



Pr. Youness KHOUREDIFI, PhD en Informatique  
Professeur à la Faculté Polydisciplinaire – Khouribga –  
Université Sultan Moulay Slimane – Béni Mellal –  
Consultant IT : SQL 2016 Database Administration,  
Core Infrastructure 2016, Azure Solutions Architect  
Expert, Data Analyst Associate, Ingénieur DevOps.  
[y.khourdifi@usms.ma](mailto:y.khourdifi@usms.ma)

# ADMINISTRATION RÉSEAUX



# Règle du cours

- ❑ Respect de l'horaire
- ❑ Respect des deadlines de remises des comptes rendus  
Respect des deadlines de remises des projets

# Systeme de notation

- ❑ Contrôles continus → 40%
- ❑ Travaux pratiques → 30%
- ❑ Projets → 20%
- ❑ Assiduité et participation → 10%

# Liste des Projets

- Liste des Projets

# L'environnement de travail

- ❑ Préparation d'un ordinateur pour la virtualisation
  - ❑ VirtualBox
- ❑ Installation du système d'exploitation Linux sur un ordinateur virtuel
  - ❑ Ubuntu 14.04.3 (20.04) LTS

# OBJECTIFS DU COURS

6

- Installer et administrer des systèmes et des services réseaux
- Mettre en œuvre une politique de sauvegarde
- Auditer la sécurité des systèmes informatiques
- Veiller à appliquer la politique de sécurité sur l'infrastructure SI
- Gérer et résoudre les incidents techniques sur le réseau et le système
- Gérer les postes de travail et accès des utilisateurs
- Intégrer des solutions dans un système d'information
- Superviser l'infrastructure
- Mener des nouveaux projets pour le SI

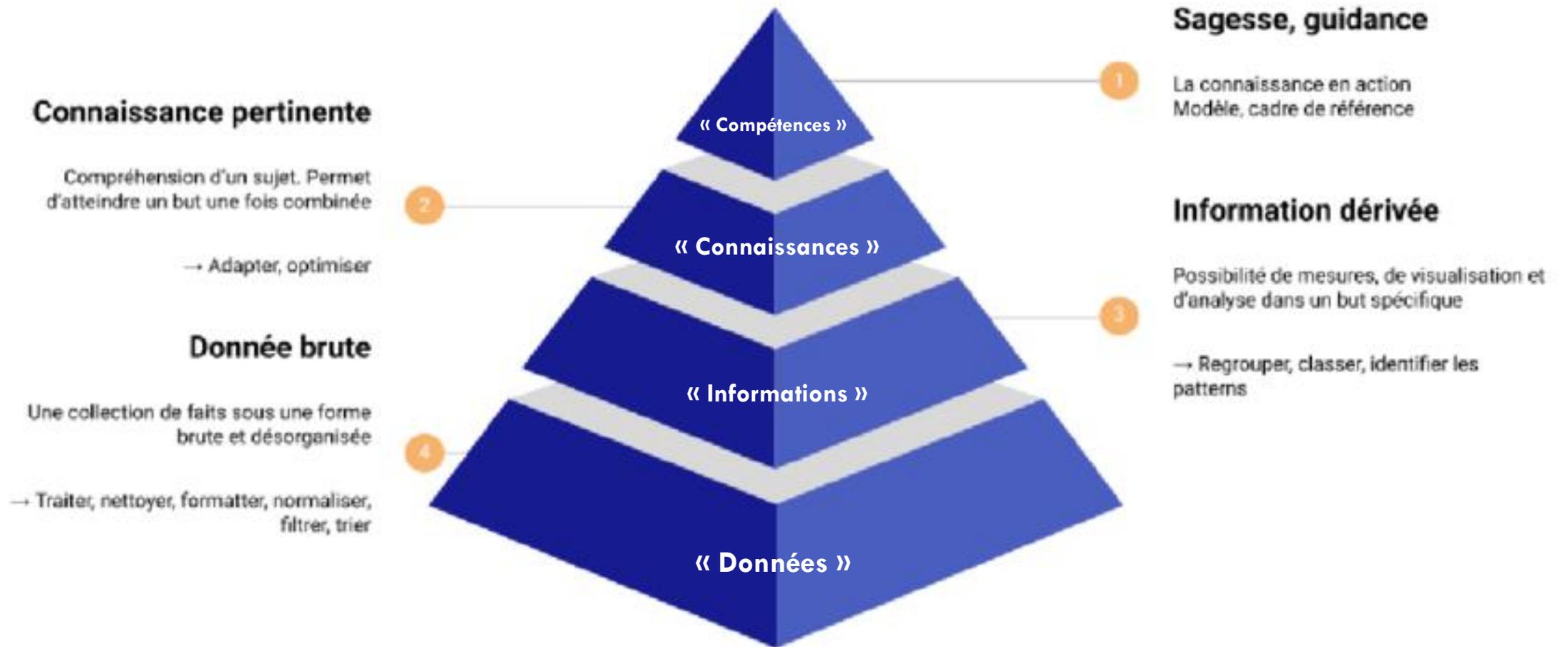
- Introduction à l'administration réseaux
- Modèle OSI et TCP/IP
- Administration des Vlans
- Translation des adresses réseaux (NAT)
- Administration des ACLs
- Administration de protocoles et services réseaux
- Administration de sauvegardes et Restaurations
- Supervision réseau
- Virtualisation système et réseau

**Chapitre**

**Introduction**

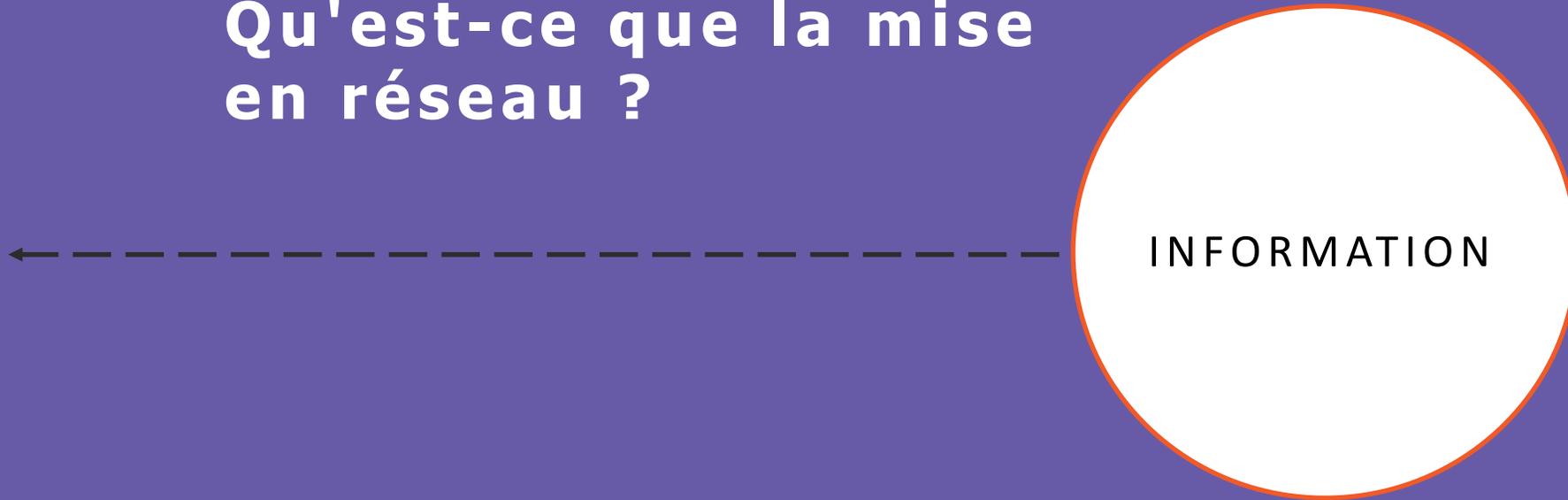
# I. 1. Introduction aux Réseaux :

## Pyramide de la connaissance (Rainer & al., 2009) :



# I. 1. Introduction aux Réseaux :

Qu'est-ce que la mise en réseau ?



Le transfert d'informations entre deux ou plusieurs parties.

## I. 1. Introduction aux Réseaux :

# Circuler l'information en parlant



Si nous pensons à cet exercice de discussion, nous transférons réellement des informations. Le type à gauche dit : "Quelle heure est-il ? Le type de droite répond, il est 14 heures. Excellent, disséquons un peu cela et voyons ce qui se passe pour pouvoir utiliser cela plus tard afin de mieux comprendre comment les réseaux de données fonctionnent également. Pour que la personne à gauche puisse dire qu'il est 14 heures, il lui faut un endroit pour projeter cette information. Heureusement, nous vivons sur Terre, et sur Terre nous avons cette fine couche d'air qui enveloppe l'extérieur de la Terre, et elle nous donne la capacité de survivre en tant que créatures vivantes, et si nous avons des cordes vocales, elle nous donne la capacité de faire vibrer cet air selon un modèle spécifique.

## I. 1. Introduction aux Réseaux :

# Circuler l'information en parlant



LANGUAGE: Français

CONCEPT: Temps

Maintenant, nous pouvons faire vibrer cet air selon toutes sortes de motifs, ces motifs sont peut-être dénués de sens, mais les vibrations des motifs de l'air de ma langue ici sont en fait très importantes. Parce que ce que je fais, c'est que je formule le concept d'il est 14 heures, qui est l'heure, et j'utilise la langue française pour le faire. Eh bien, la langue française parlée n'est rien d'autre qu'une vibration précise des ondes.

# I. 1. Introduction aux Réseaux :

## Protocole :

Si nous regardons notre protocole. Le protocole est un ensemble de règles utilisées pour transférer des informations entre deux personnes ou appareils. On parle ici spécifiquement des protocoles de communication de données.

Donc, lorsque nous parlons de protocoles de communication de données, nous parlons d'un ensemble de règles. Et si nous y réfléchissons, nous avons un protocole que nous utilisons chaque fois que nous composons un numéro de téléphone. Et ce protocole, si nous ne le suivons pas précisément, notre appel téléphonique ne fonctionnera pas, et nous ne pourrons pas nous connecter à la personne ayant l'adresse globale que nous essayons d'atteindre. Tout au long de notre apprentissage de la mise en réseau des données, nous allons apprendre les protocoles que nous utilisons pour communiquer.

# I. 1. Introduction aux Réseaux :

## Protocoles réseau

SSH (Secure Shell)  
802.1Q SFTP HTTP RSTP TCP  
IP v6 (Internet Protocol v6)  
TFTP EIGRP SMTP ICMP BGP OSPF  
IP (Internet Protocol) PPP RIP Ethernet POP3  
UDP FTP (File Transfer Protocol) SSL (Secure Sockets Layer)  
NTP HSRP

Ces protocoles peuvent se présenter sous toutes sortes de formes, ils ont toutes sortes de noms, ce sont généralement des abréviations de trois ou quatre lettres, parfois ce sont des mots entiers comme Ethernet, Ethernet est un protocole que nous utilisons pour communiquer. SFTP, protocole de transfert de fichiers sécurisé, HTTP et HTTPS sont probablement ceux que vous connaissez le mieux. Vous avez peut-être entendu parler d'IP et de TCP, et d'UDP et de BGP, nous allons couvrir tous ces protocoles, et à la fin de ce cours, vous allez connaître les détails de ce que font tous ces protocoles, ce qui est vraiment surprenant car il y a une quantité incroyable de détails à apprendre dans tous ces protocoles, mais ce qui est bien, c'est que lorsque vous regardez ces protocoles individuellement, ils ne sont pas si complexes.

# I. 1. Introduction aux Réseaux :

## Services réseau

Dans le cas d'un réseau informatique, un service réseau est une application exécutée depuis la couche d'application réseau et au-dessus. Il fournit des capacités de stockage, de manipulation, de présentation, de communication ou d'autres services qui sont souvent mises en œuvre en utilisant une architecture client-serveur ou pair à pair basée sur un protocole de communication de la couche « application » du modèle OSI.

# I. 1. Introduction aux Réseaux :

## Modèle de communication

- ❑ Un modèle de communication en couches organise les communications entre les hôtes sous forme d'enveloppement hiérarchique.
- ❑ Chaque couche remplit une fonction spécifique à la communication et dispose de caractéristiques propres.
- ❑ Chaque couche utilise un protocole, soit un ensemble de règles de communication.

# I. 1. Introduction aux Réseaux :

## Modèles inter-réseaux

Comprendre qu'un modèle de référence est un modèle conceptuel de la façon dont les communications devraient avoir lieu. Il traite tous les processus requis pour une communication efficace et les divise en groupes logiques appelés couches. Lorsqu'un système de communication est conçu de cette manière, il est connu sous le nom d'architecture hiérarchique ou en couches.

# I. 1. Introduction aux Réseaux :

## Modèles inter-réseaux

**Voici une liste des avantages les plus importants pour l'utilisation d'un modèle de référence:**

- Il divise le processus de communication réseau en composants plus petits et plus simples, facilitant le développement, la conception et le dépannage des composants.
- Il permet le développement de plusieurs fournisseurs grâce à la standardisation des composants réseau.
- Il encourage la normalisation de l'industrie en définissant clairement les fonctions qui se produisent à chaque couche du modèle.
- Il permet à différents types de matériels et de logiciels de communication.
- Il empêche les changements dans une couche d'affecter d'autres couches pour accélérer le développement.

