

Chapitre 8: Introduction aux réseaux

INF1070

Utilisation et administration des systèmes informatiques

Jean Privat & Alexandre Blondin Massé

Université du Québec à Montréal

Hiver 2019

Plan

- 1 Internet
- 2 Couche réseau
- 3 Couche transport
- 4 Communication sécurisée
- 5 Réseaux locaux

Réseaux

Exemple d'utilisation

- Afficher une page dans un navigateur
- Se connecter à un serveur avec `ssh`

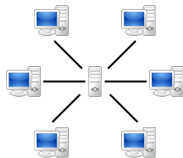
Principe des réseaux informatiques

- Des programmes **communiquent**
- Sur des ordinateurs **différents** (matériel et logiciel)
- Connectés via une **infrastructure**
- Communications régies par des **protocoles** (HTTP, SSH, etc.)

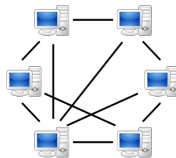
Difficultés du domaine

- Beaucoup d'acronymes
- Concepts imbriqués
- Nombreux détails techniques nécessaires

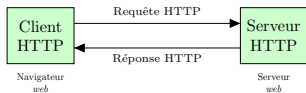
Représentation schématisée



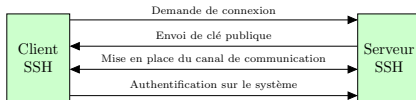
Client-serveur
(source: [Wikipedia](#))



Pair-à-pair
(source: [Wikipedia](#))



Navigateur-Serveur *web*



Client SSH-Serveur SSH

Le web et Internet

Le web = espace d'information

- Ensemble de ressources (hypertexte, images, sons, ...)
- Identifiées de façon unique (on va y revenir)
- Interconnectées par des hyperliens
- Aussi appelé la *toile*

Internet = réseau de réseaux

- Réseaux de fournisseurs d'accès (ISP, *internet service provider*):
Bell, Videotron, RISQ, CANARIE
- Réseaux d'entreprises et d'institutions
- Réseaux locaux personnels
- Réseaux cellulaires

Attention: Internet \neq le web

Internet

À qui appartient Internet?

Gouvernance

- Pas d'autorité centrale
- Réseau distribué
- Basé sur la coopération
- Who runs the Internet?

Trois joueurs majeurs

- ICANN (*Internet Corporation for Assigned Names and Numbers*)
→ Regroupe les acteurs techniques, gère les adresses IP, noms de domaine, numéros de port, ...
- ISOC (*Internet Society*)
→ Regroupe les utilisateurs généraux, garantit un développement ouvert, tenant compte des utilisateurs.
- IGF (*Internet Governance Forum*)
→ Créé par les Nations unies, impact politique.



Protocole

- Codifie la communication entre membres d'un réseau
- Définit le format et l'ordre des messages envoyés
- Définit les actions à faire lors de la réception de messages

Règles

- Requêtes déraisonnables ou malicieuses
- Problèmes de sécurité



- HTTP:



- HTTP: *Hypertext Transfer Protocol*, le protocole du web
- SSH:



- HTTP: *Hypertext Transfer Protocol*, le protocole du web
- SSH: *Secure Shell*, connexion sécurisée (chiffrée)
- IP:



- HTTP: *Hypertext Transfer Protocol*, le protocole du web
- SSH: *Secure Shell*, connexion sécurisée (chiffrée)
- IP: *Internet Protocol*, protocole principal d'internet
- SMTP:



- HTTP: *Hypertext Transfer Protocol*, le protocole du web
- SSH: *Secure Shell*, connexion sécurisée (chiffrée)
- IP: *Internet Protocol*, protocole principal d'internet
- SMTP: *Simple Mail Transfer Protocol* transfert de courriels
- DHCP:



- HTTP: *Hypertext Transfer Protocol*, le protocole du web
- SSH: *Secure Shell*, connexion sécurisée (chiffrée)
- IP: *Internet Protocol*, protocole principal d'internet
- SMTP: *Simple Mail Transfer Protocol* transfert de courriels
- DHCP: *Dynamic Host Configuration Protocol*, configuration automatique des paramètres internet d'une machine
- Ethernet: Commutation de paquets dans réseau local
- UDP:



