 |
|                 |
|                 |
|.       S        |
|.o o . .         |
|++= o o          |
```

```
|BXE. . o         |
|%+=+. .          |
+--[MD5]----------+
```

Important - Les clés générées seront placées dans le répertoire **~/.ssh/**.

Utilisation

La commande ssh prend la forme suivante:

```
ssh -l nom_de_compte numero_ip (nom_de_machine)
```

En saisissant cette commande sur votre propre machine, vous obtiendrez un résultat similaire à celle-ci :

```
trainee@SLES12SP1:~> su -
Mot de passe :
SLES12SP1:~ # ssh -l trainee localhost
The authenticity of host 'localhost (::1)' can't be established.
ECDSA key fingerprint is 48:6a:6a:5c:00:51:02:e5:55:c5:df:59:38:68:d9:73 [MD5].
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'localhost' (ECDSA) to the list of known hosts.
Welcome to SUSE Linux Enterprise Server 12 SP1  (x86_64) - Kernel %r (%t).
Password:
Last login: Mon Nov 13 12:27:23 2017 from 2.2.0.10.rev.sfr.net
trainee@SLES12SP1:~> pwd
/home/trainee
trainee@SLES12SP1:~> whoami
trainee
trainee@SLES12SP1:~> exit
logout
```

```
Connection to localhost closed.
```

Tunnels SSH

Le protocole SSH peut être utilisé pour sécuriser les protocoles tels telnet, pop3 etc.. En effet, on peut créer un *tunnel* SSH dans lequel passe les communications du protocole non-sécurisé.

La commande pour créer un tunnel ssh prend la forme suivante :

```
ssh -N -f compte@hôte -Lport-local:localhost:port_distant
```

Dans votre cas, vous allez créer un tunnel dans votre propre vm entre le port 15023 et le port 23 :

```
SLES12SP1:~ # ssh -N -f trainee@localhost -L15023:localhost:23
Welcome to SUSE Linux Enterprise Server 12 SP1  (x86_64) - Kernel %r (%t).
Password: trainee
```

Installez maintenant le client et le serveur telnet :

```
SLES12SP1:~ # zypper install telnet-server
Refreshing service 'SUSE_Linux_Enterprise_Server_12_SP1_x86_64'.
Loading repository data...
Reading installed packages...
Resolving package dependencies...

The following NEW package is going to be installed:
  telnet-server

1 new package to install.
Overall download size: 32.3 KiB. Already cached: 0 B. After the operation, additional 60.9 KiB will be used.
Continue? [y/n/? shows all options] (y): y
```

Le serveur telnet est contrôlé par le service xinetd. Activez donc le serveur telnet :

```
SLES12SP1:~ # vi /etc/xinetd.d/telnet
SLES12SP1:~ # cat /etc/xinetd.d/telnet
# default: off
# description: Telnet is the old login server which is INSECURE and should \
#   therefore not be used. Use secure shell (openssh).
#   If you need telnetd not to "keep-alives" (e.g. if it runs over a ISDN \
#   uplink), add "-n".  See 'man telnetd' for more details.
service telnet
{
    socket_type    = stream
    protocol    = tcp
    wait        = no
    user        = root
    server        = /usr/sbin/in.telnetd
    disable        = no
}
```

Activez et démarrez le service xinetd :

```
SLES12SP1:~ # systemctl status xinetd.service
xinetd.service - Xinetd A Powerful Replacement For Inetd
   Loaded: loaded (/usr/lib/systemd/system/xinetd.service; disabled)
   Active: inactive (dead)

SLES12SP1:~ # systemctl enable xinetd.service
ln -s '/usr/lib/systemd/system/xinetd.service' '/etc/systemd/system/multi-user.target.wants/xinetd.service'
SLES12SP1:~ # systemctl start xinetd.service
SLES12SP1:~ # systemctl status xinetd.service
xinetd.service - Xinetd A Powerful Replacement For Inetd
   Loaded: loaded (/usr/lib/systemd/system/xinetd.service; enabled)
   Active: active (running) since Mon 2017-11-13 15:00:12 CET; 3s ago
 Main PID: 3286 (xinetd)
```

```
   CGroup: /system.slice/xinetd.service
           └─3286 /usr/sbin/xinetd -stayalive -dontfork

Nov 13 15:00:12 SLES12SP1.fenestros.loc xinetd[3286]: Reading included configuration file: /etc/xinetd.d/servers
[file=/etc/xi...e=12]
Nov 13 15:00:12 SLES12SP1.fenestros.loc xinetd[3286]: Reading included configuration file: /etc/xinetd.d/services
[file=/etc/x...e=14]
Nov 13 15:00:12 SLES12SP1.fenestros.loc xinetd[3286]: Reading included configuration file: /etc/xinetd.d/systat
[file=/etc/xin...e=14]
Nov 13 15:00:12 SLES12SP1.fenestros.loc xinetd[3286]: Reading included configuration file: /etc/xinetd.d/telnet
[file=/etc/xin...e=17]
Nov 13 15:00:12 SLES12SP1.fenestros.loc xinetd[3286]: Reading included configuration file: /etc/xinetd.d/time
[file=/etc/xinet...e=14]
Nov 13 15:00:12 SLES12SP1.fenestros.loc xinetd[3286]: Reading included configuration file: /etc/xinetd.d/time-udp
[file=/etc/x...e=15]
Nov 13 15:00:12 SLES12SP1.fenestros.loc xinetd[3286]: Reading included configuration file: /etc/xinetd.d/vnc
[file=/etc/xinetd...e=15]
Nov 13 15:00:12 SLES12SP1.fenestros.loc xinetd[3286]: Reading included configuration file: /etc/xinetd.d/vsftpd
[file=/etc/xin...e=90]
Nov 13 15:00:12 SLES12SP1.fenestros.loc xinetd[3286]: xinetd Version 2.3.15 started with libwrap loadavg options
compiled in.
Nov 13 15:00:12 SLES12SP1.fenestros.loc xinetd[3286]: Started working: 1 available service
Hint: Some lines were ellipsized, use -l to show in full.
```

Connectez-vous ensuite via telnet sur le port 15023, vous constaterez que votre connexion n'aboutit pas :

```
SLES12SP1:~ # telnet localhost 15023
Trying ::1...
Connected to localhost.
Escape character is '^]'.
Welcome to SUSE Linux Enterprise Server 12 SP1  (x86_64) - Kernel 3.12.74-60.64.40-default (1).
SLES12SP1 login: trainee
Password:
Last login: Mon Nov 13 12:34:12 CET 2017 from localhost on pts/1
```

```
Last login: Mon Nov 13 15:01:27 from localhost
trainee@SLES12SP1:~> exit
logout
Connection closed by foreign host.
```

Important : Notez bien que votre communication telnet passe par le tunnel SSH.

SCP

Introduction

La commande **scp** est le successeur et la remplaçante de la commande **rcp** de la famille des commandes **remote**. Il permet de faire des transferts sécurisés à partir d'une machine distante :