# I. 1. Introduction aux Réseaux :

## Modèles inter-réseaux : Le Modèle OSI

OSI est un modèle logique, pas physique. Il s'agit essentiellement d'un ensemble de directives que les développeurs peuvent utiliser pour créer et implémenter des applications à exécuter sur un réseau. Il fournit également un cadre pour la création et l'implémentation de normes de réseau, de dispositifs et de schémas d'interconnexion de réseaux.

**L'OSI a sept couches différentes, divisées en deux groupes.** Les trois couches supérieures définissent la manière dont les applications dans les stations terminales communiqueront entre elles ainsi qu'avec les utilisateurs. Les quatre couches inférieures définissent la manière dont les données sont transmises de bout en bout.

# I. 1. Introduction aux Réseaux :

## Modèles inter-réseaux : Le Modèle OSI

TCP/IP Model	OSI Model	
Accès réseau	Physique	Wires/Radio/Glass
	Liaison de données	Ethernet
Internet	Réseau	Internet Protocol (IP)
Transport	Transport	TCP
Application	Session	-
	Présentation	-
	Application	Website/Email

# I. 1. Introduction aux Réseaux :

## Modèles inter-réseaux : Le Modèle OSI

### La Couche d'application :

La couche Application fonctionne en tant qu'interface entre le programme d'application réel et la couche suivante en fournissant à l'application des moyens d'envoyer des informations à travers la pile de protocoles.

L'identification et la confirmation de la disponibilité du partenaire de communication et la vérification des ressources requises pour permettre le type de communication spécifié ont également lieu au niveau de la couche Application.

# I. 1. Introduction aux Réseaux :

## Modèles inter-réseaux : Le Modèle OSI

### La Couche d'application :

Les applications sont divisées en plusieurs catégories on cite par exemple:

- Transferts de fichiers
- Email
- Activation de l'accès à distance
- Activités de gestion de réseau
- Processus client / serveur
- Emplacement de l'information

# I. 1. Introduction aux Réseaux :

## Modèles inter-réseaux : Le Modèle OSI

### La couche présentation :

La couche Présentation tire son nom de son objectif: elle présente des données à la couche Application et responsable de la traduction des données et de la mise en forme du code.

Les ordinateurs sont configurés pour recevoir ces données formatées de manière générique, puis les reformater dans leur état natif pour les lire. Un exemple de ce type de service de traduction se produit lors de la conversion d'anciennes données EBCDIC (Code d'échange décimal codé binaire étendu) en ASCII, le code standard américain pour l'échange d'informations (souvent prononcé "askee").

Il suffit donc de se rappeler qu'en fournissant des services de traduction, la couche Présentation garantit que les données transférées à partir de la couche Application d'un système peuvent être lues par la couche Application d'une autre.

# I. 1. Introduction aux Réseaux :

## Modèles inter-réseaux : Le Modèle OSI

### La couche présentation :

Dans cet esprit, il s'ensuit que l'OSI inclurait des protocoles qui définissent comment les données standard doivent être formatées, ainsi des fonctions clés telles que la compression de données, la décompression, le cryptage et le décryptage sont également associées à cette couche. Certaines normes de la couche Présentation sont également impliquées dans les opérations multimédias.

# I. 1. Introduction aux Réseaux :

## Modèles inter-réseaux : Le Modèle OSI

### La couche Session :

La couche Session est responsable de la configuration, de la gestion et du démontage des sessions entre les entités de la couche Présentation et de la séparation des données utilisateur. Le contrôle de la boîte de dialogue entre les périphériques se produit également sur cette couche.

La communication entre les différentes applications des hôtes de la couche Session, d'un client à un serveur, est coordonnée et organisée selon trois modes différents: **simplex, half-duplex et full-duplex**:

- ❑ Simplex est une simple communication à sens unique, un peu comme dire quelque chose et ne pas avoir de réponse.
- ❑ La semi-duplex est une communication bidirectionnelle réelle, mais elle peut avoir lieu dans une seule direction à la fois, empêchant l'interruption du dispositif de transmission.
- ❑ Le full-duplex est exactement comme une véritable conversation où les appareils peuvent transmettre et recevoir en même temps.

# I. 1. Introduction aux Réseaux :

## Modèles inter-réseaux : Le Modèle OSI

### La couche transport :

La couche Transport segmente et réassemble les données en un seul flux de données. Les services situés à cette couche prennent toutes les diverses données reçues des applications de couche supérieure, puis les combinent dans le même flux de données concis. Ces protocoles fournissent des services de transport de données de bout en bout et peuvent établir une connexion logique entre l'hôte émetteur et l'hôte de destination sur un inter-réseau.



**Une paire de protocoles bien connus appelés TCP et UDP font partie intégrante de cette couche**

# I. 1. Introduction aux Réseaux :

## Modèles inter-réseaux : Le Modèle OSI

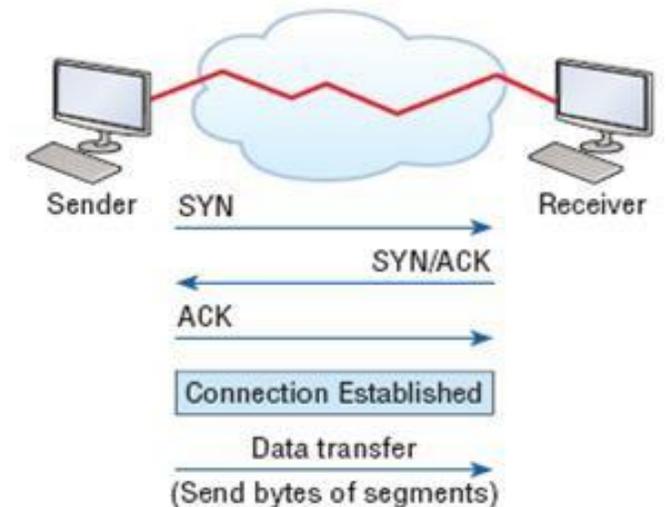
### La couche transport :

La couche de transport peut être sans connexion ou orientée connexion.

La session de communication orientée connexion avec un périphérique distant est connue sous le nom de **call setup** ou a (**Three-Way Handshake**).

Une fois ce processus terminé, le transfert de données se produit, et quand il est terminé, une terminaison d'appel a lieu pour démonter le circuit virtuel.

La figure ci-contre illustre une session fiable typique entre les systèmes émetteurs et récepteurs:



# I. 1. Introduction aux Réseaux :

## Modèles inter-réseaux : Le Modèle OSI

### La couche réseau :

La couche réseau, ou couche 3, gère l'adressage du périphérique, suit l'emplacement des périphériques sur le réseau et détermine la meilleure façon de déplacer les données. Cela signifie qu'il appartient à la couche réseau de transporter le trafic entre les périphériques qui ne sont pas connectés localement. Les routeurs, qui sont des dispositifs de couche 3, sont spécifiés sur cette couche et fournissent les services de routage dans un inter-réseau.

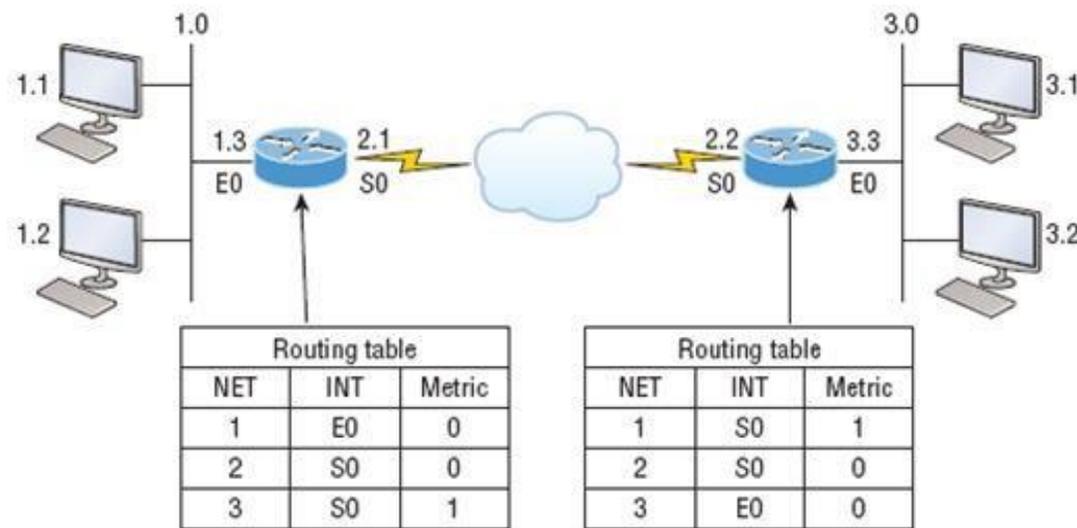
Les protocoles qui envoient des paquets de mise à jour de route sont appelés protocoles de routage; les plus importants sont **RIP, RIPv2, EIGRP, BGP et OSPF**. Les paquets de mise à jour d'itinéraire sont utilisés pour créer et gérer des tables de routage.

# I. 1. Introduction aux Réseaux :

## Modèles inter-réseaux : Le Modèle OSI

### La couche réseau :

La figure ci-dessous montre un exemple de table de routage. La table de routage que chaque routeur conserve et renvoie contient les informations suivantes:



# I. 1. Introduction aux Réseaux :

## Modèles inter-réseaux : Le Modèle OSI

### La couche réseau :

- ❑ **Adresses réseau** : Adresses réseau spécifiques au protocole. Un routeur doit gérer une table de routage pour les protocoles de routage individuels, car chaque protocole routé conserve la trace d'un réseau avec un schéma d'adressage différent. Par exemple, les tables de routage pour IP, IPv6 et IPv4 sont complètement différentes, de sorte que le routeur conserve une table pour chacun.
- ❑ **Interface** : Interface de sortie qu'un paquet prendra lorsqu'il est destiné à un réseau spécifique.
- ❑ **Métrique** : Distance jusqu'au réseau distant. Différents protocoles de routage utilisent différentes manières de calculer cette distance.

# I. 1. Introduction aux Réseaux :

## Modèles inter-réseaux : Le Modèle OSI

### La couche réseau :

Voici quelques caractéristiques de routeur que vous ne devriez jamais oublier:

- Par défaut, les routeurs ne transfèrent aucun paquet de diffusion ou de multidiffusion.
- Les routeurs utilisent l'adresse logique dans un en-tête de couche réseau pour déterminer le routeur du tronçon suivant vers lequel transférer le paquet.
- Les routeurs peuvent utiliser des listes d'accès, créées par un administrateur, pour contrôler la sécurité en fonction des types de paquets autorisés à entrer ou quitter une interface.
- Les routeurs peuvent fournir des fonctions de pontage de couche 2 si nécessaire et peuvent acheminer simultanément via la même interface.

# I. 1. Introduction aux Réseaux :

## Modèles inter-réseaux : Le Modèle OSI

### La couche réseau :

- ❑ Les périphériques de couche 3 (dans ce cas, les routeurs) fournissent des
- ❑ connexions entre des réseaux locaux virtuels (VLAN).
- ❑ Les routeurs peuvent fournir une qualité de service (QoS) pour des types spécifiques de trafic réseau.

# I. 1. Introduction aux Réseaux :

## Modèles inter-réseaux : Le Modèle OSI

### La couche de liaison de données :

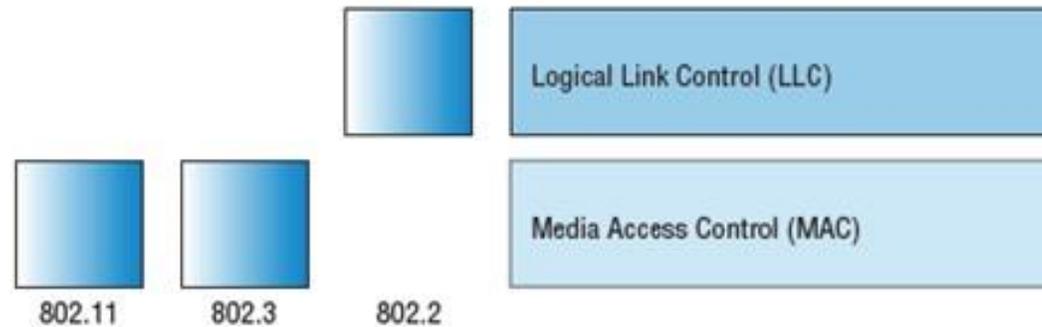
La couche de liaison de données assure la transmission physique des données et gère la notification d'erreur, la topologie du réseau et le contrôle de flux. Cela signifie que la couche Liaison de données garantit que les messages sont transmis au périphérique approprié sur un réseau local à l'aide d'adresses matérielles et convertit les messages de la couche réseau en bits que la couche physique doit transmettre.

# I. 1. Introduction aux Réseaux :

## Modèles inter-réseaux : Le Modèle OSI

### La couche de liaison de données :

La figure montre la couche Data Link avec les spécifications Ethernet et IEEE.



# I. 1. Introduction aux Réseaux :

## Modèles inter-réseaux : Le Modèle OSI

### La couche de liaison de données :

La sous-couche de contrôle d'accès au support (Media Access Control en anglais ou MAC) est la moitié basse de la couche de liaison de données du modèle OSI, selon les standards de réseaux informatiques IEEE

802.x. Elle sert d'interface entre la partie logicielle contrôlant la liaison d'un nœud (Contrôle de la liaison logique) et la couche physique (matérielle).