- HTTP: *Hypertext Transfer Protocol*, le protocole du web
- SSH: *Secure Shell*, connexion sécurisée (chiffrée)
- IP: *Internet Protocol*, protocole principal d'internet
- SMTP: *Simple Mail Transfer Protocol* transfert de courriels
- DHCP: *Dynamic Host Configuration Protocol*, configuration automatique des paramètres internet d'une machine
- Ethernet: Commutation de paquets dans réseau local
- UDP: *User Datagram Protocol*, un autre protocole d'internet
- TCP:



- HTTP: *Hypertext Transfer Protocol*, le protocole du web
 - SSH: *Secure Shell*, connexion sécurisée (chiffrée)
 - IP: *Internet Protocol*, protocole principal d'internet
 - SMTP: *Simple Mail Transfer Protocol* transfert de courriels
 - DHCP: *Dynamic Host Configuration Protocol*, configuration automatique des paramètres internet d'une machine
 - Ethernet: Commutation de paquets dans réseau local
 - UDP: *User Datagram Protocol*, un autre protocole d'internet
 - TCP: *Transmission Control Protocol*, un autre protocole d'internet
- « **TCP/IP** » désigne abusivement les protocoles d'internet en général

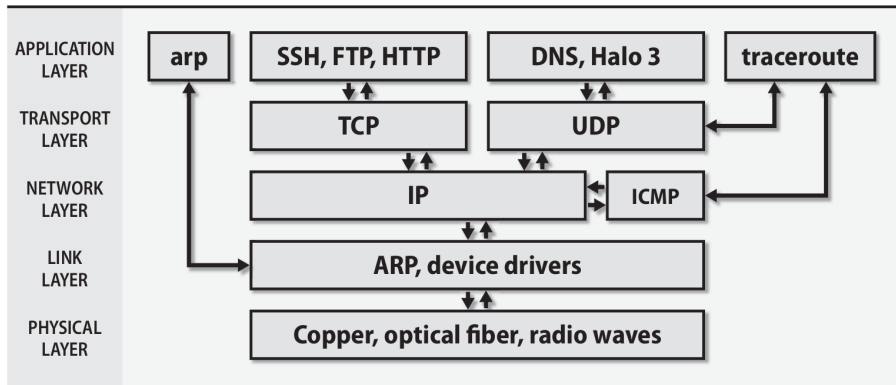


Figure 1: Tiré de [Unix and Linux System Administration Handbook](#)

Plus de détails dans INF3271 Téléinformatique

Couche physique et liaison

Transmission effective des signaux entre des machines adjacentes sur un même réseau

- Normes IEEE 802
- Ethernet (normes IEEE 802.3)
- Wi-Fi (normes IEEE 802.11)

Adresse MAC (*Media Access Control*)

- Appelée aussi **adresse physique**
- Associée à une **interface réseau**
- Ethernet: 48 bits, 6 groupes hexadécimaux. 5e:ff:56:a2:af:15
- Les 3 premiers octets désignent le fabriquant (OUI, *Organizationally Unique Identifier*)

```
$ ip -brief link show
```

```
wlp108s0    6a:10:d8:8e:f0:34
```

```
eth0        64:4b:f0:01:9a:ad
```

Couche réseau



Objectifs

- Livrer des paquets de données
- Sur Internet à travers les différents sous-réseaux
- Vers un hôte destination
- Seulement en se fiant à **son adresse IP**

Adresse IP

- Attribuée aux **interfaces réseaux** d'une machine connectée à Internet.
- Même idée qu'une adresse postale

La commande ping

- `ping` — vérifie la disponibilité d'un hôte
- Utilise le protocole ICMP
- Envoie une requête `ECHO_REQUEST` et attend une réponse `ECHO_RESPONSE`
- Utile pour vérifier la connexion
- Certaines machines ou routeurs bloquent les pings

```
$ ping wikipedia.org
```

```
PING wikipedia.org (208.80.154.224) 56(84) bytes of data.  
64 bytes from text-lb.eqiad.wikimedia.org (208.80.154.224):  
64 bytes from text-lb.eqiad.wikimedia.org (208.80.154.224):  
[...]
```