```
$ scp compte@numero_ip(nom_de_machine):/chemin_distant/fichier_distant /chemin_local/fichier_local
```

ou vers une machine distante :

```
$ scp /chemin_local/fichier_local compte@numero_ip(nom_de_machine):/chemin_distant/fichier_distant
```

Utilisation

Il convient maintenant de se connecter sur le «serveur» en utilisant ssh et vérifiez la présence du répertoire ~/.ssh :

En saisissant cette commande, vous obtiendrez une fenêtre similaire à celle-ci :

```
SLES12SP1:~ # ssh -l trainee 127.0.0.1
The authenticity of host '127.0.0.1 (127.0.0.1)' can't be established.
```

```
ECDSA key fingerprint is 48:6a:6a:5c:00:51:02:e5:55:c5:df:59:38:68:d9:73 [MD5].
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '127.0.0.1' (ECDSA) to the list of known hosts.
Welcome to SUSE Linux Enterprise Server 12 SP1  (x86_64) - Kernel %r (%t).
Password:
Last login: Mon Nov 13 15:01:27 2017 from localhost
trainee@SLES12SP1:~> ls -la | grep .ssh
drwx------ 1 trainee users  152 Nov 13 12:28 .ssh
trainee@SLES12SP1:~> exit
logout
Connection to 127.0.0.1 closed.
```

Important - Si le dossier distant .ssh n'existe pas dans le répertoire personnel de l'utilisateur connecté, il faut le créer avec des permissions de 700. Dans votre cas, puisque votre machine joue le rôle de serveur **et** du client, le dossier /home/trainee/.ssh existe **déjà**.

Créez ensuite le fichier **authorized_keys** localement en tant que trainee :

```
SLES12SP1:~ # exit
logout
trainee@SLES12SP1:~> cd .ssh/
trainee@SLES12SP1:~/.ssh> ls -l
total 32
-rw------- 1 trainee users  668 13 nov.  12:27 id_dsa
-rw-r--r-- 1 trainee users  621 13 nov.  12:27 id_dsa.pub
-rw------- 1 trainee users  227 13 nov.  12:28 id_ecdsa
-rw-r--r-- 1 trainee users  193 13 nov.  12:28 id_ecdsa.pub
-rw------- 1 trainee users  432 13 nov.  12:28 id_ed25519
-rw-r--r-- 1 trainee users  113 13 nov.  12:28 id_ed25519.pub
-rw------- 1 trainee users 1675 13 nov.  12:28 id_rsa
-rw-r--r-- 1 trainee users  413 13 nov.  12:28 id_rsa.pub
trainee@SLES12SP1:~/.ssh> cat id_ecdsa.pub id_rsa.pub id_dsa.pub id_ed25519.pub > authorized_keys
```

```
trainee@SLES12SP1:~/.ssh> cat authorized_keys
ecdsa-sha2-nistp256
AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBBcWTX8+QC5prnjPOP6UsJu3w8bnYF9kh9x9Z/UJ+UyODvZEeqsGW9IOy4nB8
0aZOO6qRHMvSc5nQtUGmFBZm4Y= trainee@SLES12SP1.fenestros.loc
ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQC3LZ55V74HMi9XFQp1hN1Op8PXMJNHjdW7HgvuAEG8Eb0c+/zbPTUpv3RRCMTHwUTFptvSsEnVcnfudqMqe
ljNxGrmIZKrnMizXo1P75j+a69PqK6cVI5MO6eCvtIFTLHJWBYjLXh7lj7CFsAFFErF+NFDcqht3gSaTRRDJe7tjbggHuRd5mg+QZ6pluecVXIFvG
LciFGz1hSAh0PGk4YtH8qSERxDBmKG6H7FRkUUJCmMGrMtYtzB1aZivTGK6Y/fC64oayYyfKQndFI7nZ3YY+s/zmhJW/sIcvY7to34Ow7wsgn9DcN
BX5Naf2PiIITLTBR2/LBOyTeTrLmUjvEV trainee@SLES12SP1.fenestros.loc
ssh-dss
AAAAB3NzaC1kc3MAAACBAP1Aq6fKVFIaZ/uttcLQsZylmpratT93ur0Ik7JewvehGzHKAPV49vfzlJsifswkXLdRayxqOXQ+Zc9YQjxCOuusZB4kR
Re6LpdQZm969LKUR+249ZrEV4YA8rVGX320uPnBvjZC8Iww1m95AA59gkGUo8mMm2Y4CX7DzS2eNe2TAAAAFQD5uwo5vMxVeHZ4zBCFIiiCDZtubQ
AAAIEAjLVuYjTyopnFNmw5hkiRaBo2TqpUOmUhBg1AcJs86oFtuhQz2opXbZ8gO99+7uEqIBYwktkttw5/v8a049qCd493PUfRBsnnAFhyAXzBczH
A8DuPeYZyWbapIk3igO916s92mlAgYKmIrUOM9EBNZw4Ov4vfacU/kTzXu+YTv20AAACAZN9czGAx63uoiivXzwjMtB7TUZHEHO7uMjEQRBAhkF+a
WLmXgOUIbLQwfjADw+//95NOeWnbLLGcY7qPqgZ52iQBXqCNx41kJZ5Wp++AJDe/5zI1icD3X+ukgWUbyIDZHQsTKrQBhXc54VKYpDmCtl/tekteP
MT1MtIPRCASPcw= trainee@SLES12SP1.fenestros.loc
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIFMCOPyxgkM6Wpn44p2b3UYFkydXKJ4XOrcaa7hjKvdO trainee@SLES12SP1.fenestros.loc
```

Le fichier **authorized_keys** doit avoir les permissions de 600 :

```
trainee@SLES12SP1:~/.ssh> chmod 600 authorized_keys
```

Ensuite, il convient de transférer le fichier local **.ssh/authorized_keys** du «client» vers le «serveur» :

```
trainee@SLES12SP1:~/.ssh> cd ..
trainee@SLES12SP1:~> scp .ssh/authorized_keys trainee@127.0.0.1:/home/trainee/.ssh/authorized_keys
Welcome to SUSE Linux Enterprise Server 12 SP1  (x86_64) - Kernel %r (%t).
authorized_keys                                                                                100% 1340
1.3KB/s   00:00
```

Lors de la connexion suivante au serveur, l'authentification utilise le couple de clefs asymétrique et aucun mot de passe n'est requis :

```
trainee@SLES12SP1:~> ssh -l trainee localhost
The authenticity of host 'localhost (::1)' can't be established.
```