La sous-couche de contrôle de la liaison logique est la moitié haute de la couche de liaison de données du modèle OSI. Elle permet de fiabiliser le protocole MAC par un contrôle d'erreur et un contrôle de flux

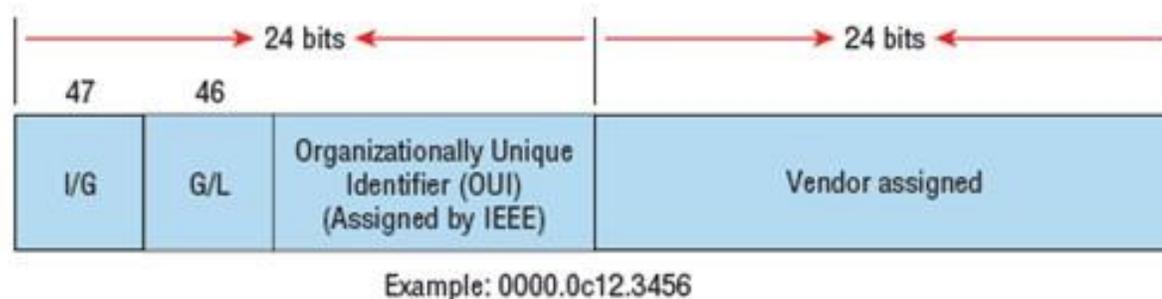
# I. 1. Introduction aux Réseaux :

## Modèles inter-réseaux : Le Modèle OSI

### La couche de liaison de données :

#### Adressage Ethernet

Une **adresse MAC** (Medium Access Control), parfois nommée **adresse physique**, est un identifiant physique stocké dans une carte réseau ou une interface réseau similaire. À moins qu'elle n'ait été modifiée par l'utilisateur, elle est unique au monde.



L'identificateur organisationnel unique (OUI) est attribué par l'IEEE à une organisation.

Il est composé de 24 bits, ou 3 octets, et il assigne à son tour une adresse administrée globalement composée de 24 bits, ou 3 octets, supposément unique à chaque adaptateur fabriqué par une organisation.

# I. 1. Introduction aux Réseaux :

## Modèles inter-réseaux : Le Modèle OSI

### La couche de liaison de données :

#### Adressage Ethernet

Le bit de poids fort est le bit Individual / Group (I / G). Quand il a une valeur de 0, on peut supposer que l'adresse est l'adresse MAC d'un périphérique et qu'il pourrait bien apparaître dans la partie source de l'en-tête MAC. Quand c'est un 1, on peut supposer que l'adresse représente une adresse de diffusion ou de multidiffusion dans Ethernet.

Le bit suivant est le bit global / local, parfois appelé bit G / L ou bit U / L, où U signifie universel. Lorsqu'il est défini sur 0, ce bit représente une adresse administrée globalement, comme attribué par l'IEEE, mais lorsqu'il s'agit d'un 1, il représente une adresse administrée localement.

# I. 1. Introduction aux Réseaux :

## Modèles inter-réseaux : Le Modèle OSI

### La couche de liaison de données :

#### Adressage Ethernet

Les 24 bits d'ordre inférieur d'une adresse Ethernet représentent un code administré localement ou assigné par le fabricant. Cette partie commence généralement avec 24 0s pour la première carte faite et continue dans l'ordre jusqu'à ce qu'il y ait 24 1s pour la dernière carte (16,777,216e) faite.

Vous constaterez que de nombreux fabricants utilisent ces mêmes six chiffres hexadécimaux que les six derniers caractères de leur numéro de série sur la même carte.

# I. 1. Introduction aux Réseaux :

## Modèles inter-réseaux : Le Modèle OSI

La couche de liaison de données :

La trame Ethernet

**Préambule** Un motif 1,0 alternatif fournit une horloge de 5 MHz au début de chaque paquet, ce qui permet aux dispositifs de réception de verrouiller le flux binaire entrant.

**Start Frame Delimiter (SFD) / Synch** Le préambule est de sept octets et le SFD est d'un octet (synch). Le SFD est 10101011, où la dernière paire de 1 permet au récepteur d'entrer dans le motif alternatif 1,0 quelque part au milieu et de se synchroniser encore pour détecter le début des données.

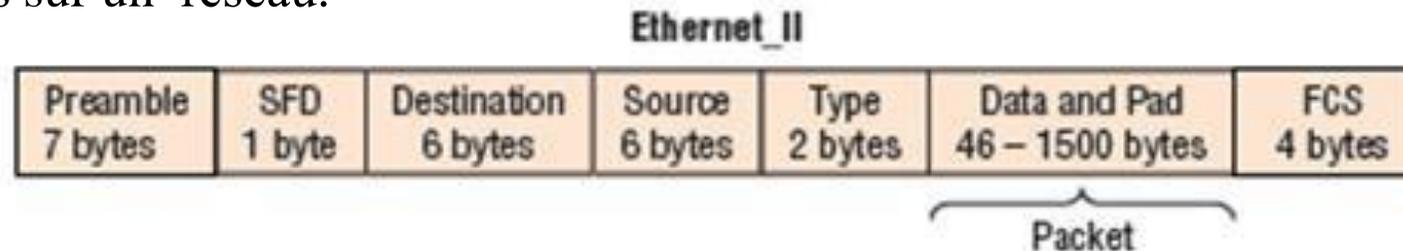
# I. 1. Introduction aux Réseaux :

## Modèles inter-réseaux : Le Modèle OSI

La couche de liaison de données :

La trame Ethernet

**Adresse de destination (DA)** Elle transmet une valeur de 48 bits en utilisant le bit le moins significatif (LSB) en premier. Le DA est utilisé par les stations réceptrices pour déterminer si un paquet entrant est adressé à un nœud particulier. L'adresse de destination peut être une adresse individuelle ou une adresse MAC de diffusion ou de multidiffusion. Rappelez-vous qu'une émission est 1s-all Fs en hexadécimal et est envoyé à tous les appareils. Une multidiffusion est envoyée uniquement à un sous-ensemble similaire de nœuds sur un réseau.



# I. 1. Introduction aux Réseaux :

## Modèles inter-réseaux : Le Modèle OSI

### La couche de liaison de données :

#### La trame Ethernet

**Adresse source (SA)** La AS est une adresse MAC 48 bits utilisée pour identifier le périphérique émetteur, et elle utilise le bit le moins significatif en premier. Les formats d'adresses de diffusion et de multidiffusion sont illégaux dans le domaine AS.

**Séquence de contrôle de trame (FCS)** FCS est un champ à la fin de la trame qui est utilisé pour stocker la réponse de contrôle de redondance cyclique (CRC). Le CRC est un algorithme mathématique qui est exécuté lorsque chaque image est construite sur la base des données du cadre. Lorsqu'un hôte destinataire reçoit la trame et exécute le CRC, la réponse doit être la même. Si ce n'est pas le cas, la trame est ignorée, en supposant que des erreurs se sont produites.

# I. 1. Introduction aux Réseaux :

## Modèles inter-réseaux : Le Modèle OSI

### La couche physique :

Enfin, en arrivant en bas, nous trouvons que la couche Physique fait deux choses: elle envoie des bits et reçoit des bits. Les bits viennent seulement dans les valeurs de 1 ou 0.

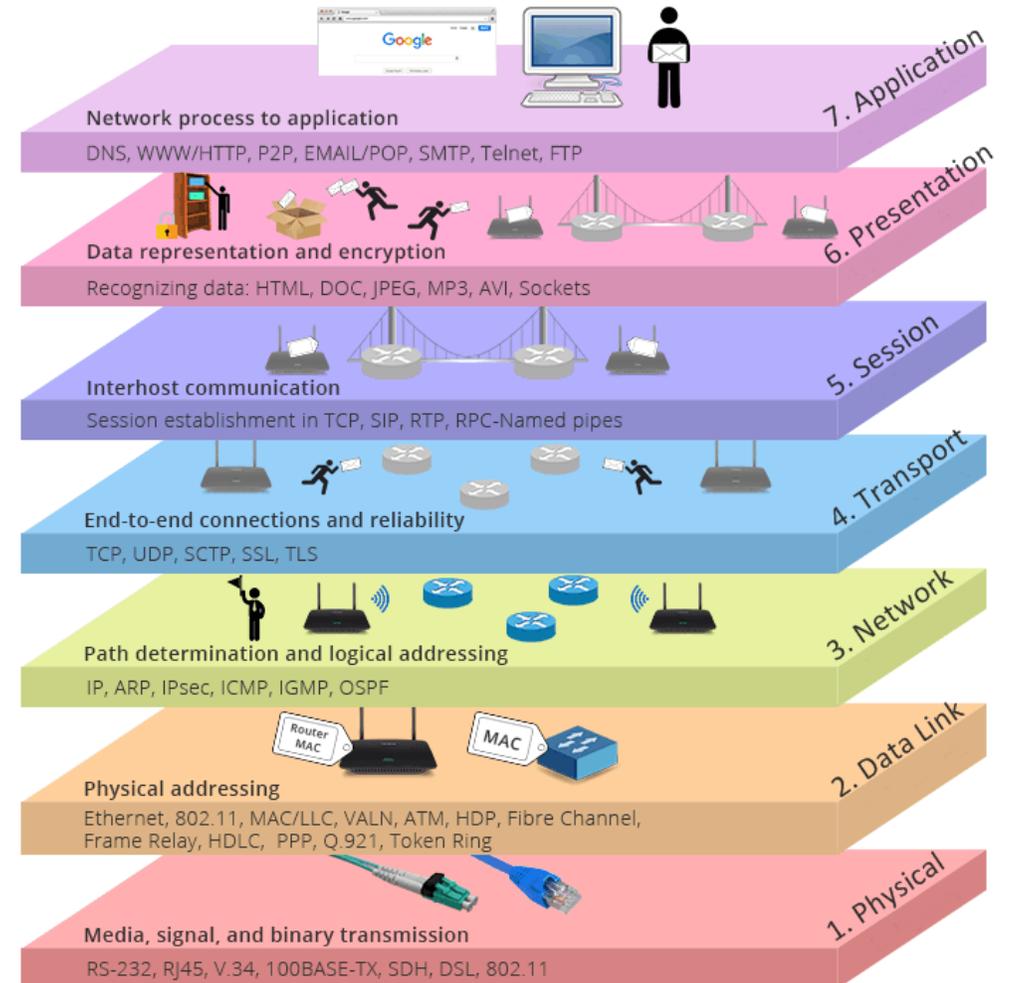
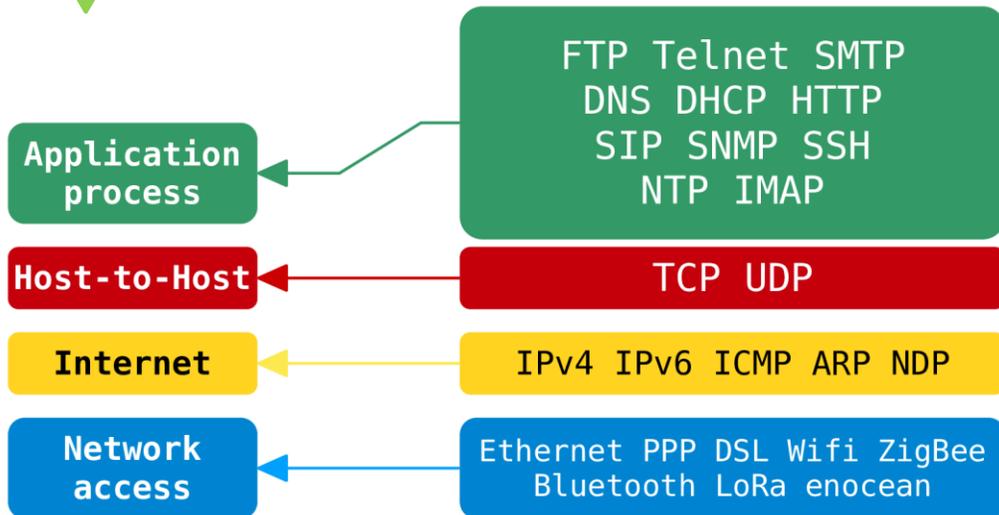
La couche Physique communique directement avec les différents types de supports de communication actuels. Différents types de médias représentent ces valeurs de bits de différentes manières.

# I. 1. Introduction aux Réseaux :

## Modèles inter-réseaux :

### Le Modèle OSI ►

### Le Modèle TCP/IP ▼



# I. 1. Introduction aux Réseaux :

## L'encapsulation :

### Encapsulation des données :

Lorsqu'un hôte transmet des données à travers un réseau à un autre périphérique, les données passent par un processus appelé encapsulation et sont enveloppés avec des informations de protocole à chaque couche du modèle OSI.

## Le modèle OSI

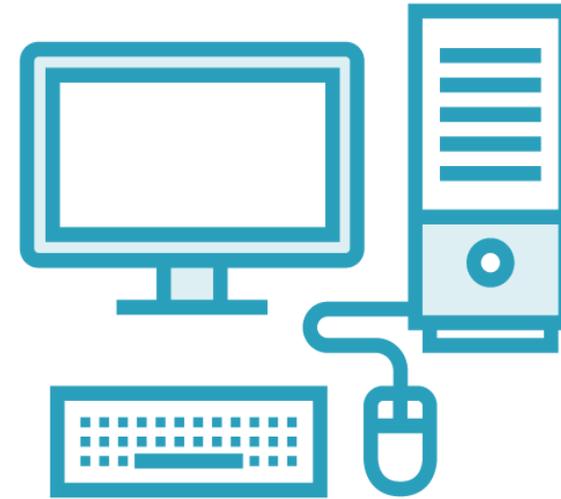
1	Physique
2	Liaison de données
3	Réseau
4	Transport
5	Session
6	Présentation
7	Application

# I. 1. Introduction aux Réseaux :

## L'encapsulation :



<http://www.fpk.ac.ma/>



La raison en est qu'il est souvent facile de conceptualiser la couche application. Il est facile de conceptualiser un site web, parce que si vous regardez un vidéo, vous êtes très probablement allé sur un site web pour accéder à la vidéo, il est possible que vous soyez passé par une application mobile, mais en fait, la plupart d'entre nous qui regardons un vidéo ont déjà navigué sur un site web sur internet, ce qui signifie que nous avons le concept de ce que le site web est pour nous. Je peux utiliser cela comme point de départ pour parler de ce qui se passe dans le reste des couches du modèle OSI.