Adresse IP

IPv4 (1981)

- Codée sur **32 bits**
- Représentée par 4 chiffres de 0 à 255
- Exemple: 132.208.246.6
- Problème: pas assez d'adresses ($\approx 4G$)
- Solution actuelle: traduction d'adresse
(NAT = *network address translation*)

IPv6 (1998)

- Codée sur **128 bits**
- Représentée par 8 groupes de 4 hexadécimaux
- Exemple: 2001:0db8:0000:85a3:0000:0000:ac1f:8001
- Résoud différents problèmes (dont l'épuisement des adresses)
- Supporté dans la plupart des cas, mais complexe à intégrer



Hôte local (*localhost*)

Désigne la machine courante

Pas besoin d'interface réseau

- Nom de domaine: `localhost`
- IPv4: `127.0.0.1` (en fait `127.0.0.0/8`, c'est-à-dire `127.*.*.*`)
- IPv6: `::1` (c'est-à-dire `0:0:0:0:0:0:0:1`)

Réseau privé

Ces adresses ne peuvent être routées

On les utilise pour des réseaux locaux (LAN, *local area networks*)

- `10.0.0.0/8`, c'est-à-dire `10.*.*.*`
- `172.16.0.0/12`, c'est-à-dire de `172.16.*.*` à `172.31.*.*`
- `192.168.0.0/16`, c'est-à-dire `192.168.*.*`

Adresse IP d'une interface réseau

`ip` — information sur les interfaces réseau (extra)
(ou `ifconfig` déprécié)

```
$ ip -brief addr show
```

```
lo          UNKNOWN 127.0.0.1/8 ::1/128
```

```
enp0s25    DOWN
```

```
wlp4s0     UP          192.168.1.128/24 fe80::3479:c2c9:cde3:d789/
```

- `lo` (*loopback device*): interface virtuelle vers elle-même
- `enp0s25`: interface physique (`DOWN` = désactivée)
- `wlp4s0`: interface sans-fil (`UP` = activée)
- Adresse IPv4: 192.168.1.128
- Adresse IPv6: fe80::3479:c2c9:cde3:d789



- DNS = Domain Name System
- Établir le lien entre **nom** et une **adresse IP** (et inversement)
- Objectif: rendre les URL plus lisibles
- Exemples: ageei-uqam.slack.com et info.uqam.ca

host — conversion entre nom de domaine et adresse IP (extra)

```
$ host uqam.ca
```

```
uqam.ca has address 132.208.246.6
```

```
$ host labunix.uqam.ca
```

```
labunix.uqam.ca has address 132.208.132.48
```

```
$ host java.labunix.uqam.ca
```

```
java.labunix.uqam.ca has address 132.208.132.52
```

Autre outil pour utiliser DNS: **dig** (extra)

Hiérarchie des noms de domaine

- **Fédération** de registres gérée par l'IANA (*Internet Assigned Numbers Authority*)
- Maintient des registres par organisations (Verisign, CIRA, etc.)
- Données réparties dans une **hiérarchie de serveurs** DNS

TLD = *Top level domain*

- Noms au premier niveau
- Exemples: ca, com, org, edu, gov, etc.

Exemple (info.uqam.ca)

- root: géré par l'IANA
- ca: géré par la CIRA (*Canadian Internet Registration Authority*)
- uqam: géré par l'UQAM
- info: géré par le département d'informatique

DHCP = *Dynamic Host Configuration Protocol*

- Beaucoup d'utilisateurs se connectent à un même réseau
- Pas tout le monde sait configurer une adresse IP manuellement
- Le nombre d'adresses IP disponibles est limité
- Les clients ne sont pas toujours tous connectés

Solution

- Attribuer une adresse IP de façon dynamique
- Configurer automatiquement le masque réseau, la passerelle et le serveur de noms

Qui l'utilise?