```
ECDSA key fingerprint is 48:6a:6a:5c:00:51:02:e5:55:c5:df:59:38:68:d9:73 [MD5].
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'localhost' (ECDSA) to the list of known hosts.
Welcome to SUSE Linux Enterprise Server 12 SP1  (x86_64) - Kernel %r (%t).
Last login: Mon Nov 13 15:05:02 2017 from localhost
trainee@SLES12SP1:~> exit
logout
Connection to localhost closed.
trainee@SLES12SP1:~> su -
Mot de passe :
SLES12SP1:~ # ssh -l trainee localhost
Welcome to SUSE Linux Enterprise Server 12 SP1  (x86_64) - Kernel %r (%t).
Password:
Last login: Mon Nov 13 15:18:19 2017 from localhost
trainee@SLES12SP1:~> exit
logout
Connection to localhost closed.
SLES12SP1:~ #
```

HERE

Le Pare-feu Netfilter/iptables

Introduction

Netfilter est composé de 5 *hooks* :

- NF_IP_PRE_ROUTING
- NF_IP_LOCAL_IN
- NF_IP_LOCAL_OUT
- NF_IP_FORWARD
- NF_IP_POSTROUTING

Ces hooks sont utilisés par deux branches, la première est celle concernée par les paquets qui entrent vers des services locaux :

- NF_IP_PRE_ROUTING > NF_IP_LOCAL_IN > NF_IP_LOCAL_OUT > NF_IP_POSTROUTING

tandis que la deuxième concerne les paquets qui traversent la passerelle:

- NF_IP_PRE_ROUTING > NF_IP_FORWARD > NF_IP_POSTROUTING

Si IPTABLES a été compilé en tant que module, son utilisation nécessite le chargement de plusieurs modules supplémentaires en fonction de la situation:

- iptable_filter
- iptable_mangle
- iptable_net
- etc

Netfilter est organisé en **tables**. La commande **iptables** de netfilter permet d'insérer des **policies** dans les **chaines**:

- La table **FILTER**
    - La chaîne INPUT
        - Concerne les paquets entrants
            - Policies: ACCEPT, DROP, REJECT
    - La chaîne OUTPUT
        - Concerne les paquets sortants
            - Policies: ACCEPT, DROP, REJECT
    - La chaîne FORWARD
        - Concerne les paquets traversant le par-feu.
            - Policies: ACCEPT, DROP, REJECT

Si aucune table n'est précisée, c'est la table FILTER qui s'applique par défaut.

- La table **NAT**
    - La chaîne PREROUTING
        - Permet de faire la translation d'adresse de destination
            - Cibles: SNAT, DNAT, MASQUERADE

        - La chaîne POSTROUTING
            - Permet de faire la translation d'adresse de la source
                - Cibles: SNAT, DNAT, MASQUERADE
        - Le cas spécifique OUTPUT
            - Permet la modification de la destination des paquets générés localement

- La table **MANGLE**
    - Permet le marquage de paquets générés localement (OUTPUT) et entrants (PREROUTING)

Les **policies** sont:

- ACCEPT
    - Permet d'accepter le paquet concerné
- DROP
    - Permet de rejeter le paquet concerné sans générer un message d'erreur
- REJECT
    - Permet de rejeter le paquet concerné en générant une message d'erreur

Les **cibles** sont:

- SNAT
    - Permet de modifier l'adresse source du paquet concerné
- DNAT
    - Permet de modifier l'adresse de destination du paquet concerné
- MASQUERADE
    - Permet de remplacer l'adresse IP privée de l'expéditeur par un socket public de la passerelle.

IPTABLES peut être configuré soit par des outils tels shorewall, soit en utilisant des lignes de commandes ou un script. Dans ce dernier cas, la ligne prend la forme:

```
# IPTABLES --action CHAINE --option1 --option2
```

Les actions sont:

Action	Abréviation	Déscription
- -append	-A	Ajouter une règle à la fin de la chaîne spécifiée
- -delete	-D	Supprimer une règle en spécifiant son numéro ou la règle à supprimer
- -replace	-R	Permet de remplacer la règle spécifée par son numéro
- -insert	-I	Permet d'insérer une règle à l'endroit spécifié
- -list	-L	Permet d'afficher des règles
- -flush	-F	Permet de vider toutes les règles d'une chaîne

Les options sont:

Option	Abréviation	Déscription
- -protocol	-p	Permet de spécifier un protocol - tcp, udp, icmp, all
- -source	-s	Permet de spécifier une adresse source
- -destination	-d	Permet de spécifier une adresse de destination
- -in-interface	-i	Permet de spécifier une interface réseau d'entrée
- -out-interface	-o	Permet de spécifier une interface réseau de sortie
- -fragment	-f	Permet de ne spécifier que les paquets fragmentés
- -source-port	-sport	Permet de spécifier un port source ou une plage de ports source
- -destination-port	-dport	Permet de spécifier un port de destination ou une plage de ports de destination
- -tcp-flags	s/o	Permet de spécifier un flag TCP à matcher - SYN, ACK, FIN, RST, URG, PSH, ALL, NONE
- -icmp-type	s/o	Permet de spécifier un type de paquet ICMP
- -mac-source	s/o	Permet de spécifier une adresse MAC

Les options spécifiques à NET sont:

- -to-destination	s/o	Permet de spécifier l'adresse de destination d'une translation
- -to-source	s/o	Permet spécifier l'adresse source d'une translation

Les options spécifiques aux LOGS sont:

- -log-level	s/o	Permet de spécifier le niveau de logs
- -log-prefix	s/o	Permet de spécifier un préfix pour les logs

L'option spécifique au STATEFUL est:

- -state	s/o	Permet de spécifier l'état du paquet à vérifier

Ce dernier cas fait référence au STATEFUL. Le STATEFUL est la capacité du par-feu à enregistrer dans une table spécifique, l'état des différentes connexions. Cette table s'appelle une **table d'état**. Le principe du fonctionnement de STATEFUL est simple, à savoir, si le paquet entrant appartient à une communication déjà établie, celui-ci n'est pas vérifié.

Il existe 4 états:

- NEW
  - Le paquet concerne une nouvelle connexion et contient donc un flag SYN à 1
- ESTABLISHED
  - Le paquet concerne une connexion déjà établie. Le paquet ne doit contenir **ni** flag SYN à 1, **ni** flag FIN à 1
- RELATED
  - Le paquet est d'une connexion qui présente une relation avec une autre connexion
- INVALID
  - La paquet provient d'une connexion anormale.

Configuration par Scripts

Dans l'exemple suivant, expliquez le fonctionnement du script en détaillant les règles écrites :

```
#!/bin/bash
####################################
# proxy server IP
PROXY_SERVER="192.168.1.2"
# Interface connected to Internet
INTERNET="eth1"
# Interface connected to LAN
LAN_IN="eth0"
# Local Interface
LOCAL="lo"
```