# I. 1. Introduction aux Réseaux :

## L'encapsulation :

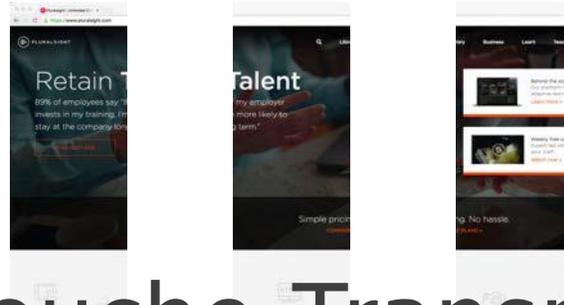
### Encapsulation des données :

Sur un périphérique de transmission, la méthode d'encapsulation de données fonctionne comme ceci:

1. Les informations de l'utilisateur sont converties en **données** pour transmission sur le réseau.
2. Les données sont converties en **segments** et une connexion fiable est établie entre les hôtes émetteurs et récepteurs.
3. Les segments sont convertis en **paquets** ou en datagrammes, et une adresse logique est placée dans l'en-tête de sorte que chaque paquet peut être acheminé via un inter réseau.
4. Les paquets ou les datagrammes sont convertis en **trames** pour transmission sur le réseau local. Les adresses matérielles (Ethernet) sont utilisées pour identifier de manière unique les hôtes sur un segment de réseau local.
5. Les trames sont converties en **bits**, et un schéma numérique de codage et d'horloge est utilisé.

# I. 1. Introduction aux Réseaux :

## L'encapsulation : 7 – Couche Application



## 4 – Couche Transport



# Segment

Un morceau de données, avec un en-tête de couche de transport.

Entête TCP

# I. 1. Introduction aux Réseaux :

## L'encapsulation :

### 4 – Couche Transport



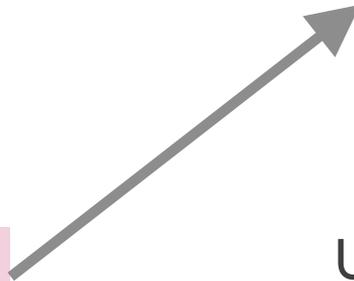
### 3 – Couche Réseau



## Paquet

Un bloc de données,  
avec un en-tête de  
couche réseau.

Entête IP



# I. 1. Introduction aux Réseaux :

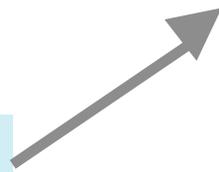
## L'encapsulation : 3 – Couche Réseau



## 2 – Couche Liaison de données

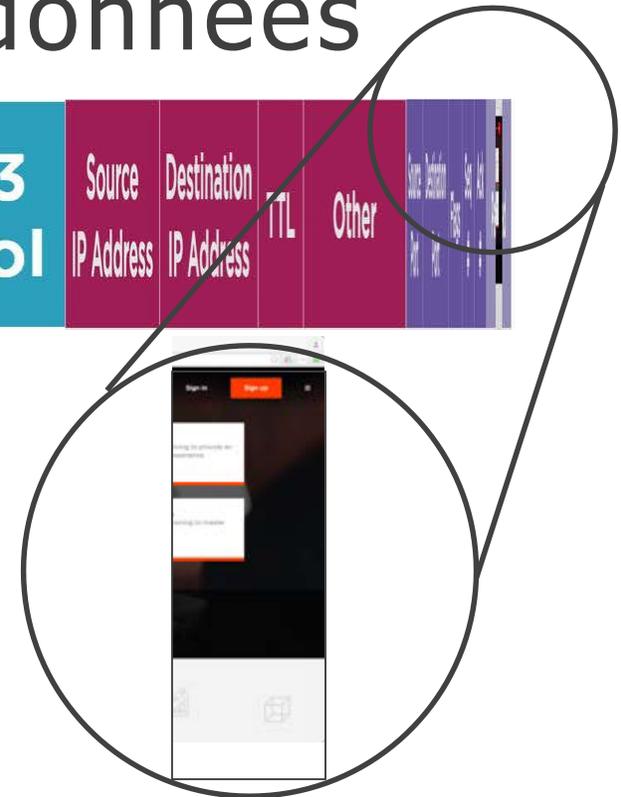


Entête Ethernet



## Trame

Un morceau de données, avec un entête de couche Data Link.



# I. 1. Introduction aux Réseaux :

## L'encapsulation :

### 2 – Couche liaison de données

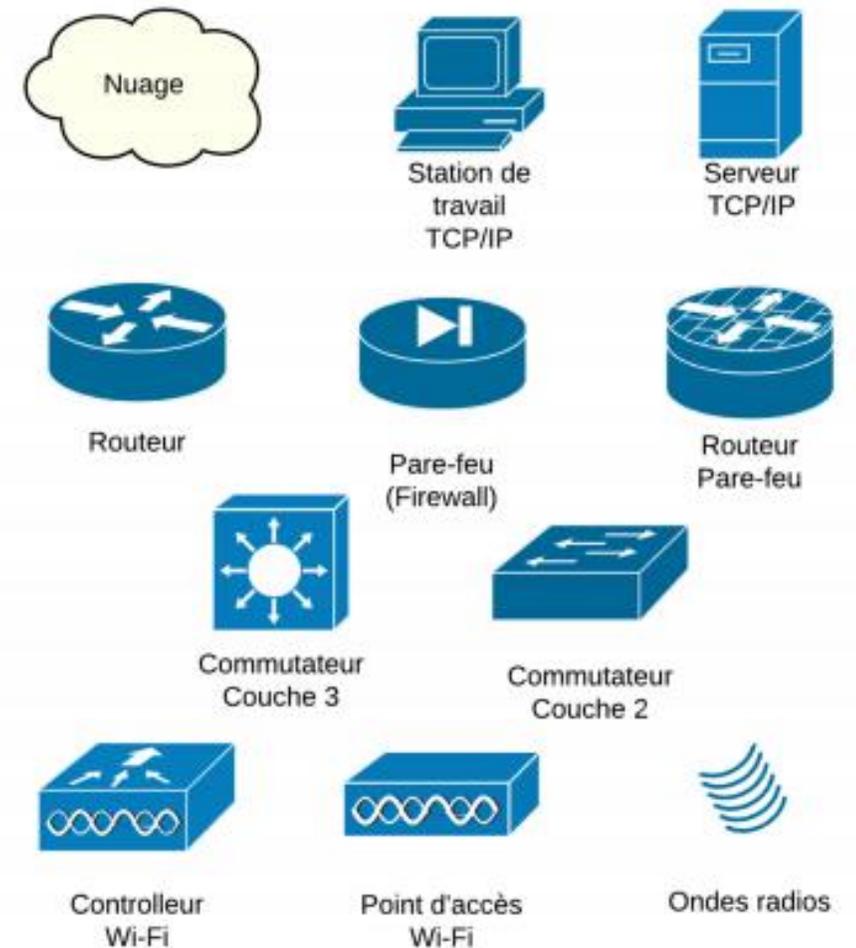


### 1– Couche Physique

# I. 1. Introduction aux Réseaux :

## Composants de base du réseau :

- ❑ Chaque couche, voire chaque protocole, dispose de sa propre vision des rôles des périphériques lors des procédures qui l'occupe. IP voit deux rôles :
  1. Les hôtes terminaux : nos ordinateurs au bout du réseau
  2. Les routeurs chargés de transférer les paquets en fonction de l'adresse L3 IP (logique, hiérarchique) de destination. Ils permettent d'interconnecter les hôtes d'extrémité.
  
- ❑ Aussi au sein du réseau local (LAN), le commutateur (switch) est chargé de transférer rapidement les trames Ethernet selon leur adresse L2 MAC (physique) de destination. Il ne s'occupe jamais des informations dites "L3", de couche 3 du modèle OSI comme les paquets ou datagrammes et les adresses IP qu'ils contiennent.



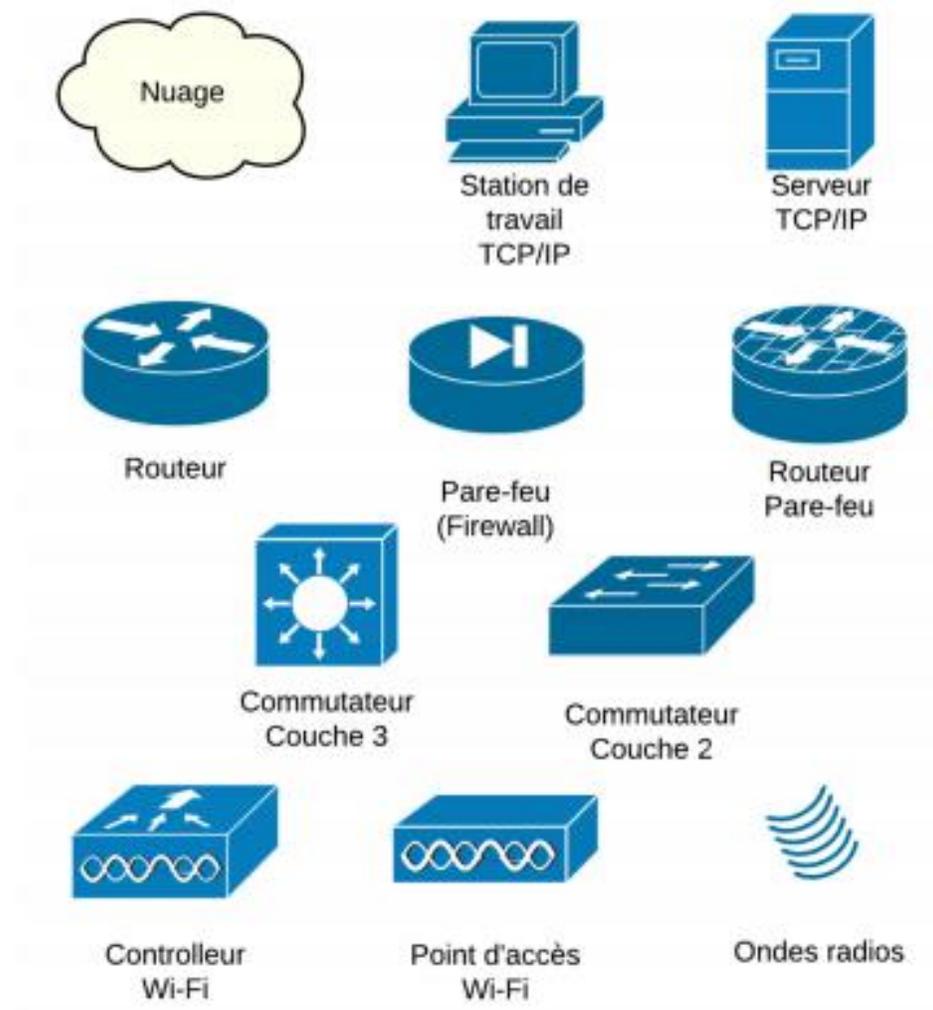
Composants de base du réseau

# I. 1. Introduction aux Réseaux :

## Composants de base du réseau :

- ❑ Chacun de ces périphériques est associé à une couche du modèle OSI avec “L” pour “Layer” :

Périphérique	Couche
Routeur (Router)	L3
Commutateur (Switch) L3	L2/L3
Commutateur (Switch) L2	L2
Pont (Bridge)	L2
Concentrateur (Hub)	L1
Répéteur (Repeater)	L1
Contrôleur WLAN	L2/L3/L7
Point d'accès sans-fil (AP) Wi-Fi	L1/L2
Carte réseau (NIC)	L2
Hôte terminal	L3/L4/L7



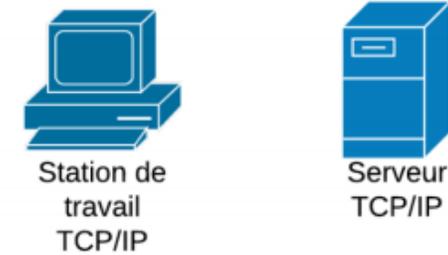
Composants de base du réseau

# I. 1. Introduction aux Réseaux :

## Composants de base du réseau :

### ❑ Périphérique terminal

Le périphérique terminal est l'hôte qui est situé à l'extrémité d'une communication : par exemple un poste client et un serveur sont des hôtes terminaux.



*Périphérique terminal*

### ❑ Routeur (router)

Le routeur est ce matériel de couche 3 du modèle OSI (L3) qui :

- interconnecte des domaines réseaux IP différents ;
- transfère le trafic qui ne lui est pas spécifiquement destiné ;
- transfère le trafic IP grâce à sa table de routage vers les bonnes destinations, s'il les connaît.