- Le fournisseur d'accès (*ISP = Internet service provider*)
- L'organisation ou l'entreprise (exemple, l'UQAM)
- Votre routeur à la maison

Suivre des paquets

`traceroute`: affiche le chemin suivi par des paquets jusqu'à un hôte

```
$ traceroute google.com
```

```
traceroute to google.com (172.217.13.142), 30 hops max,  
60 byte packets
```

```
[...]
```

```
12 yul02s05-in-f14.1e100.net (172.217.13.142) 4.248 ms  
4.284 ms 3.899 ms
```

Certaines machines ou routeurs bloquent traceroute (*timeout*)



Stratégie de sondage

- Sonde par essai et erreur
- Jusqu'à destination
- Ou jusqu'à ce que le nombre max de sauts soit atteint
- Le comportement est configurable

Options de traceroute

- -m, --max-hops le nombre maximum de sauts
- -p, --port spécifie le port
- -I, --icmp sonde avec ICMP ECHO
- -T, --tcp sonde avec TCP SYN
- -U, --udp sonde avec UDP

Exemple

```
$ sudo traceroute -T wikipedia.org
traceroute to wikipedia.org (208.80.154.224), 30 hops max,
 60 byte packets
 1  _gateway (132.208.137.1) 0.232 ms 0.207 ms 0.365 ms
 2  132.208.2.81 (132.208.2.81) 0.439 ms 0.496 ms 0.531 ms
 3  132.208.2.126 (132.208.2.126) 0.523 ms 0.565 ms *
 [...]
```

```
15  wikimedia-ic-308845-ash-b1.c.telial.net
    (80.239.132.226) 23.335 ms 26.381 ms 26.414 ms
16  text-lb.eqiad.wikimedia.org (208.80.154.224)
    26.066 ms 25.745 ms 25.839 ms
```

Couche transport

Les protocoles TCP et UDP

TCP = *Transmission control protocol*

- Transmission avec connexion (*handshaking*)
- Flots d'octets (*stream*)
- Fiable contre: perte de paquets, duplication de données, mauvais ordonnancement, congestion, *timeout*
- Quitte à ce que ce soit plus lent

UDP = *User datagram protocol*

- Transmission sans connexion (unidirectionnelle)
- Paquets de données (*datagram*)
- Moins fiable (pas les garanties de TCP)
- Plus rapide
- Utilisé pour du temps réel
- Ou dans les réseaux qu'on sait fiables



- Point de communication
- Associé à un **processus** sur une machine
- Identifié par un **numéro**
- Utilisé en particulier par TCP et UDP

Ports connus (*well-known ports*), ou **ports du système**

- De 1 à 1023, liste maintenue par IANA
- Sous Unix, nécessite les droits root pour être utilisés

Ports enregistrés

- De 1024 à 49151, liste maintenue par IANA

Ports éphémères, ou **ports dynamiques**

- De 49152 à 65535
- Souvent attribués par le système pour les clients

Quiz sur les numéros de port



- 80 sur TCP:

Quiz sur les numéros de port



- 80 sur TCP: HTTP
- 22 sur TCP:

Quiz sur les numéros de port



- 80 sur TCP: HTTP
- 22 sur TCP: SSH
- 443 sur TCP:

Quiz sur les numéros de port



- 80 sur TCP: HTTP
- 22 sur TCP: SSH
- 443 sur TCP: HTTP sur TLS (HTTPS)
- 25 sur TCP:

Quiz sur les numéros de port



- 80 sur TCP: HTTP
- 22 sur TCP: SSH
- 443 sur TCP: HTTP sur TLS (HTTPS)
- 25 sur TCP: SMTP
- 53 sur TCP et UDP:

Quiz sur les numéros de port



- 80 sur TCP: HTTP
- 22 sur TCP: SSH
- 443 sur TCP: HTTP sur TLS (HTTPS)
- 25 sur TCP: SMTP
- 53 sur TCP et UDP: DNS
- 67 et 68 sur UDP:

Quiz sur les numéros de port



- 80 sur TCP: HTTP
- 22 sur TCP: SSH
- 443 sur TCP: HTTP sur TLS (HTTPS)
- 25 sur TCP: SMTP
- 53 sur TCP et UDP: DNS
- 67 et 68 sur UDP: DHCP

Noms de services

Dans de nombreux outils, un nom du service facile à mémoriser peut être utilisé à la place du numéros du port

`/etc/services` fait la correspondance entre des noms et numéros

Sockets (prises)

C'est quoi?