```
# Squid port
PROXY_PORT="8080"
# DO NOT MODIFY BELOW
# Clean old firewall
iptables -F
iptables -X
iptables -t nat -F
iptables -t nat -X
iptables -t mangle -F
iptables -t mangle -X
# Load IPTABLES modules for NAT and IP conntrack support
modprobe ip_conntrack
modprobe ip_conntrack_ftp
# For win xp ftp client
modprobe ip_nat_ftp
echo 1 > /proc/sys/net/ipv4/ip_forward
# Setting default filter policy
iptables -P INPUT DROP
iptables -P OUTPUT ACCEPT
# Unlimited access to loop back
iptables -A INPUT -i lo -j ACCEPT
iptables -A OUTPUT -o lo -j ACCEPT
# Allow UDP, DNS and Passive FTP
iptables -A INPUT -i $INTERNET -m state --state ESTABLISHED,RELATED -j ACCEPT
# set this system as a router for Rest of LAN
iptables --table nat --append POSTROUTING --out-interface $INTERNET -j MASQUERADE
iptables --append FORWARD --in-interface $LAN_IN -j ACCEPT
# unlimited access to LAN
iptables -A INPUT -i $LAN_IN -j ACCEPT
iptables -A OUTPUT -o $LAN_IN -j ACCEPT
# DNAT port 80 request comming from LAN systems to squid 3128 ($SQUID_PORT) aka transparent proxy
iptables -t nat -A PREROUTING -i $LAN_IN -p tcp --dport 80 -j DNAT --to $PROXY_SERVER:$PROXY_PORT
# iptables -t nat -A PREROUTING -i br0 -p tcp --dport 80 -j REDIRECT --to-port 3128
# if it is same system
```

```
iptables -t nat -A PREROUTING -i $INTERNET -p tcp --dport 80 -j REDIRECT --to-port $PROXY_PORT
# DROP everything and Log it
iptables -A INPUT -j LOG
iptables -A INPUT -j DROP
```

Dans l'exemple suivant, expliquez le fonctionnement du script en détaillant les règles écrites :

```
#!/bin/sh
#
# Generated iptables firewall script for the Linux 2.4 kernel
# Script generated by Easy Firewall Generator for IPTables 1.15
# copyright 2002 Timothy Scott Morizot
#
# Redhat chkconfig comments - firewall applied early,
#                             removed late
# chkconfig: 2345 08 92
# description: This script applies or removes iptables firewall rules
#
# This generator is primarily designed for RedHat installations,
# although it should be adaptable for others.
#
# It can be executed with the typical start and stop arguments.
# If used with stop, it will stop after flushing the firewall.
# The save and restore arguments will save or restore the rules
# from the /etc/sysconfig/iptables file.  The save and restore
# arguments are included to preserve compatibility with
# Redhat's or Fedora's init.d script if you prefer to use it.

# Redhat/Fedora installation instructions
#
# 1. Have the system link the iptables init.d startup script into run states
#    2, 3, and 5.
#    chkconfig --level 235 iptables on
#
```

```
# 2. Save this script and execute it to load the ruleset from this file.
#    You may need to run the dos2unix command on it to remove carraige returns.
#
# 3. To have it applied at startup, copy this script to
#    /etc/init.d/iptables.  It accepts stop, start, save, and restore
#    arguments.  (You may wish to save the existing one first.)
#    Alternatively, if you issue the 'service iptables save' command
#    the init.d script should save the rules and reload them at runtime.
#
# 4. For non-Redhat systems (or Redhat systems if you have a problem), you
#    may want to append the command to execute this script to rc.local.
#    rc.local is typically located in /etc and /etc/rc.d and is usually
#    the last thing executed on startup.  Simply add /path/to/script/script_name
#    on its own line in the rc.local file.

##############################################################################
#
# Local Settings
#

# sysctl location.  If set, it will use sysctl to adjust the kernel parameters.
# If this is set to the empty string (or is unset), the use of sysctl
# is disabled.

SYSCTL="/sbin/sysctl -w"

# To echo the value directly to the /proc file instead
# SYSCTL=""

# IPTables Location - adjust if needed

IPT="/sbin/iptables"
IPTS="/sbin/iptables-save"
IPTR="/sbin/iptables-restore"
```

```
# Internet Interface
INET_IFACE="eth1"

# Local Interface Information
LOCAL_IFACE="eth0"
LOCAL_IP="192.168.1.1"
LOCAL_NET="192.168.1.0/24"
LOCAL_BCAST="192.168.1.255"

# Localhost Interface

LO_IFACE="lo"
LO_IP="127.0.0.1"

# Save and Restore arguments handled here
if [ "$1" = "save" ]
then
    echo -n "Saving firewall to /etc/sysconfig/iptables ... "
    $IPTS > /etc/sysconfig/iptables
    echo "done"
    exit 0
elif [ "$1" = "restore" ]
then
    echo -n "Restoring firewall from /etc/sysconfig/iptables ... "
    $IPTR < /etc/sysconfig/iptables
    echo "done"
    exit 0
fi

##############################################################################
#
# Load Modules
#
```

```
echo "Loading kernel modules ..."

# You should uncomment the line below and run it the first time just to
# ensure all kernel module dependencies are OK.  There is no need to run
# every time, however.

# /sbin/depmod -a

# Unless you have kernel module auto-loading disabled, you should not
# need to manually load each of these modules.  Other than ip_tables,
# ip_conntrack, and some of the optional modules, I've left these
# commented by default.  Uncomment if you have any problems or if
# you have disabled module autoload.  Note that some modules must
# be loaded by another kernel module.

# core netfilter module
/sbin/modprobe ip_tables

# the stateful connection tracking module
/sbin/modprobe ip_conntrack

# filter table module
# /sbin/modprobe iptable_filter

# mangle table module
# /sbin/modprobe iptable_mangle

# nat table module
# /sbin/modprobe iptable_nat

# LOG target module
# /sbin/modprobe ipt_LOG

# This is used to limit the number of packets per sec/min/hr
```

```
# /sbin/modprobe ipt_limit

# masquerade target module
# /sbin/modprobe ipt_MASQUERADE

# filter using owner as part of the match
# /sbin/modprobe ipt_owner

# REJECT target drops the packet and returns an ICMP response.
# The response is configurable.  By default, connection refused.
# /sbin/modprobe ipt_REJECT

# This target allows packets to be marked in the mangle table
# /sbin/modprobe ipt_mark

# This target affects the TCP MSS
# /sbin/modprobe ipt_tcpmss

# This match allows multiple ports instead of a single port or range
# /sbin/modprobe multiport

# This match checks against the TCP flags
# /sbin/modprobe ipt_state

# This match catches packets with invalid flags
# /sbin/modprobe ipt_unclean

# The ftp nat module is required for non-PASV ftp support
/sbin/modprobe ip_nat_ftp

# the module for full ftp connection tracking
/sbin/modprobe ip_conntrack_ftp

# the module for full irc connection tracking
```

```
/sbin/modprobe ip_conntrack_irc


###############################################################################
#
# Kernel Parameter Configuration
#
# See http://ipsysctl-tutorial.frozentux.net/chunkyhtml/index.html
# for a detailed tutorial on sysctl and the various settings
# available.