Routeur

# I. 1. Introduction aux Réseaux :

## Composants de base du réseau :

### ❑ Commutateur d'entreprise (switch)

Le commutateur d'entreprise (switch), typiquement de couche 2 liaison de données (L2) est ce matériel qui

- interconnecte les périphériques terminaux du LAN pour un transfert local rapide ;
- transfère le trafic en fonction de l'adresse MAC de destination trouvée dans les trames et de sa table de commutation (CAM table, ternary content addressable memory), la prise de décision de transfert est réalisée au niveau matériel hardware grâce à des puces ASIC ;
- transfère le trafic unicast (à destination d'un seul hôte) uniquement sur le bon port de sortie ;
- transfère le trafic Broadcast (à destination de tous) et multicast (à destination de certains) est transféré par tous les ports sauf le port d'origine.



Commutateur  
Couche 2

# I. 1. Introduction aux Réseaux :

## Composants de base du réseau :

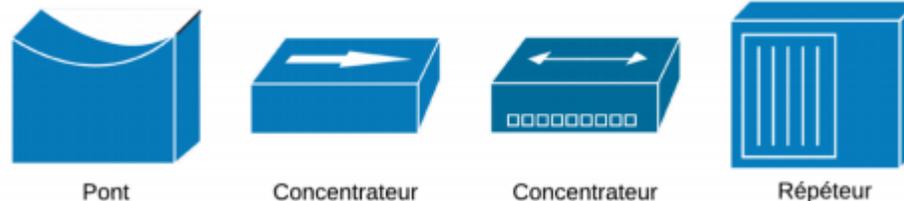
### ❑ Pont, concentrateur et Répéteur

Les “Ponts”, “Concentrateurs” et “Répéteurs” sont des éléments aujourd’hui plus conceptuels que réels (sauf dans certains déploiement en Wi-Fi).

Un pont (bridge) filtre le trafic entre deux segments physiques en fonction des adresses MAC. Le point d’accès Wi-Fi est une sorte de pont.

Un concentrateur (hub) est un périphérique qui concentre les connexions et étend le segment physique. À la différence du commutateur, le trafic sort par tous ses ports ; il ne prend aucune décision quant au trafic. En Ethernet, il était souvent identifié comme un répéteur multi-ports, ils ont été progressivement remplacés par des commutateurs de telle que ce matériel de “concentration” n’est plus fabriqué depuis longtemps.

Un répéteur (repeater) étend le signal entre deux ou plusieurs segments. Il ne prend aucune décision quant au trafic à transférer. On peut encore trouver des répéteurs dans les architectures Wi-Fi.

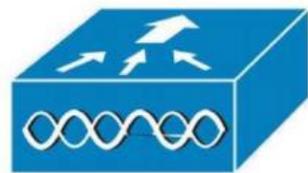


# I. 1. Introduction aux Réseaux :

## Composants de base du réseau :

### ❑ Matériel sans-fil

Avec des points d'accès légers (Lightweight AP), le Wireless LAN Controller (WLC) prend en charge les fonctions d'association ou d'authentification des APs, ces derniers devenant des interfaces physiques fournissant la connectivité. Le contrôleur fournit l'intelligence, la gestion, la configuration des APs. Il participe à une vue unifiée du réseau filaire et sans-fil.



Controlleur  
Wi-Fi



Point d'accès  
Wi-Fi



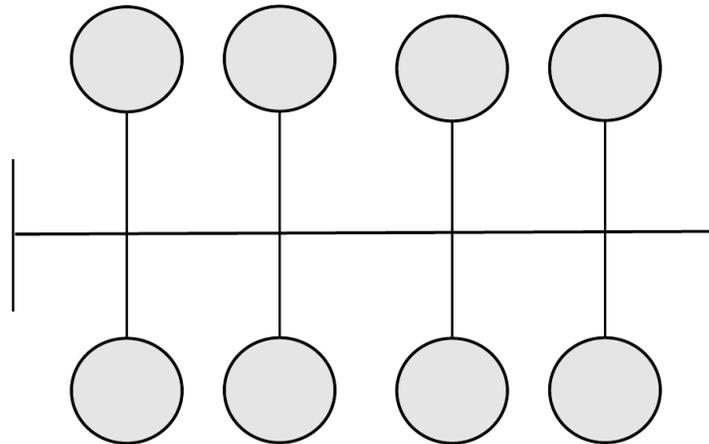
Ondes radios

# I. 1. Introduction aux Réseaux :

## Topologies du réseau :

### □ Topologie en Bus

Dans une topologie en bus, un seul canal est partagé par des points multiples. Une rupture de connexion empêcherait tout point de communiquer avec l'autre.



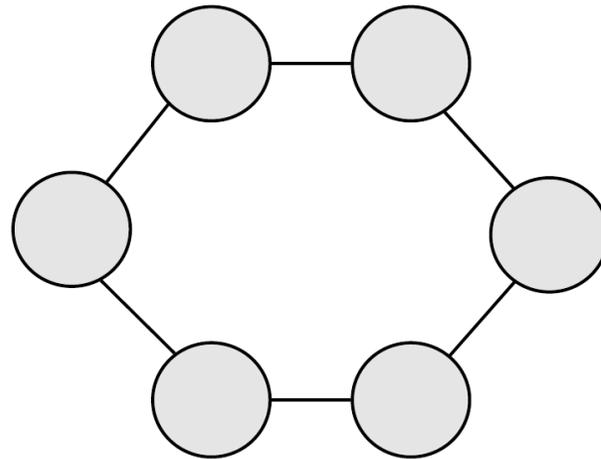
Topologie en bus

# I. 1. Introduction aux Réseaux :

## Topologies du réseau :

### ❑ Topologie en anneau

Dans une topologie en anneau, la rupture d'une liaison n'empêchera aucun point de communiquer avec l'autre.



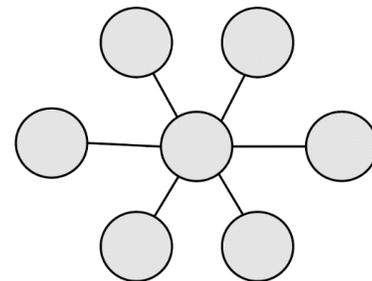
Topologie en anneau

# I. 1. Introduction aux Réseaux :

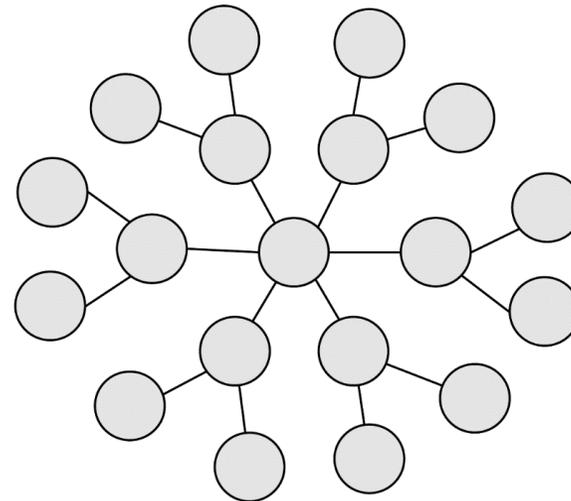
## Topologies du réseau :

### ❑ Topologie en étoile

Dans une simple topologie en étoile, le point de concentration est un point unique de défaillance. Selon l'endroit d'une rupture dans une topologie en étoile étendue, une partie des nœuds seront isolés les uns par rapport à autres nœuds.



Topologie en étoile  
(Star)



Topologie en étoile étendue  
(Extended Star)

Chapitre

# Systemes: GNU/Linux - Initiation et utilisation -



## II. 1. Linux et la communauté Open Source :

### Historique et définitions :

- ❑ **1984:** Richard Matthew Stallman, chercheur en informatique du MIT consacre son temps à l'écriture d'un système d'exploitation Libre du nom de GNU (GNU's Not Unix) .
- ❑ **1985:** Il annonce la création de la FSF (Free Software Foundation) afin de supporter ce projet.
- ❑ **1989:** Il a publié la première version de la licence GPL (General Public Licence) qui sera alors le fondement éthique, juridique et politique du mouvement du Libre.

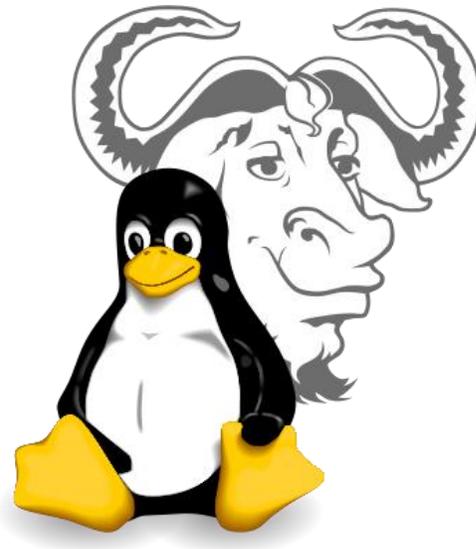


*Richard Matthew Stallman*

## II. 1. Linux et la communauté Open Source :

### Historique et définitions :

- ❑ **1990** : Linus Torvalds a développé un noyau qui a nommé LINUX.
- ❑ **1991** : Linus Torvalds rejoint la FSF, est persuadé d'intégrer son noyau à la GPL publié à cette époque
  - Naissance de l'OS GNU/Linux (Gnus sur Linux),.



*Richard Matthew Stallman*



*Linus Torvalds*

## II. 1. Linux et la communauté Open Source :

### Historique et définitions :

**Le concept libre:** fait référence à la liberté et non pas au prix. L'expression « **Logiciel Libre** » fait référence à la liberté pour les utilisateurs d'**exécuter**, de **copier**, de **distribuer**, d'**étudier**, de **modifier** et d'**améliorer** le logiciel.

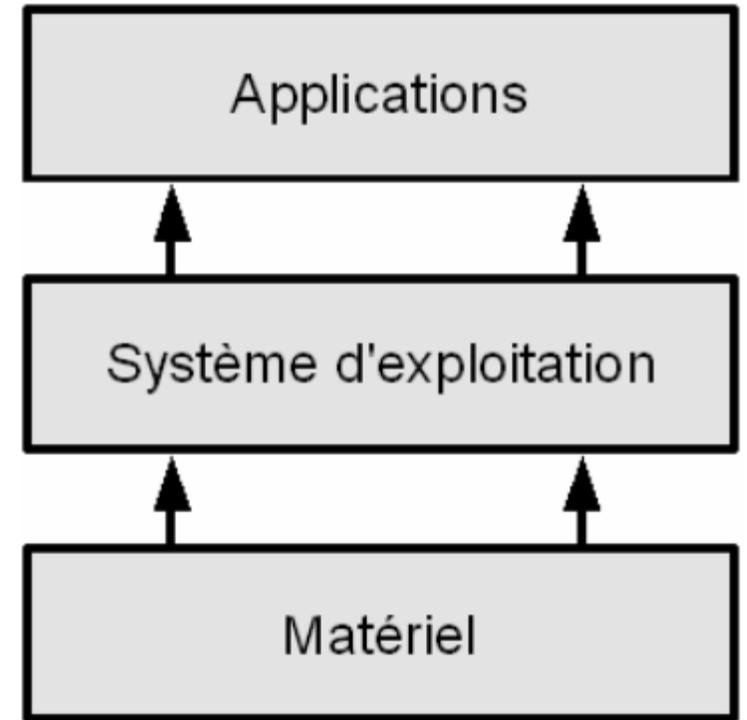
Plus précisément, elle fait référence à quatre types de liberté pour l'utilisateur du logiciel :

- ❑ **Liberté 1:** La liberté d'exécuter le programme, pour tous les usages.
- ❑ **Liberté 2:** La liberté d'étudier le fonctionnement du programme, et de l'adapter aux besoins. Pour ceci l'accès au code source est une condition requise.
- ❑ **Liberté 3:** La liberté de redistribuer des copies.
- ❑ **Liberté 4:** La liberté d'améliorer le programme et de publier ces améliorations, pour en faire profiter toute la communauté. Pour se faire, l'accès au code source est une condition requise.

## II. 1. Linux et la communauté Open Source :

### Systeme d'exploitation: definition :

- ❑ Un **systeme d'exploitation** est un programme ou un ensemble de programmes assurant la gestion de l'ordinateur et des peripheriques.
- ❑ Il sert d'interface entre le materiel (hardware) et le logiciel (software).
- ❑ C'est un ensemble de programmes tres complexes dont le but est de rendre plus simples les programmes et l'utilisation de l'ordinateur.