- *Socket* = « interface de connexion » ou « prise réseau »
- Point d'entrée/sortie de processus avec le réseau
- Abstraction utilisée par les programmes
- C'est rare de les manipuler directement

Composante d'une *socket* internet TCP ou UDP

- Une adresse IP
- Un numéro de port
- Un protocole de transport (TCP, UDP)

D'autres sockets ?

- Protocoles réseaux plus rares
- Mécanisme de communication inter-processus (sans réseau)

Information sur les *sockets*

`ss` — affiche des informations sur les *sockets* (extra)
(ou `netstat` déprécié)

- `-a` affiche toutes les *sockets*
- `-l` affiche seulement les *sockets* en mode « écoute » (`LISTEN`)
- `-m` affiche l'utilisation mémoire des *sockets*
- `-t` affiche les *sockets* TCP
- `-u` affiche les *sockets* UDP

```
$ ss -ta
```

State	Local	Address:Port	Peer	Address:Port
LISTEN		0.0.0.0:ssh		0.0.0.0:*
LISTEN		127.0.0.1:ipp		0.0.0.0:*
ESTAB	192.168.1.128	56434	173.194.66.188	https
ESTAB	192.168.1.128	46174	151.101.193.69	https
[...]				
ESTAB	192.168.1.128	40676	172.217.13.134	https

La commande netcat

`nc` — crée une socket (extra)

- `-l` pour se mettre en mode « écoute »

```
$ nc [options] hôte port
```

Exemple:

```
# Terminal 1
```

```
# Crée une socket qui écoute sur le port 3333
```

```
$ nc -l 3333 | lolcat
```

```
# Terminal 2
```

```
# Adresse IP est 192.168.1.128
```

```
# Transmet du texte via une socket
```

```
$ ls | nc 192.168.1.128 3333
```

Communication sécurisée

Sécurité de l'information

Contexte

- Le réseau est un environnement potentiellement **hostile**
- Toute suite d'octets peut être lue, interceptée, trafiquée
- Il faut des **objectifs** de sécurité
- Et des **moyens** pour les atteindre

Cours connexes

On va survoler des concepts de base

Pour en savoir plus:

- INF4471 Introduction à la sécurité informatique
- INF600C Sécurité des logiciels et exploitation de vulnérabilités

Objectif: critères de sensibilité (triade CIA)



Confidentialité (*Confidentiality*)

- L'information n'est pas disponible aux acteurs non autorisés
- Un adversaire ne peut lire ou comprendre l'information

Intégrité (*Integrity*)

- L'information est exacte et complète
- Un adversaire ne peut altérer ou injecter de l'information

Disponibilité (*Availability*)

- L'information est effectivement disponible aux acteurs autorisés
- Un serveur éteint est sécuritaire mais pas très utile



Identification

- Assertion de l'identité
- « Je suis John Doe »

Authentification

- Vérification de l'identité d'un acteur identifié
- « Pour preuve, mon PIN est 1234 »

Autorisation

- Détermine l'accès aux ressources d'un acteur authentifié
- « Je veux retirer 500\$ du compte 123456789 »

Moyen: Cryptographie

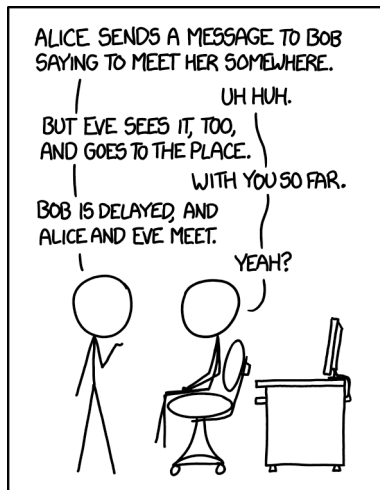


- **Cryptologie**: la science du secret, au croisement de l'informatique et des mathématiques
- **Cryptographie**: science des codes secrets permettant de protéger la confidentialité de messages
- **Cryptanalyse**: science des attaques pour briser ces codes secrets