# Required to enable IPv4 forwarding.
# Redhat users can try setting FORWARD_IPV4 in /etc/sysconfig/network to true
# Alternatively, it can be set in /etc/sysctl.conf
if [ "$SYSCTL" = "" ]
then
    echo "1" > /proc/sys/net/ipv4/ip_forward
else
    $SYSCTL net.ipv4.ip_forward="1"
fi

# This enables dynamic address hacking.
# This may help if you have a dynamic IP address \(e.g. slip, ppp, dhcp\).
#if [ "$SYSCTL" = "" ]
#then
#    echo "1" > /proc/sys/net/ipv4/ip_dynaddr
#else
#    $SYSCTL net.ipv4.ip_dynaddr="1"
#fi

# This enables SYN flood protection.
# The SYN cookies activation allows your system to accept an unlimited
# number of TCP connections while still trying to give reasonable
# service during a denial of service attack.
```

```
if [ "$SYSCTL" = "" ]
then
    echo "1" > /proc/sys/net/ipv4/tcp_syncookies
else
    $SYSCTL net.ipv4.tcp_syncookies="1"
fi

# This enables source validation by reversed path according to RFC1812.
# In other words, did the response packet originate from the same interface
# through which the source packet was sent?  It's recommended for single-homed
# systems and routers on stub networks.  Since those are the configurations
# this firewall is designed to support, I turn it on by default.
# Turn it off if you use multiple NICs connected to the same network.
if [ "$SYSCTL" = "" ]
then
    echo "1" > /proc/sys/net/ipv4/conf/all/rp_filter
else
    $SYSCTL net.ipv4.conf.all.rp_filter="1"
fi

# This option allows a subnet to be firewalled with a single IP address.
# It's used to build a DMZ.  Since that's not a focus of this firewall
# script, it's not enabled by default, but is included for reference.
# See: http://www.sjdjweis.com/linux/proxyarp/
#if [ "$SYSCTL" = "" ]
#then
#    echo "1" > /proc/sys/net/ipv4/conf/all/proxy_arp
#else
#    $SYSCTL net.ipv4.conf.all.proxy_arp="1"
#fi

# The following kernel settings were suggested by Alex Weeks. Thanks!

# This kernel parameter instructs the kernel to ignore all ICMP
```

```
# echo requests sent to the broadcast address.  This prevents
# a number of smurfs and similar DoS nasty attacks.
if [ "$SYSCTL" = "" ]
then
    echo "1" > /proc/sys/net/ipv4/icmp_echo_ignore_broadcasts
else
    $SYSCTL net.ipv4.icmp_echo_ignore_broadcasts="1"
fi

# This option can be used to accept or refuse source routed
# packets.  It is usually on by default, but is generally
# considered a security risk.  This option turns it off.
if [ "$SYSCTL" = "" ]
then
    echo "0" > /proc/sys/net/ipv4/conf/all/accept_source_route
else
    $SYSCTL net.ipv4.conf.all.accept_source_route="0"
fi

# This option can disable ICMP redirects.  ICMP redirects
# are generally considered a security risk and shouldn't be
# needed by most systems using this generator.
#if [ "$SYSCTL" = "" ]
#then
#    echo "0" > /proc/sys/net/ipv4/conf/all/accept_redirects
#else
#    $SYSCTL net.ipv4.conf.all.accept_redirects="0"
#fi

# However, we'll ensure the secure_redirects option is on instead.
# This option accepts only from gateways in the default gateways list.
if [ "$SYSCTL" = "" ]
then
    echo "1" > /proc/sys/net/ipv4/conf/all/secure_redirects
```

```
else
    $SYSCTL net.ipv4.conf.all.secure_redirects="1"
fi

# This option logs packets from impossible addresses.
if [ "$SYSCTL" = "" ]
then
    echo "1" > /proc/sys/net/ipv4/conf/all/log_martians
else
    $SYSCTL net.ipv4.conf.all.log_martians="1"
fi


##############################################################################
#
# Flush Any Existing Rules or Chains
#

echo "Flushing Tables ..."

# Reset Default Policies
$IPT -P INPUT ACCEPT
$IPT -P FORWARD ACCEPT
$IPT -P OUTPUT ACCEPT
$IPT -t nat -P PREROUTING ACCEPT
$IPT -t nat -P POSTROUTING ACCEPT
$IPT -t nat -P OUTPUT ACCEPT
$IPT -t mangle -P PREROUTING ACCEPT
$IPT -t mangle -P OUTPUT ACCEPT

# Flush all rules
$IPT -F
$IPT -t nat -F
$IPT -t mangle -F
```

```
# Erase all non-default chains
$IPT -X
$IPT -t nat -X
$IPT -t mangle -X

if [ "$1" = "stop" ]
then
    echo "Firewall completely flushed!  Now running with no firewall."
    exit 0
fi

##############################################################################
#
# Rules Configuration
#

##############################################################################
#
# Filter Table
#
##############################################################################

# Set Policies

$IPT -P INPUT DROP
$IPT -P OUTPUT DROP
$IPT -P FORWARD DROP

##############################################################################
#
# User-Specified Chains
#
# Create user chains to reduce the number of rules each packet
# must traverse.
```

```
echo "Create and populate custom rule chains ..."

# Create a chain to filter INVALID packets

$IPT -N bad_packets

# Create another chain to filter bad tcp packets

$IPT -N bad_tcp_packets

# Create separate chains for icmp, tcp (incoming and outgoing),
# and incoming udp packets.

$IPT -N icmp_packets

# Used for UDP packets inbound from the Internet
$IPT -N udp_inbound

# Used to block outbound UDP services from internal network
# Default to allow all
$IPT -N udp_outbound

# Used to allow inbound services if desired
# Default fail except for established sessions
$IPT -N tcp_inbound

# Used to block outbound services from internal network
# Default to allow all
$IPT -N tcp_outbound

##############################################################################
#
# Populate User Chains
#
```

```
# bad_packets chain
#

# Drop packets received on the external interface
# claiming a source of the local network
$IPT -A bad_packets -p ALL -i $INET_IFACE -s $LOCAL_NET -j LOG \
    --log-prefix "fp=bad_packets:2 a=DROP "
$IPT -A bad_packets -p ALL -i $INET_IFACE -s $LOCAL_NET -j DROP

# Drop INVALID packets immediately
$IPT -A bad_packets -p ALL -m state --state INVALID -j LOG \
    --log-prefix "fp=bad_packets:1 a=DROP "

$IPT -A bad_packets -p ALL -m state --state INVALID -j DROP

# Then check the tcp packets for additional problems
$IPT -A bad_packets -p tcp -j bad_tcp_packets

# All good, so return
$IPT -A bad_packets -p ALL -j RETURN

# bad_tcp_packets chain
#
# All tcp packets will traverse this chain.
# Every new connection attempt should begin with
# a syn packet.  If it doesn't, it is likely a
# port scan.  This drops packets in state
# NEW that are not flagged as syn packets.