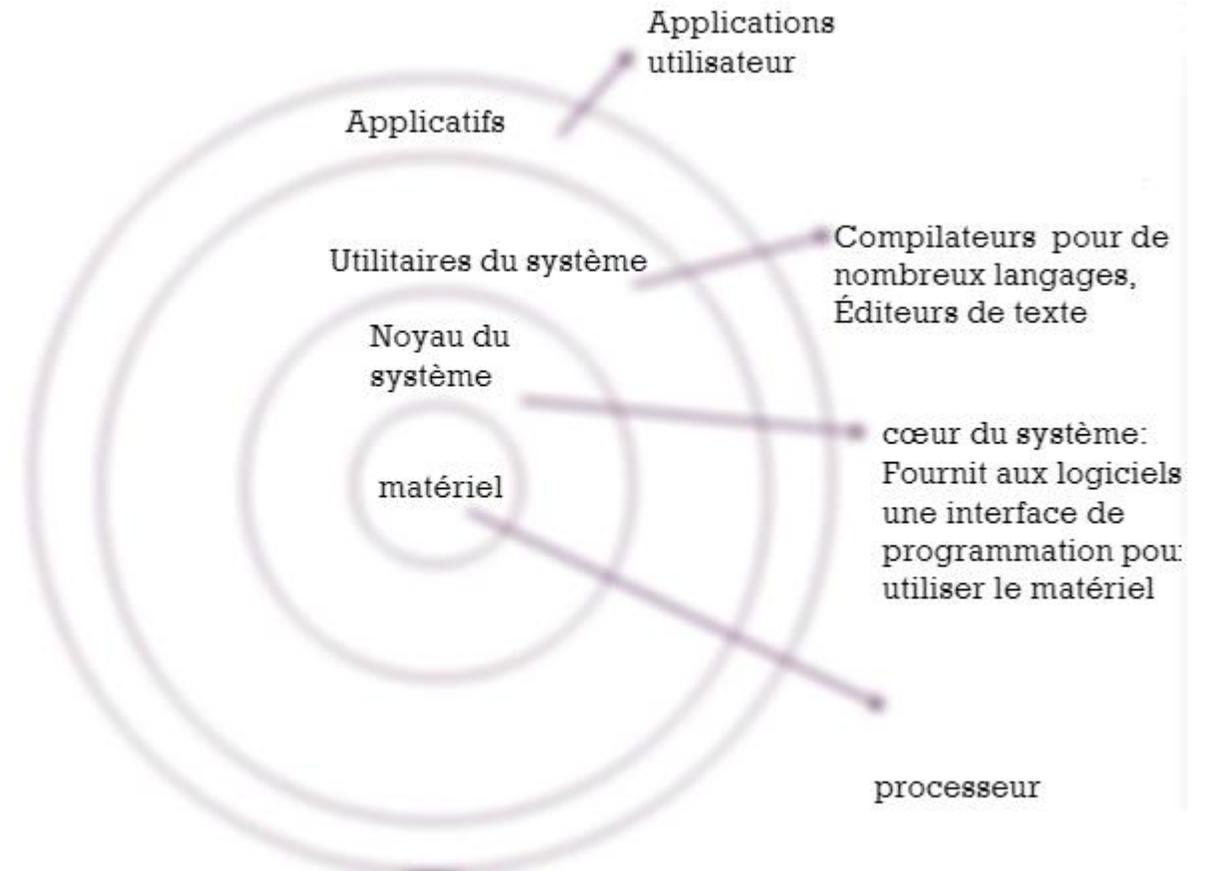
Principe du système d'exploitation

## II. 1. Linux et la communauté Open Source :

### Systeme d'exploitation: définition :

C'est le rôle du système d'exploitation de gérer :

- ❑ La mémoire,
- ❑ Les accès aux périphériques,
- ❑ Les données sur les disques,
- ❑ Les programmes,
- ❑ La sécurité,
- ❑ La collecte des informations



## II. 1. Linux et la communauté Open Source :

### Systeme d'exploitation: définition :

- ❑ GNU/Linux est un SE constitué d'un **Noyau Linux** d'un **interpréteur de commandes** et d'un grand nombre **d'utilitaires**;
- ❑ **Le noyau** : Gestion des ressources physiques (processeur, mémoires , périphériques) et logicielles(processus, fichiers ..);
- ❑ L'interface entre les programmes utilisateurs et le noyaux est assurée par **des procédures et des fonctions**;
- ❑ Tout est écrite en langage C;
- ❑ Comme tout SE , l'utilisateur n'accède pas directement au noyau , mais à un interpréteur de commande:  
**Le shell.**

## II. 1. Linux et la communauté Open Source :

### Caractéristiques linux :

**GNU/Linux** est un système d'exploitation permettant de contrôler un PC et ses différents périphériques.

Il se distingue par les caractéristiques suivantes :

- ❑ **Multi-utilisateurs** : qui peut être utilisé simultanément par plusieurs personnes;
- ❑ **Multi-tâches** : un utilisateur peut exécuter plusieurs programmes en même temps;
- ❑ Repose sur un **noyau** (kernel) utilisant 4 concepts principaux **fichiers, droits d'accès, processus** et **communication interprocessus** (IPC).

## II. 1. Linux et la communauté Open Source :

### Caractéristiques linux :

#### Unix : multi utilisateurs

- ❑ Plusieurs utilisateurs sous **GNU/Linux**
- ❑ Chacun dispose de l'ensemble des ressources du système.
- ❑ Comme tout système multi-utilisateur, **GNU/Linux** comporte des mécanismes d'identification et de protection permettant d'éviter toute interférence entre utilisateurs.
- ❑ **2 types de Users :**
  - ❑ **1/ Users normaux** : compte avec Login, password, Espace de travail protégé (rep. privé -home directory),
  - ❑ **2/ Super-User** root gère tout le système,

## II. 1. Linux et la communauté Open Source :

### Caractéristiques linux :

#### Unix : multi tâches

- ❑ **GNU/Linux** est multi-tâche car plusieurs programmes peuvent être en cours d'exécution en même temps sur une même machine.
- ❑ Un processus est une tâche en train de s'exécuter.
- ❑ On appelle processus, l'image de l'état du processeur et de la mémoire au cours de l'exécution du programme.
- ❑ En fait, à chaque instant, le processeur ne traite qu'au plus un seul des programmes lancés.
- ❑ La gestion des processus est effectuée par le système.

## II. 1. Linux et la communauté Open Source :

### Systeme d'exploitation: structure

#### Fonctions principales d'UNIX

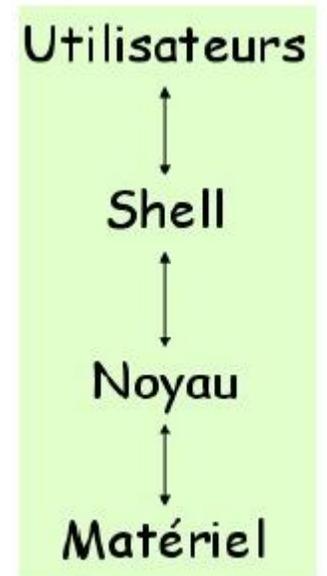
- ❑ **Partage des ressources équitables** : veiller au partage équitable des ressources entre tous les processus.
- ❑ **Interface avec le matériel** : passage par des fichiers spéciaux gérés par le SE. pour accéder à une ressource matériel (disque dur, lecteur de disquettes, CD Rom)
- ❑ **Gestion de la mémoire** : partage correct de la RAM entre processus.
- ❑ **Gestion des fichiers** : Unix fournit un mécanisme de protection des fichiers.

## II. 1. Linux et la communauté Open Source :

### Systeme d'exploitation: structure

#### Schéma d'exploitation de la machine

- ❑ **Shell** : interpréteur de commandes Unix (vérifie, interprète les commandes, exécute et renvoie les réponses). Le Shell envoie des appels au noyau en fonction des requêtes des utilisateurs
- ❑ **noyau** : couche logicielle la plus interne du S.E dédiée à la gestion des composants matériels : processeur, mémoire, périph.
- ❑ Autour du noyau gravite un certain nombre d'utilitaires.



## II. 1. Linux et la communauté Open Source :

### En résumé

- ❑ Unix est un système d'exploitation **Multi-Utilisateur** : Plusieurs utilisateurs peuvent se connecter sur le même serveur Unix simultanément
- ❑ Unix est un système **multi-tâches** : un utilisateur peut lancer plusieurs tâches simultanément ; un processus correspond à l'exécution d'un programme à un instant donné
- ❑ Le système de fichiers est un système hiérarchisé arborescent , les périphériques sont considérés , du point de vue de l'utilisateur, comme des fichiers
- ❑ Le système est écrit à 99% en C, ainsi il a été écrit de façon à être réellement **portable**

## II. 1. Linux et la communauté Open Source :

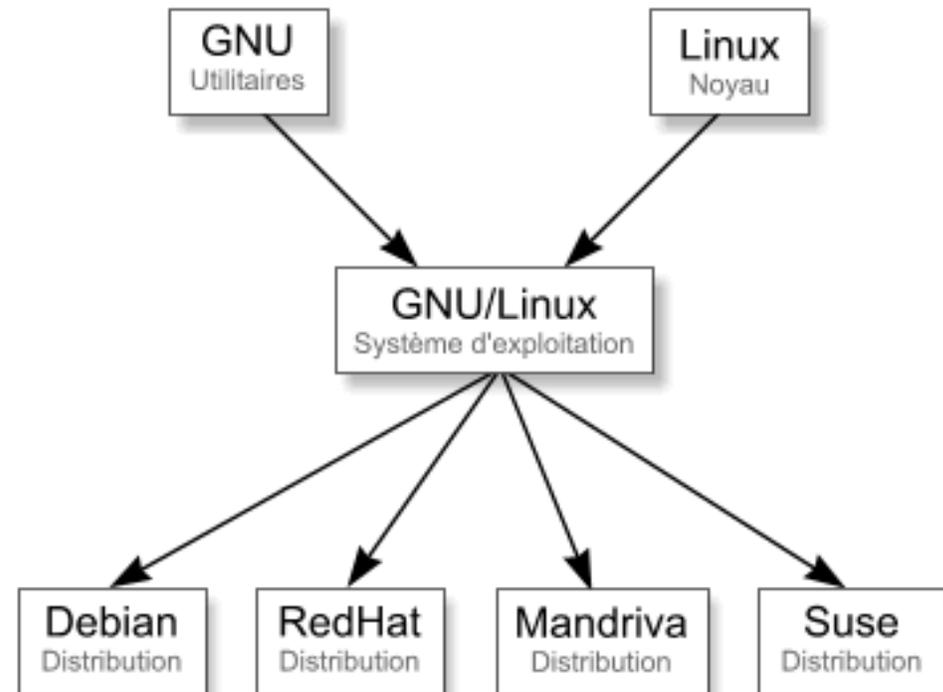
### Les distributions linux:

- ❑ Une distribution Linux est un ensemble cohérent de logiciels rassemblant un système d'exploitation composé d'un noyau Linux et d'applications, la plupart étant des logiciels libres;
- ❑ **Distribution** = un ensemble de programme (Noyau, sources des utilitaires, commandes, applications..)formant après installation un système complet;
- ❑ Plusieurs distributions : débiter avec linux = choisir sa distribution selon ses attentes du système
- ❑ **1994** : Redhat première société ayant pour objectif de rassembler tout ce qui est nécessaire dans une distribution

## II. 1. Linux et la communauté Open Source :

### Les distributions linux:

- ❑ **Plusieurs distributions :** Ubuntu , Debian , Knoppix , Fedora , Centos, Fedora , Suse ...
- ❑ Le kernel ( noyau linux et commun ) → distribution = noyau + (applications +utilitaires ..)



## II. 2. Installation de Linux :

### Préparation de la plateforme de travail :

1. Choisir sa distribution;
2. Choisir le mode d'installation :
  1. En double boot avec un autre OS (Dual boot);
  2. Live;
  3. Système invité sur un OS host ( virtualisation):
    - ❑ VirtualBox ,Vmware (Player; Workstation ), ou autre

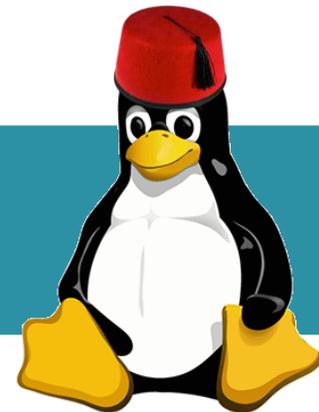
## II. 2. Installation de Linux :

Préparation de la plateforme de travail :

**TP1: Installation de Linux sur un ordinateur virtuel**

Chapitre

# Systemes: GNU/Linux « Initiation et utilisation »



## III. 1. Systèmes: GNU/Linux « Initiation et utilisation »

### Systèmes de Fichiers : Le Shell

#### L'environnement shell :

Après connexion, l'utilisateur est connecté dans son environnement cela signifie que le shell met à sa disposition des variables d'environnement, c'est-à-dire un conteneur mémoire dans lequel des données sont stockées.

#### Connexion-Déconnexion :

Pour permettre à plusieurs users de travailler en même temps, GNU/Linux met en œuvre des mécanismes d'identification des users, de protection et de confidentialité de l'information.

Tout user est identifié par un (login name).

## III. 1. Systèmes: GNU/Linux « Initiation et utilisation »

### Systèmes de Fichiers :

### Connexion-Déconnexion :

**Connexion** : S'identifier pour ouvrir une session (de travail) :

- ❑ Entrer nom de connexion après le message login;
- ❑ Entrer mot de passage après le message password;
- ❑ L'utilisateur se trouve alors dans son répertoire privé correspondant à son login (home directory)

**Déconnexion** : En l'absence d'environnement graphique, la commande `exit` suffit pour terminer une session de travail.

## III. 1. Systèmes: GNU/Linux « Initiation et utilisation »

### Systèmes de Fichiers :

#### Mot de passe :

- ❑ Lors de la première connexion, il est fortement conseillé à l'utilisateur de s'attribuer un mot de passe.
- ❑ Ce mot de passe est chiffré.
- ❑ Il est impossible de le retrouver à partir du mot chiffré.
- ❑ Le root ne peut pas le détruire et définir un nouveau.
- ❑ L'utilisateur peut à tout moment changer son mot de passe, et ce par la commande `passwd`.