Un peu de terminologie

- **Chiffrement** (*encryption*)
Transformation d'un message pour qu'il soit illisible
- **Déchiffrement** (*decryption*)
Transformation inverse permettant de récupérer le message
- **Clé**
Information permettant de (dé-)chiffrer un message
- **Chiffre** (*cipher*)
Algorithme de chiffrement/déchiffrement

Alice, Bob, Ève et compagnie



I'VE DISCOVERED A WAY TO GET COMPUTER
SCIENTISTS TO LISTEN TO ANY BORING STORY.

Source: <https://xkcd.com/1323/>

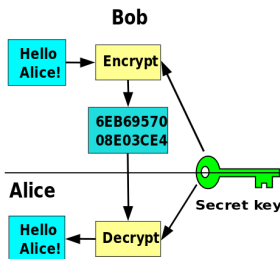
Chiffrement symétrique



- La même clé est connue de Alice et de Bob
- Elle est utilisée pour chiffrer et déchiffrer

Hypothèses mathématiques

- Facile de chiffrer/déchiffrer si on connaît la clé
- Difficile de chiffrer/déchiffrer sans la clé

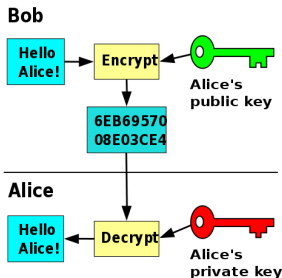


Source: [Wikipedia](#)

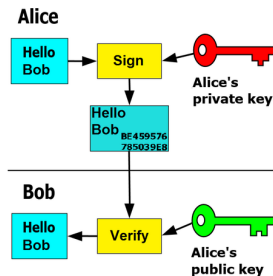
Chiffrement asymétrique



- Chacun possède une clé **privée** et une clé **publique**
- La clé privée ne doit pas être divulguée
- La clé publique peut être diffusée sans restriction
- Mêmes hypothèses sur la facilité/difficulté de chiffrer/déchiffrer
- Peut être utilisé pour des **signatures électroniques**

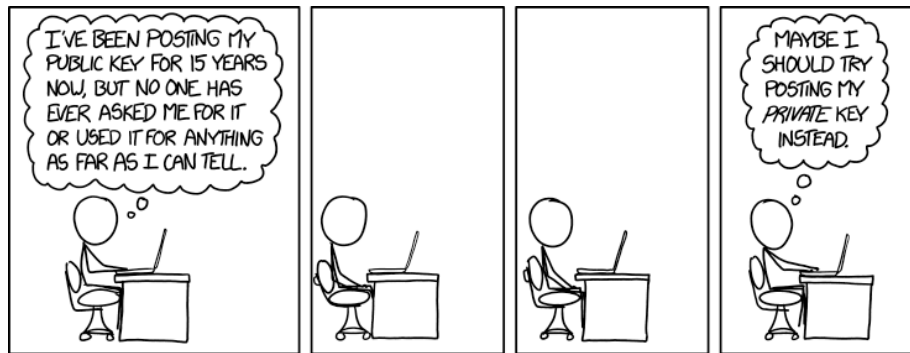


Source: [Wikipedia](#)



Source: [Wikipedia](#)

Public Key



Source: <https://xkcd.com/1553/> (2015)

HTTPS = HTTP sur TLS

- HTTPS = *Hypertext transfer protocol secure*
- TLS = *Transport layer security*
Anciennement: SSL = *Secure sockets layer*
- Port par défaut: 443
- Indique au navigateur de chiffrer les échanges d'information:
URL, requêtes, en-tête, *cookies*, réponses, etc.