# Return to the calling chain if the bad packets originate
# from the local interface. This maintains the approach
# throughout this firewall of a largely trusted internal
# network.
$IPT -A bad_tcp_packets -p tcp -i $LOCAL_IFACE -j RETURN
```

```
# However, I originally did apply this filter to the forward chain
# for packets originating from the internal network.  While I have
# not conclusively determined its effect, it appears to have the
# interesting side effect of blocking some of the ad systems.
# Apparently some ad systems have the browser initiate a NEW
# connection that is not flagged as a syn packet to retrieve
# the ad image.  If you wish to experiment further comment the
# rule above. If you try it, you may also wish to uncomment the
# rule below.  It will keep those packets from being logged.
# There are a lot of them.
# $IPT -A bad_tcp_packets -p tcp -i $LOCAL_IFACE ! --syn -m state \
#     --state NEW -j DROP

$IPT -A bad_tcp_packets -p tcp ! --syn -m state --state NEW -j LOG \
    --log-prefix "fp=bad_tcp_packets:1 a=DROP "
$IPT -A bad_tcp_packets -p tcp ! --syn -m state --state NEW -j DROP

$IPT -A bad_tcp_packets -p tcp --tcp-flags ALL NONE -j LOG \
    --log-prefix "fp=bad_tcp_packets:2 a=DROP "
$IPT -A bad_tcp_packets -p tcp --tcp-flags ALL NONE -j DROP

$IPT -A bad_tcp_packets -p tcp --tcp-flags ALL ALL -j LOG \
    --log-prefix "fp=bad_tcp_packets:3 a=DROP "
$IPT -A bad_tcp_packets -p tcp --tcp-flags ALL ALL -j DROP

$IPT -A bad_tcp_packets -p tcp --tcp-flags ALL FIN,URG,PSH -j LOG \
    --log-prefix "fp=bad_tcp_packets:4 a=DROP "
$IPT -A bad_tcp_packets -p tcp --tcp-flags ALL FIN,URG,PSH -j DROP

$IPT -A bad_tcp_packets -p tcp --tcp-flags ALL SYN,RST,ACK,FIN,URG -j LOG \
    --log-prefix "fp=bad_tcp_packets:5 a=DROP "
$IPT -A bad_tcp_packets -p tcp --tcp-flags ALL SYN,RST,ACK,FIN,URG -j DROP

$IPT -A bad_tcp_packets -p tcp --tcp-flags SYN,RST SYN,RST -j LOG \
```

```
    --log-prefix "fp=bad_tcp_packets:6 a=DROP "
$IPT -A bad_tcp_packets -p tcp --tcp-flags SYN,RST SYN,RST -j DROP

$IPT -A bad_tcp_packets -p tcp --tcp-flags SYN,FIN SYN,FIN -j LOG \
    --log-prefix "fp=bad_tcp_packets:7 a=DROP "
$IPT -A bad_tcp_packets -p tcp --tcp-flags SYN,FIN SYN,FIN -j DROP

# All good, so return
$IPT -A bad_tcp_packets -p tcp -j RETURN

# icmp_packets chain
#
# This chain is for inbound (from the Internet) icmp packets only.
# Type 8 (Echo Request) is not accepted by default
# Enable it if you want remote hosts to be able to reach you.
# 11 (Time Exceeded) is the only one accepted
# that would not already be covered by the established
# connection rule.  Applied to INPUT on the external interface.
#
# See: http://www.ee.siue.edu/~rwalden/networking/icmp.html
# for more info on ICMP types.
#
# Note that the stateful settings allow replies to ICMP packets.
# These rules allow new packets of the specified types.

# ICMP packets should fit in a Layer 2 frame, thus they should
# never be fragmented.  Fragmented ICMP packets are a typical sign
# of a denial of service attack.
$IPT -A icmp_packets --fragment -p ICMP -j LOG \
    --log-prefix "fp=icmp_packets:1 a=DROP "
$IPT -A icmp_packets --fragment -p ICMP -j DROP

# Echo - uncomment to allow your system to be pinged.
# Uncomment the LOG command if you also want to log PING attempts
```

```
#
# $IPT -A icmp_packets -p ICMP -s 0/0 --icmp-type 8 -j LOG \
#    --log-prefix "fp=icmp_packets:2 a=ACCEPT "
# $IPT -A icmp_packets -p ICMP -s 0/0 --icmp-type 8 -j ACCEPT

# By default, however, drop pings without logging. Blaster
# and other worms have infected systems blasting pings.
# Comment the line below if you want pings logged, but it
# will likely fill your logs.
$IPT -A icmp_packets -p ICMP -s 0/0 --icmp-type 8 -j DROP

# Time Exceeded
$IPT -A icmp_packets -p ICMP -s 0/0 --icmp-type 11 -j ACCEPT

# Not matched, so return so it will be logged
$IPT -A icmp_packets -p ICMP -j RETURN

# TCP & UDP
# Identify ports at:
#    http://www.chebucto.ns.ca/~rakerman/port-table.html
#    http://www.iana.org/assignments/port-numbers

# udp_inbound chain
#
# This chain describes the inbound UDP packets it will accept.
# It's applied to INPUT on the external or Internet interface.
# Note that the stateful settings allow replies.
# These rules are for new requests.
# It drops netbios packets (windows) immediately without logging.