#### Exemple:

```
youness@youness-VirtualBox:~$ passwd
Changement du mot de passe pour youness.
Mot de passe actuel :
Nouveau mot de passe :
Retapez le nouveau mot de passe :
passwd : le mot de passe a été mis à jour avec succès
youness@youness-VirtualBox:~$
```

## III. 1. Systèmes: GNU/Linux « Initiation et utilisation »

### Systèmes de Fichiers :

#### Fichiers relatifs à la connexion `/etc/passwd` :

- ❑ La liste des users du systèmes est stockée dans le fichier `/etc/passwd`.
- ❑ Le fichier est accessible en lecture à tous les utilisateurs.
- ❑ Pour chaque utilisateur ce fichier contient les champs suivant :

Nom de connexion(login): password (\* ou x): le UID: le GID: [commentaire]: le répertoire d'accueil: le programme

## III. 1. Systèmes: GNU/Linux « Initiation et utilisation »

### Systèmes de Fichiers :

#### Fichiers relatifs à la connexion /etc/passwd :

- ❑ **Nom d'utilisateur**, jusqu'à 8 caractères (sensible à la casse).
- ❑ Un **“x”** ou **« \* »** dans le champ mot de passe. Les mots de passe sont stockés dans le fichier */etc/shadow*.
- ❑ **Numéro d'identification de l'utilisateur**: le système utilise ce champ, plus le champ suivant du groupe, pour identifier quels fichiers appartiennent à l'utilisateur.
- ❑ **Numéro d'identification du groupe**: Tout utilisateur est affecté à un et un seul groupe de base. Il peut, par ailleurs, faire partie d'un grand nombre d'autres groupes, habituellement, le numéro de groupe sera le même que le numéro d'utilisateur.

## III. 1. Systèmes: GNU/Linux « Initiation et utilisation »

### Systèmes de Fichiers :

#### Fichiers relatifs à la connexion /etc/passwd :

- ❑ **Commentaire:** contient généralement le nom complet de l'utilisateur, une description du compte ou une remarque (moins de 30 caractères).
- ❑ **Répertoire personnel de l'utilisateur:** Habituellement, /home/username. Tous les fichiers personnels de l'utilisateur, ses pages web, ses mails à envoyer, etc,... seront stockés dans son espace de travail (répertoire d'accueil).
- ❑ **Programme :** (interpréteur de commande) Souvent fixé à ``/bin/bash" pour permettre l'accès au shell bash,

# III. 1. Systèmes: GNU/Linux « Initiation et utilisation »

## Systèmes de Fichiers :

### Fichiers relatifs à la connexion /etc/passwd :

- ❑ Ci-après un extrait du fichier /etc/passwd

```
youness@youness-VirtualBox:~$ cat /etc/passwd  
root:x:0:0:root:/root:/bin/bash  
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin  
bin:x:2:2:bin:/bin:/usr/sbin/nologin
```

```
nologin  
youness:x:1000:1000:youness,,,:/home/youness:/bin/bash  
systemd-coredump:x:999:999:systemd Core Dumper:/:/usr/sbin
```

## III. 1. Systèmes: GNU/Linux « Initiation et utilisation »

### Systèmes de Fichiers :

#### Fichiers relatifs à la connexion /etc/passwd :

- ❑ **Le shell:** est un programme ayant pour fonction d'assurer l'interface entre l'utilisateur et le système Linux.
- ❑ C'est un interpréteur de commandes. Autrement dit, c'est quelque chose qui permet de communiquer avec l'ordinateur par le biais du clavier et de l'écran.
- ❑ **Le shell** est la partie système qui permet de saisir des commandes.
- ❑ Un script **shell** est une série de commandes écrites dans un fichier.
- ❑ Le **shell** lit les commandes depuis le fichier comme si ils sont invoquées directement dans un terminal.

## III. 1. Systèmes: GNU/Linux « Initiation et utilisation »

### Systèmes de Fichiers :

#### Fichiers relatifs à la connexion /etc/passwd :

- ❑ Il y donc deux manières d'utiliser Bash :
  - ❑ comme interface utilisateur ;
  - ❑ comme environnement de programmation (traité dans le chapitre programmation shell).

# III. 1. Systèmes: GNU/Linux « Initiation et utilisation »

## Systèmes de Fichiers

### Commande Linux en console

- Unix fonctionne en *mode ligne de commandes* et non en *mode graphique* permet des opérations plus complexes.
- Une *commande* est un programme. Pour l'exécuter, on tape son *nom* éventuellement suivi d'*options* et d'*arguments*.
- Une option dans une ligne de commande est une lettre, et elle commence par un tiret « - »,
- L'espace est le caractère séparateur des différents éléments d'une commande

**Syntaxe :**

```
nom_commande [-liste_options][liste_arguments]
```

# III. 1. Systèmes: GNU/Linux « Initiation et utilisation »

## Systèmes de Fichiers

### Commande Linux en console:

#### Conventions utilisées pour la syntaxe des commandes

```
nom_commande [-liste_options][liste_arguments]
```

- 1) **Commande et options:** Le texte de la commande ainsi que celui des options est à reprendre tel quel. Dans le manuel Linux, ce texte est représenté en caractères gras ou soulignés.
- 2) **Paramètre:** Un nom symbolique décrit l'usage de chaque paramètre. Ce nom symbolique doit être remplacé par la valeur effective que l'utilisateur désire donner au paramètre.
- 3) **[...]** : Le paramètre ou l'option entre crochets est optionnel.
- 4) **Paramètre...** : Les trois points qui suivent un nom symbolique d'un paramètre désignent une liste de paramètres.

# III. 1. Systèmes: GNU/Linux « Initiation et utilisation »

## Systèmes de Fichiers

### Commande Unix en console

#### Exemple

*ls -l*

Lors de l'appui sur la touche Entrée, le shell analyse la ligne de commande et l'interprète.

Différence entre majuscules et minuscules. : On dit que la console Unix est sensible à la casse.

La commande : *date*

```
youness@youness-VirtualBox:~$ date
jeu. 14 avril 2022 02:38:34 +00
youness@youness-VirtualBox:~$ █
```

Si la commande existe, elle est exécutée et affiche à l'écran le résultat, en cas d'erreur, le programme indique l'erreur à l'écran.

# III. 1. Systèmes: GNU/Linux « Initiation et utilisation »

## Systèmes de Fichiers

### Définition:

*fichier* : objet recevant et délivrant des données, constitué d'une chaîne de caractères non structurée.

### Type de Fichiers :

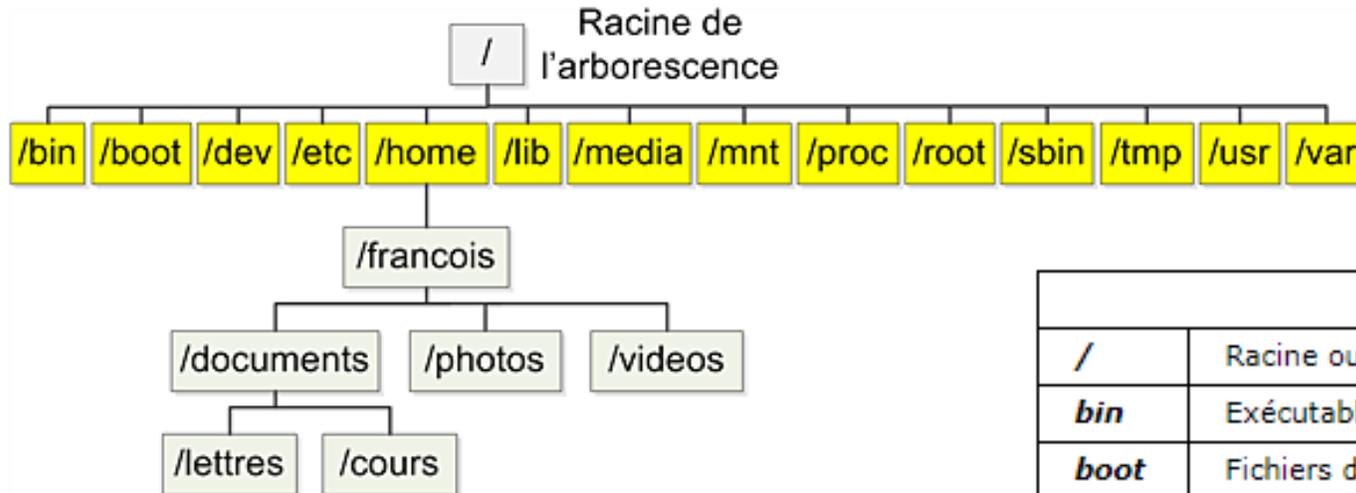
- ***Les fichiers ordinaires*** : données stockées sur un disque, ce type de fichier sert à mémoriser les programmes et les données des users et du système.
- ***Les fichiers répertoire*** : ensemble d'informations permettant l'accès à d'autres fichiers;
- ***Les fichiers spéciaux*** : ils désignent les périphériques, dispositif d'entrée/sortie (terminal, lecteur,...)

### Description de Fichiers

dans un i-noeud (inode) comportant **type** de fichier, **mode de protection**, **nb. liens**, **num. propriétaire num groupe**, **taille fichier**, **adr.physique** **direct** **date** et **heure** dernière modif., **date** **heure** dernier **accès**.

# III. 1. Systèmes: GNU/Linux « Initiation et utilisation »

## Systèmes de Fichiers



*Arborescence des fichiers linux*

Principaux répertoires de l'arborescence Linux	
<b>/</b>	Racine ou root, contient les répertoires de l'arborescence Linux.
<b>bin</b>	Exécutables binaires du système <i>cp, ls, mount, rm...</i>
<b>boot</b>	Fichiers de démarrage de Linux.
<b>dev</b>	Fichiers spéciaux assurant la liaison avec les périphériques.
<b>etc</b>	Fichiers de configuration du système, des services...
<b>home</b>	Répertoire personnel des utilisateurs.
<b>lib</b>	Bibliothèques système partagées.
<b>media</b>	Point de montage des clés USB, CD-ROM...
<b>mnt</b>	Point de montage temporaire de partitions et périphériques.
<b>proc</b>	Informations sur les processus et le noyau Linux.
<b>root</b>	Répertoire personnel du super-utilisateur.
<b>sbin</b>	Binaires ...

# III. 1. Systèmes: GNU/Linux « Initiation et utilisation »

## Systèmes de Fichiers

### Chemin absolu et relatif

#### ***Notion de chemin d'accès :***

Pour identifier un fichier : suite de noms étiquetant les arêtes le long de l'arborescence.

racine absolue : /

/ sert aussi de séparation entre sous-répertoires.

***Référence absolue*** : chemin d'accès pathname depuis la racine (permettant le repérage sans ambiguïté)

**Exemple** : /home/prot1/formation/GI16

***Référence relative*** : Selon l'endroit où l'on se situe ( répertoire de travail = working directory), repérer un fichier peut s'effectuer de manière relative.

**Exemple** : ../../DESS

# III. 1. Systèmes: GNU/Linux « Initiation et utilisation »

## Systèmes de Fichiers

### Désignation des fichiers

Des fichiers physiques différents appartenant à des disques logiques distincts peuvent avoir le même index de i-noeud ) impossible de créer des liens . Le système GNU/linux permet de créer des liens symboliques entre des fichiers.

#### Définition (Lien symbolique)

fichier contenant la référence absolue d'un autre fichier. Toute opération sur ce fichier (lecture, écriture, : : :) s'effectue sur le fichier référencé. Un lien est créé pour pouvoir accéder au même fichier à différents endroits de l'arborescence.

Commandes :

```
ln -s f_cible lien_nom
```

créé un lien symbolique `lien_nom` contenant la référence à `f_cible`

`ls -l` fait apparaître le lien sous la forme `fich_dest -> fich_source`

# III. 1. Systèmes: GNU/Linux « Initiation et utilisation »

## Systèmes de Fichiers

### Le Shell = utilisation de CLI (ligne de commande)

#### Pourquoi utiliser la ligne de commande ?

Beaucoup de solutions sont données en ligne de commande, non pas que GNU/Linux n'ait pas d'interface graphique, mais pour certaines tâches, l'utilisation de la ligne de commande s'avère bien plus pratique et plus puissante que la fameuse souris.

# III. 1. Systèmes: GNU/Linux « Initiation et utilisation »

## Systèmes de Fichiers

### La manipulation des répertoires

Après l'ouverture de sa session, l'utilisateur se trouve sous le contrôle d'un interpréteur de commande (shell). Ce dernier est prêt à lire, analyser et exécuter les commandes qui lui sont soumises.

pwd

***Print working directory***

Affiche le chemin d'accès du répertoire courant

Exemple:

```
youness@youness-VirtualBox:~$ pwd
/home/youness
youness@youness-VirtualBox:~$
```

# III. 1. Systèmes: GNU/Linux « Initiation et utilisation »

## Systèmes de Fichiers

### La manipulation des répertoires

cd

#### *Change directory*

Permet de changer le répertoire du travail.

Exemple:

```
youness@youness-VirtualBox:~$ pwd
/home/youness
youness@youness-VirtualBox:~$ cd ..
youness@youness-VirtualBox:/home$ pwd
/home
youness@youness-VirtualBox:/home$ cd
youness@youness-VirtualBox:~$ pwd
/home/youness
youness@youness-VirtualBox:~$ █
```

← Permet de se repositionner sur son répertoire d'accueil.

# III. 1. Systèmes: GNU/Linux « Initiation et utilisation »

## Systèmes de Fichiers

### La manipulation des répertoires

mkdir

***MaKe DIRectory***

Permet de créer un nouveau répertoire.

Exemple:

```
youness@youness-VirtualBox:~$ mkdir smi
youness@youness-VirtualBox:~$ cd smi/
youness@youness-VirtualBox:~/smi$ pwd
/home/youness/smi
youness@youness-VirtualBox:~/smi$
```

# III. 1. Systèmes: GNU/Linux « Initiation et utilisation »

## Systèmes de Fichiers

### La manipulation des répertoires

rmkdir

*ReMove DIRectory*

Permet de supprimer un répertoire vide.