Objectifs de TLS

- Confidentialité: les messages sont illisibles par des tiers
- Intégrité: les messages ne peuvent être altérés (*MITM attack*)
- Authentification: l'identité du serveur est vérifiée

Attention La communication n'est pas invisible pour autant
« de 2h à 3h du matin, l'IP 132.208.132.52 a échangé avec le serveur
`https://xn--jinf1070-ff7e.xyz/` pour 3Go de données. »

Certificat électronique

- Le certificat permet l'authentification du serveur
- Il est fourni par le serveur lors de la connexion
- Il est vérifié par le client (navigateur)
- Pour plus de détails: voir le cadenas (🔒) à côté de l'URL

Autorités de certification

- Délivre les certificats (service commercial)
- Confirme l'identité du propriétaire (audits)
- Fournit une preuve de la validité du certificat (signature)
- Exemples: [Let's Encrypt](#), [IdenTrust](#), [Comodo](#), [DigiCert](#), etc.

Chaîne de confiance

- Le certificat du serveur est signé par un autre certificat
- Ainsi de suite jusqu'à un certificat racine
- Les certificats racines des autorités sont connus des navigateurs
- L'utilisateur fait confiance au navigateur

Réseaux locaux

Réseau local

- LAN = Local area network

Technologie

- Ensemble d'ordinateurs
- Connectés par une infrastructure réseau commune
- Ethernet et Wi-Fi

Besoins

- Permettre aux machines du réseau de communiquer
- Permettre l'accès à internet
- Protéger les machines des attaques de l'extérieur

Routeur

- Passerelle entre deux réseaux
- Utilise une table de routage pour distribuer les paquets

Routeur et +

Les routeurs personnels modernes ont plus de services

- Point d'accès Wi-Fi: pour les clients sans fil
- Serveur DHCP: pour attribuer des adresses IP aux clients
- Serveur DNS: pour cacher et relayer les requêtes DNS
- Pare-feu: pour appliquer une politique de sécurité
- NAT: pour router les adresses privées

Pare-feu (*firewall*)

Applique une politique de sécurité réseau

Filtrage

- Sens des paquets
- Origine ou destination des paquets (IP, protocole, port)
- Données des paquets (applicatif)

Contrôle et surveillance

- Surveillance d'activité

Traduction d'adresse réseau (NAT, *network address translation*)

- Permet l'accès au réseau
- Cache l'adresse réelle des clients
- Maintient l'état des connexions

Pare-feu système

Le système d'exploitation a son propre pare-feu

`iptables` outil d'administration du pare-feu (Linux)

- Voir <https://netfilter.org/>

`ufw` (*uncomplicated firewall*) gestion simplifiée du pare-feu (extra)

- `enable/disable` — activer/désactiver le pare-feu
- `status` — afficher l'état d'un pare-feu
- `allow/deny/reject/limit` — ajouter une règle
- `delete` — supprimer une règle

```
$ ufw allow in http      # Requête HTTP entrante permise
$ ufw reject out smtp    # Pas d'envoi de courriel
$ ufw allow 80/tcp       # Permettre TCP sur le port 80
$ ufw deny 53            # Interdire les accès au port 53
```

Exemple

Permettre l'accès à Apache:

```
$ sudo ufw enable
```

```
$ sudo ufw app list
```

Available applications:

Apache

Apache Full

OpenSSH

...

```
$ sudo ufw allow 'Apache Full'
```

```
$ sudo ufw status
```

Status: active

To	Action	From
--	-----	----
Apache Full	ALLOW	Anywhere
Apache Full (v6)	ALLOW	Anywhere (v6)

Note: sous Debian il faut utiliser « WWW Full » par exemple

Tunnels et réseau privé virtuel

La communication entre deux ordinateurs sur internet peut être limitée

- Environnement hostile
- Pare-feux agressifs
- Adresses non-routables

Tunnel

- Établissement d'un canal de communication sécurisé
- Utilisation de ce canal pour faire passer un autre protocole

Réseau privé virtuel (VPN)

- Étendre un réseau privé sur un réseau public
 - Transparent pour les applications
- Elles communiquent comme sur un seul réseau privé
- Peut cacher l'identité des machines (NAT)

Tunnel SSH: redirection de port

- D'un côté: un réseau local inaccessible de l'extérieur
 - De l'autre côté: un réseau distant inaccessible de l'extérieur
- Sauf une machine distante `example.com` accessible par ssh