# Drop netbios calls
# Please note that these rules do not really change the way the firewall
# treats netbios connections.  Connections from the localhost and
# internal interface (if one exists) are accepted by default.
```

```
# Responses from the Internet to requests initiated by or through
# the firewall are also accepted by default.  To get here, the
# packets would have to be part of a new request received by the
# Internet interface.  You would have to manually add rules to
# accept these.  I added these rules because some network connections,
# such as those via cable modems, tend to be filled with noise from
# unprotected Windows machines.  These rules drop those packets
# quickly and without logging them.  This prevents them from traversing
# the whole chain and keeps the log from getting cluttered with
# chatter from Windows systems.
$IPT -A udp_inbound -p UDP -s 0/0 --destination-port 137 -j DROP
$IPT -A udp_inbound -p UDP -s 0/0 --destination-port 138 -j DROP

# Ident requests (Port 113) must have a REJECT rule rather than the
# default DROP rule.  This is the minimum requirement to avoid
# long delays while connecting.  Also see the tcp_inbound rule.
$IPT -A udp_inbound -p UDP -s 0/0 --destination-port 113 -j REJECT

# A more sophisticated configuration could accept the ident requests.
# $IPT -A udp_inbound -p UDP -s 0/0 --destination-port 113 -j ACCEPT

# However, if this is a gateway system that masquerades/nats for internal systems
# and the internal systems wish to chat, a simple changing these rules to
# ACCEPT won't work.  The ident daemon on the gateway will need to know how
# to handle the requests.  The stock daemon in most linux distributions
# can't do that.   oidentd is one package that can.
# See: http://dev.ojnk.net/

# Network Time Protocol (NTP) Server
$IPT -A udp_inbound -p UDP -s 0/0 --destination-port 123 -j ACCEPT

# Dynamic Address
# If DHCP, the initial request is a broadcast. The response
# doesn't exactly match the outbound packet.  This explicitly
```

```
# allow the DHCP ports to alleviate this problem.
# If you receive your dynamic address by a different means, you
# can probably comment this line.
$IPT -A udp_inbound -p UDP -s 0/0 --source-port 67 --destination-port 68 \
     -j ACCEPT



# Not matched, so return for logging
$IPT -A udp_inbound -p UDP -j RETURN

# udp_outbound chain
#
# This chain is used with a private network to prevent forwarding for
# UDP requests on specific protocols.  Applied to the FORWARD rule from
# the internal network.  Ends with an ACCEPT



# No match, so ACCEPT
$IPT -A udp_outbound -p UDP -s 0/0 -j ACCEPT

# tcp_inbound chain
#
# This chain is used to allow inbound connections to the
# system/gateway.  Use with care.  It defaults to none.
# It's applied on INPUT from the external or Internet interface.

# Ident requests (Port 113) must have a REJECT rule rather than the
# default DROP rule.  This is the minimum requirement to avoid
# long delays while connecting.  Also see the tcp_inbound rule.
$IPT -A tcp_inbound -p TCP -s 0/0 --destination-port 113 -j REJECT

# A more sophisticated configuration could accept the ident requests.
# $IPT -A tcp_inbound -p TCP -s 0/0 --destination-port 113 -j ACCEPT
```

```
# However, if this is a gateway system that masquerades/nats for internal systems
# and the internal systems wish to chat, a simple changing these rules to
# ACCEPT won't work.  The ident daemon on the gateway will need to know how
# to handle the requests.  The stock daemon in most linux distributions
# can't do that.   oidentd is one package that can.
# See: http://dev.ojnk.net/

# Web Server

# HTTP
$IPT -A tcp_inbound -p TCP -s 0/0 --destination-port 80 -j ACCEPT

# HTTPS (Secure Web Server)
$IPT -A tcp_inbound -p TCP -s 0/0 --destination-port 443 -j ACCEPT

# FTP Server (Control)
$IPT -A tcp_inbound -p TCP -s 0/0 --destination-port 21 -j ACCEPT

# FTP Client (Data Port for non-PASV transfers)
$IPT -A tcp_inbound -p TCP -s 0/0 --source-port 20 -j ACCEPT

# Passive FTP
#
# With passive FTP, the server provides a port to the client
# and allows the client to initiate the connection rather
# than initiating the connection with the client from the data port.
# Web browsers and clients operating behind a firewall generally
# use passive ftp transfers.  A general purpose FTP server
# will need to support them.
#
# However, by default an FTP server will select a port from the entire
# range of high ports.  It is not particularly safe to open all
# high ports.  Fortunately, that range can be restricted.  This
# firewall presumes that the range has been restricted to a specific
```

```
# selected range.  That range must also be configured in the ftp server.
#
# Instructions for specifying the port range for the wu-ftpd server
# can be found here:
# http://www.wu-ftpd.org/man/ftpaccess.html
# (See the passive ports option.)
#
# Instructions for the ProFTPD server can be found here:
# http://proftpd.linux.co.uk/localsite/Userguide/linked/x861.html

$IPT -A tcp_inbound -p TCP -s 0/0 --destination-port 62000:64000 -j ACCEPT

# Email Server (SMTP)
$IPT -A tcp_inbound -p TCP -s 0/0 --destination-port 25 -j ACCEPT

# Email Server (POP3)
$IPT -A tcp_inbound -p TCP -s 0/0 --destination-port 110 -j ACCEPT

# Email Server (IMAP4)
$IPT -A tcp_inbound -p TCP -s 0/0 --destination-port 143 -j ACCEPT

# SSL Email Server (POP3)
$IPT -A tcp_inbound -p TCP -s 0/0 --destination-port 995 -j ACCEPT

# SSL Email Server (IMAP4)
$IPT -A tcp_inbound -p TCP -s 0/0 --destination-port 993 -j ACCEPT

# sshd
$IPT -A tcp_inbound -p TCP -s 0/0 --destination-port 22 -j ACCEPT

# ICQ File Transfers & Other Advanced Features
#
# ICQ supports a number of options beyond simple instant messaging.
# For those to function, the instant messaging system must allow
```

```
# new connections initiated from remote systems. This option will
# open a specified port range on the firewalled system.  The ICQ client
# on the firewalled system must also be configured to use the specified
# port range.

$IPT -A tcp_inbound -p TCP -s 0/0 --destination-port 5000:5100 -j ACCEPT

# MSN Messenger File Transfers
#
# Messenger supports file transfers.  For transfers initiated by
# remote systems to function, the system must allow
# new connections initiated from remote systems a specific port range.
# This option defaults to the port range 6891 through 6900.
# Unless the MSN Messenger client can be configured to specify any
# port range, don't change the default.

$IPT -A tcp_inbound -p TCP -s 0/0 --destination-port 6891:6900 -j ACCEPT

# User specified allowed UDP protocol
$IPT -A tcp_inbound -p TCP -s 0/0 --destination-port 32500:36000 -j ACCEPT



# Not matched, so return so it will be logged
$IPT -A tcp_inbound -p TCP -j RETURN

# tcp_outbound chain
#
# This chain is used with a private network to prevent forwarding for
# requests on specific protocols.  Applied to the FORWARD rule from
# the internal network.  Ends with an ACCEPT



# No match, so ACCEPT
$IPT -A tcp_outbound -p TCP -s 0/0 -j ACCEPT
```

```
#########################################################################
#
# INPUT Chain
#

echo "Process INPUT chain ..."

# Allow all on localhost interface
$IPT -A INPUT -p ALL -i $LO_IFACE -j ACCEPT

# Drop bad packets
$IPT -A INPUT -p ALL -j bad_packets

# DOCSIS compliant cable modems
# Some DOCSIS compliant cable modems send IGMP multicasts to find
# connected PCs.  The multicast packets have the destination address
# 224.0.0.1.  You can accept them.  If you choose to do so,
# Uncomment the rule to ACCEPT them and comment the rule to DROP
# them  The firewall will drop them here by default to avoid
# cluttering the log.  The firewall will drop all multicasts
# to the entire subnet (224.0.0.1) by default.  To only affect
# IGMP multicasts, change '-p ALL' to '-p 2'.  Of course,
# if they aren't accepted elsewhere, it will only ensure that
# multicasts on other protocols are logged.
# Drop them without logging.
$IPT -A INPUT -p ALL -d 224.0.0.1 -j DROP
# The rule to accept the packets.
# $IPT -A INPUT -p ALL -d 224.0.0.1 -j ACCEPT

# Rules for the private network (accessing gateway system itself)
$IPT -A INPUT -p ALL -i $LOCAL_IFACE -s $LOCAL_NET -j ACCEPT
$IPT -A INPUT -p ALL -i $LOCAL_IFACE -d $LOCAL_BCAST -j ACCEPT
```

```
# Inbound Internet Packet Rules

# Accept Established Connections
$IPT -A INPUT -p ALL -i $INET_IFACE -m state --state ESTABLISHED,RELATED \
     -j ACCEPT

# Route the rest to the appropriate user chain
$IPT -A INPUT -p TCP -i $INET_IFACE -j tcp_inbound
$IPT -A INPUT -p UDP -i $INET_IFACE -j udp_inbound
$IPT -A INPUT -p ICMP -i $INET_IFACE -j icmp_packets

# Drop without logging broadcasts that get this far.
# Cuts down on log clutter.
# Comment this line if testing new rules that impact
# broadcast protocols.
$IPT -A INPUT -m pkttype --pkt-type broadcast -j DROP

# Log packets that still don't match
$IPT -A INPUT -j LOG --log-prefix "fp=INPUT:99 a=DROP "

################################################################################
#
# FORWARD Chain
#

echo "Process FORWARD chain ..."

# Used if forwarding for a private network

# Drop bad packets
$IPT -A FORWARD -p ALL -j bad_packets

# Accept TCP packets we want to forward from internal sources
$IPT -A FORWARD -p tcp -i $LOCAL_IFACE -j tcp_outbound
```

```
# Accept UDP packets we want to forward from internal sources
$IPT -A FORWARD -p udp -i $LOCAL_IFACE -j udp_outbound

# If not blocked, accept any other packets from the internal interface
$IPT -A FORWARD -p ALL -i $LOCAL_IFACE -j ACCEPT

# Deal with responses from the internet
$IPT -A FORWARD -i $INET_IFACE -m state --state ESTABLISHED,RELATED \
     -j ACCEPT

# Port Forwarding is enabled, so accept forwarded traffic
$IPT -A FORWARD -p tcp -i $INET_IFACE --destination-port 8080 \
     --destination 192.168.1.50 -j ACCEPT

# Log packets that still don't match
$IPT -A FORWARD -j LOG --log-prefix "fp=FORWARD:99 a=DROP "

###############################################################################
#
# OUTPUT Chain
#

echo "Process OUTPUT chain ..."

# Generally trust the firewall on output

# However, invalid icmp packets need to be dropped
# to prevent a possible exploit.
$IPT -A OUTPUT -m state -p icmp --state INVALID -j DROP

# Localhost
$IPT -A OUTPUT -p ALL -s $LO_IP -j ACCEPT
$IPT -A OUTPUT -p ALL -o $LO_IFACE -j ACCEPT
```

```
# To internal network
$IPT -A OUTPUT -p ALL -s $LOCAL_IP -j ACCEPT
$IPT -A OUTPUT -p ALL -o $LOCAL_IFACE -j ACCEPT

# To internet
$IPT -A OUTPUT -p ALL -o $INET_IFACE -j ACCEPT

# Log packets that still don't match
$IPT -A OUTPUT -j LOG --log-prefix "fp=OUTPUT:99 a=DROP "

##############################################################################
#
# nat table
#
##############################################################################

# The nat table is where network address translation occurs if there
# is a private network.  If the gateway is connected to the Internet
# with a static IP, snat is used.  If the gateway has a dynamic address,
# masquerade must be used instead.  There is more overhead associated
# with masquerade, so snat is better when it can be used.
# The nat table has a builtin chain, PREROUTING, for dnat and redirects.
# Another, POSTROUTING, handles snat and masquerade.

echo "Load rules for nat table ..."

##############################################################################
#
# PREROUTING chain
#

# Port Forwarding
#
# Port forwarding forwards all traffic on a port or ports from
```

```
# the firewall to a computer on the internal LAN.  This can
# be required to support special situations.  For instance,
# this is the only way to support file transfers with an ICQ
# client on an internal computer.  It's also required if an internal
# system hosts a service such as a web server.  However, it's also
# a dangerous option.  It allows Internet computers access to
# your internal network.  Use it carefully and only if you're
# certain you know what you're doing.

$IPT -t nat -A PREROUTING -p tcp -i $INET_IFACE --destination-port 80:80 \
     -j DNAT --to-destination 192.168.1.50:8080

# This is a sample that will exempt a specific host from the transparent proxy
#$IPT -t nat -A PREROUTING -p tcp -s 192.168.1.50 --destination-port 80 \
#     -j RETURN
#$IPT -t nat -A PREROUTING -p tcp -s 192.168.1.50 --destination-port 443 \
#     -j RETURN

# Redirect HTTP for a transparent proxy
$IPT -t nat -A PREROUTING -p tcp --destination-port 80 \
     -j REDIRECT --to-ports 3128
# Redirect HTTPS for a transparent proxy - commented by default
# $IPT -t nat -A PREROUTING -p tcp --destination-port 443 \
#     -j REDIRECT --to-ports 3128

########################################################################
#
# POSTROUTING chain
#

$IPT -t nat -A POSTROUTING -o $INET_IFACE -j MASQUERADE

########################################################################
#
```

```
# mangle table
#
###############################################################################

# The mangle table is used to alter packets.  It can alter or mangle them in
# several ways.  For the purposes of this generator, we only use its ability
# to alter the TTL in packets.  However, it can be used to set netfilter
# mark values on specific packets.  Those marks could then be used in another
# table like filter, to limit activities associated with a specific host, for
# instance.  The TOS target can be used to set the Type of Service field in
# the IP header.  Note that the TTL target might not be included in the
# distribution on your system.  If it is not and you require it, you will
# have to add it.  That may require that you build from source.

echo "Load rules for mangle table ..."

# Set the TTL in outbound packets to the same consistent value.
# A value around 128 is a good value.  Do not set this too high as
# it will adversely affect your network.  It is also considered bad
# form on the Internet.
$IPT -t mangle -A OUTPUT -o $INET_IFACE -j TTL --ttl-set 128
```