Accéder à un service distant à partir du réseau local

- ssh ouvre un port `portlocal` sur la machine locale
- Qui est redirigé vers un port `portdistant`
- Sur une machine distante `hotedistant`

```
$ ssh -L portlocal:hotedistant:portdistant serveur.com
```

Accéder à un service local à partir du réseau distant

- ssh ouvre un port `portdistant` sur la machine distante
- Qui est redirigé vers un port `portlocal`
- Sur une machine locale `hotelocal`

```
$ ssh -R portdistant:hotelocal:portlocal example.com
```

Exemple complet

- Notre machine n'est pas accessible de l'extérieur !
- Comment accéder à notre Apache depuis la machine java ?

```
$ ssh -R 8888:localhost:80 java.labunix.uqam.ca  
$ curl http://localhost:8888
```

Attention

- La première ligne est exécutée sur notre machine
→ Le premier « localhost » désigne notre machine
- La seconde ligne est exécutée sur java
→ Le second « localhost » désigne la machine java

Pièce en un acte

Personnages

- apache: serveur HTTP sur notre machine locale
- ssh: client ssh sur notre machine locale
- sshd: serveur ssh sur la machine java distante
- curl: client HTTP sur la machine java distante

Acte 1, scène 1 (ssh, sshd)

```
$ ssh -R 8888:localhost:80 java.labunix.uqam.ca
```

- ssh/sshd: crée un tunnel entre les deux machines
- sshd: ouvre un socket sur java, port 8888
- sshd: ouvre un shell interactif pour l'utilisateur

Pièce en un acte (suite)

Acte 1, scène 2 (ssh, sshd, curl, apache)

```
$ curl http://localhost:8888
```

- curl: se connecte au port 8888 de java et écrit la requête HTTP
- sshd: reçoit la requête et l'envoie à ssh via le tunnel
- ssh: se connecte sur notre machine, port 80 et copie la requête
- apache: lit la requête et envoie la réponse (à ssh)
- ssh: passe la réponse dans le tunnel (à sshd)
- sshd: passe la réponse (à curl)
- curl: affiche la réponse à l'écran

Redirection d'une application graphique

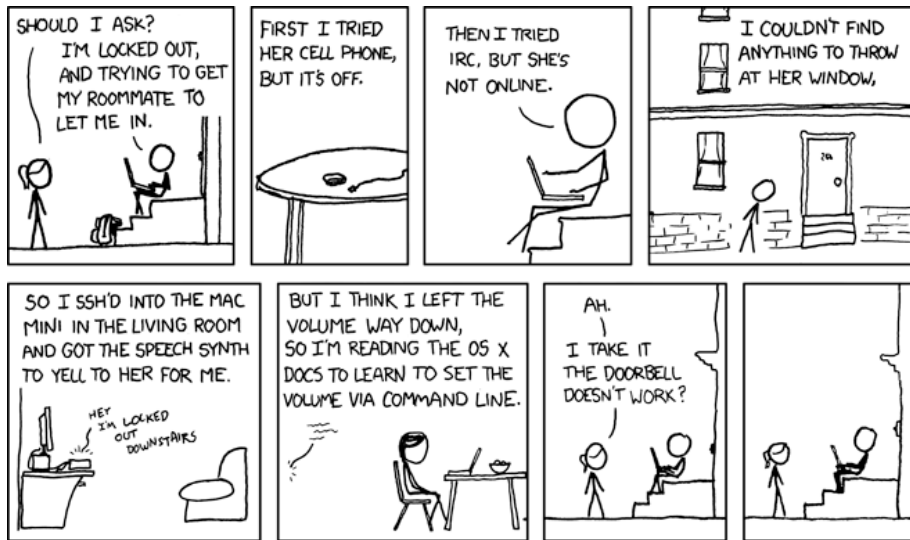


```
$ ssh -X blondin_al@java.labunix.uqam.ca
Password:
$ cd Pictures/
$ display debian.png
$ gedit
```

Attention!

- Possiblement lent
 - Plus ou moins déprécié
 - La machine distante a un accès total à votre interface
- capture d'écran `import -window root ecran.png`
- capture clavier `xinput test-xi2 --root`

I'm an Idiot



Source: <https://xkcd.com/530/> (2009)