Exemple:

```
youness@youness-VirtualBox:~$ rmdir smi
youness@youness-VirtualBox:~$ pwd
/home/youness
youness@youness-VirtualBox:~$
```

# III. 1. Systèmes: GNU/Linux « Initiation et utilisation »

## Systèmes de Fichiers

### La manipulation des répertoires

du

#### *Disk Usage*

Donne l'occupation disques des fichiers, des rep et sous rep en KO.

Exemple:

```
youness@youness-VirtualBox:~$ du /home/youness/rep1
4      /home/youness/rep1
youness@youness-VirtualBox:~$
```

# III. 1. Systèmes: GNU/Linux « Initiation et utilisation »

## Systèmes de Fichiers

### La manipulation des fichiers

Quel que soit le travail à faire sur la machine, l'utilisateur serait amené à effectuer certaines tâches élémentaires telles que, lister le contenu d'un répertoire, copier, effacer ou afficher les fichiers, ci après une liste non exhaustive des commandes qui les réalisent:

ls

#### ***LiSt files***

Permet d'obtenir la liste et les caractéristiques des fichiers contenus dans un répertoire, si aucun argument n'est donné avec la commande, ls affiche la liste des noms des fichiers du répertoire courant par ordre alphabétique.

**Exemple:**

```
youness@youness-VirtualBox:~$ ls
Bureau      Images      Musique     rep1        Vidéos
Documents   Modèles     Public      Téléchargements
youness@youness-VirtualBox:~$
```

# III. 1. Systèmes: GNU/Linux « Initiation et utilisation »

## Systèmes de Fichiers

### La manipulation des fichiers

Ls - a

afficher tous les fichiers et dossiers cachés

Exemple:

```
youness@youness-VirtualBox:~$ ls -a
.          Musique
..         .pam_environment
.bash_history .profile
.bash_logout Public
.bashrc    rep1
Bureau    .ssh
.cache    .sudo_as_admin_successful
.config   Téléchargements
Documents .vboxclient-clipboard.pid
.gnupg    .vboxclient-display-svgx-x11.pid
Images    .vboxclient-draganddrop.pid
.local    .vboxclient-seamless.pid
Modèles   Vidéos
youness@youness-VirtualBox:~$
```

# III. 1. Systèmes: GNU/Linux « Initiation et utilisation »

## Systèmes de Fichiers

### La manipulation des fichiers

Ls - F

indique le type d'élément.  
Il rajoute à la fin des éléments un symbole pour qu'on puisse faire la distinction entre les dossiers, fichiers, raccourcis

Exemple:

```
youness@youness-VirtualBox:~$ ls -F
Bureau/      Images/      Musique/     rep1/        Vidéos/
Documents/   Modèles/     Public/     Téléchargements/
youness@youness-VirtualBox:~$
```

@ pour dire que c'est un raccourcis

# III. 1. Systèmes: GNU/Linux « Initiation et utilisation »

## Systèmes de Fichiers

### La manipulation des fichiers

Ls - l

Liste détaillée

Il affiche une liste détaillant chaque élément du dossier

Exemple:

```
youness@youness-VirtualBox:~$ ls -l
total 36
drwxr-xr-x 2 youness youness 4096 00:14 14 أ ب ر ي ل Bureau
drwxr-xr-x 2 youness youness 4096 00:14 14 أ ب ر ي ل Document
s
drwxr-xr-x 2 youness youness 4096 01:49 14 أ ب ر ي ل Images
drwxr-xr-x 2 youness youness 4096 00:14 14 أ ب ر ي ل Modèles
drwxr-xr-x 2 youness youness 4096 00:14 14 أ ب ر ي ل Musique
drwxr-xr-x 2 youness youness 4096 00:14 14 أ ب ر ي ل Public
drwxrwxr-x 2 youness youness 4096 02:49 14 أ ب ر ي ل rep1
drwxr-xr-x 2 youness youness 4096 00:14 14 أ ب ر ي ل Téléchar
gements
drwxr-xr-x 2 youness youness 4096 00:14 14 أ ب ر ي ل Vidéos
youness@youness-VirtualBox:~$
```

Chaque colonne a sa propre signification. De gauche à droite :

- 1.droits sur le fichier** (on fera un chapitre entier pour expliquer comment fonctionnent les droits sous Linux) ;
- 2.nombre de liens physiques** (cela ne nous intéresse pas ici) ;
- 3.nom de la personne propriétaire** du fichier; c'est l'utilisateur qui a crée ce fichier.
- 4.groupe auquel appartient le fichier** Il se peut que le nom du groupe soit le même que celui du propriétaire ;
- 5.taille du fichier**, en octets ;
- 6.date de dernière modification** ;
- 7.nom du fichier** (ou dossier).

# III. 1. Systèmes: GNU/Linux « Initiation et utilisation »

## Systèmes de Fichiers

### La manipulation des fichiers

Ls - hl

Permet d'afficher la taille en Ko, Mo, Go  
Lisible par les humains

Exemple:

```
youness@youness-VirtualBox:~$ ls -hl
total 36K
drwxr-xr-x 2 youness youness 4.0K 00:14 14  أ ب ر ي ل  Bureau
drwxr-xr-x 2 youness youness 4.0K 00:14 14  أ ب ر ي ل  Document
s
drwxr-xr-x 2 youness youness 4.0K 01:49 14  أ ب ر ي ل  Images
drwxr-xr-x 2 youness youness 4.0K 00:14 14  أ ب ر ي ل  Modèles
drwxr-xr-x 2 youness youness 4.0K 00:14 14  أ ب ر ي ل  Musique
drwxr-xr-x 2 youness youness 4.0K 00:14 14  أ ب ر ي ل  Public
drwxrwxr-x 2 youness youness 4.0K 02:49 14  أ ب ر ي ل  rep1
drwxr-xr-x 2 youness youness 4.0K 00:14 14  أ ب ر ي ل  Téléchar
gements
drwxr-xr-x 2 youness youness 4.0K 00:14 14  أ ب ر ي ل  Vidéos
youness@youness-VirtualBox:~$
```

# III. 1. Systèmes: GNU/Linux « Initiation et utilisation »

## Systèmes de Fichiers

### La manipulation des fichiers

Ls - lt

trier par date de dernière modification

Exemple:

```
youness@youness-VirtualBox:~$ ls -lt
total 36
drwxrwxr-x 2 youness youness 4096 02:49 14 أ ب ر ي ل rep1
drwxr-xr-x 2 youness youness 4096 01:49 14 أ ب ر ي ل Images
drwxr-xr-x 2 youness youness 4096 00:14 14 أ ب ر ي ل Bureau
drwxr-xr-x 2 youness youness 4096 00:14 14 أ ب ر ي ل Document
s
drwxr-xr-x 2 youness youness 4096 00:14 14 أ ب ر ي ل Modèles
drwxr-xr-x 2 youness youness 4096 00:14 14 أ ب ر ي ل Musique
drwxr-xr-x 2 youness youness 4096 00:14 14 أ ب ر ي ل Public
drwxr-xr-x 2 youness youness 4096 00:14 14 أ ب ر ي ل Téléchar
gements
drwxr-xr-x 2 youness youness 4096 00:14 14 أ ب ر ي ل Vidéos
youness@youness-VirtualBox:~$
```

# III. 1. Systèmes: GNU/Linux « Initiation et utilisation »

## Systèmes de Fichiers

### La manipulation des fichiers

cat

#### *conCATenate*

Est une commande multi usages, permet d'**afficher**, de **créer**, de **copier** et de **concaténer** des fichiers.

Exemple:

```
youness@youness-VirtualBox:~$ cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
```

1. Affichage du contenu du fichier */etc/passwd*

# III. 1. Systèmes: GNU/Linux « Initiation et utilisation »

## Systèmes de Fichiers

### La manipulation des fichiers

cat

*conCATenate*

Est une commande multi usages, permet d'afficher, de créer, de copier et de concaténer des fichiers.

Exemple:

```
youness@youness-VirtualBox:~$ cat >test
Bonjour tout le monde
Bienvenu à ce cours d'initiation ;)
q:

youness@youness-VirtualBox:~$ cat test
Bonjour tout le monde
Bienvenu à ce cours d'initiation ;)
q:
```

2. Création du fichier test  
Ctrl +d : caractère de fin de fichier

Affichage du contenu du fichier test

# III. 1. Systèmes: GNU/Linux « Initiation et utilisation »

## Systèmes de Fichiers

### La manipulation des fichiers

more

#### ***MORE***

Permet d'afficher à l'écran un fichier page par page, le programme *more* est utilisé par la commande *man* pour afficher la documentation d'une commande quelconque sous linux.

**Exemple:**

```
youness@youness-VirtualBox:~$ more test
Bonjour tout le monde
Bienvenu à ce cours d'initiation ;)
q:
```

# III. 1. Systèmes: GNU/Linux « Initiation et utilisation »

## Systèmes de Fichiers

### La manipulation des fichiers

cp

#### *CoPy*

Permet de copier des fichiers; elle s'utilise sous quatre forme

#### 1. La copie d'un fichier source dans un fichier destinataire

Exemple:

```
youness@youness-VirtualBox:~$ ls
Bureau      f1  Images  Musique  Public  rep1          Vidéos
Documents  f2  Modèles projet   rep     Téléchargements
youness@youness-VirtualBox:~$ cat f2
youness@youness-VirtualBox:~$ cat f1
Bonjour tout le monde
bienvenu à ce cours d'initiation ;)
youness@youness-VirtualBox:~$ cp f1 f2
youness@youness-VirtualBox:~$ cat f2
Bonjour tout le monde
bienvenu à ce cours d'initiation ;)
youness@youness-VirtualBox:~$
```

# III. 1. Systèmes: GNU/Linux « Initiation et utilisation »

## Systèmes de Fichiers

### La manipulation des fichiers

cp

*CoPy*

Permet de copier des fichiers; elle s'utilise sous quatre forme

#### 2. La copie d'un fichier dans un répertoire

Exemple:

```
youness@youness-VirtualBox:~$ mkdir r2
youness@youness-VirtualBox:~$ cp f2 r2
youness@youness-VirtualBox:~$ ls
Bureau      f1  Images  Musique  Public  rep  Téléchargements
Documents  f2  Modèles projet   r2      rep1  Vidéos
youness@youness-VirtualBox:~$ cd r2
youness@youness-VirtualBox:~/r2$ ls
f2
youness@youness-VirtualBox:~/r2$
```

# III. 1. Systèmes: GNU/Linux « Initiation et utilisation »

## Systèmes de Fichiers

### La manipulation des fichiers

cp

*CoPy*

Permet de copier des fichiers; elle s'utilise sous quatre forme

Exemple:

3. La copie d'un répertoire dans un autre (seuls les fichiers sont copiés: on obtient un message d'erreur pour ce genre de copie).

```
youness@youness-VirtualBox:~/r2$ mkdir r3
youness@youness-VirtualBox:~/r2$ cd ..
youness@youness-VirtualBox:~$ cp r2 r3
cp: -r non spécifié ; omission du répertoire 'r2'
youness@youness-VirtualBox:~$ cp -r r2 r3
youness@youness-VirtualBox:~$ cd r3
youness@youness-VirtualBox:~/r3$ ls
f2  r3
youness@youness-VirtualBox:~/r3$ cd ..
youness@youness-VirtualBox:~$ ls
Bureau      f1  Images  Musique  Public  r3   rep1          Vidéos
Documents  f2  Modèles projet   r2      rep  Téléchargements
youness@youness-VirtualBox:~$
```

# III. 1. Systèmes: GNU/Linux « Initiation et utilisation »

## Systèmes de Fichiers

### La manipulation des fichiers

cp

*CoPy*

Permet de copier des fichiers; elle s'utilise sous quatre formes

4. **La copie récursive permet la copie d'une arborescence,**

Pour **copier** un répertoire, y compris tous ses fichiers et sous-répertoires, utilisez l'option **-R** ou **-r**. C'est une **copie** alors **récursive** qui **copie** l'**arborescence** complète avec l'ensemble des sous-dossiers.

# III. 1. Systèmes: GNU/Linux « Initiation et utilisation »

## Systèmes de Fichiers

### La manipulation des fichiers

mv

#### *MoVe*

Permet de changer le nom d'un fichier ou d'un répertoire, en première analyse cette commande est équivalente à une copie, suivie d'une suppression (couper), elle s'utilise sous deux formes:

**Exemple:**

```
youness@youness-VirtualBox:~$ cat f2
Bonjour tout le monde
bienvenu à ce cours d'initiation ;)
youness@youness-VirtualBox:~$ mv f2 f3
youness@youness-VirtualBox:~$ ls
Bureau      f1  Images  Musique  Public  r3    rep1          Vidéos
Documents  f3  Modèles projet   r2      rep  Téléchargements
youness@youness-VirtualBox:~$ cat f3
Bonjour tout le monde
bienvenu à ce cours d'initiation ;)
youness@youness-VirtualBox:~$
```

1. Transfert du fichier *fichier1* dans *fichier2* et suppression du *fichier1*. si *fichier2* existe, il est effacé.

# III. 1. Systèmes: GNU/Linux « Initiation et utilisation »

## Systèmes de Fichiers

### La manipulation des fichiers

mv

#### *MoVe*

Permet de changer le nom d'un fichier ou d'un répertoire, en première analyse cette commande est équivalente à une copie, suivie d'une suppression (couper), elle s'utilise sous deux formes:

#### Exemple:

```
youness@youness-VirtualBox:~$ ls
Bureau  f1 Images  Musique  Public  r3  rep1  Vidéos
Documents f3 Modèles projet  r2    rep  Téléchargements
youness@youness-VirtualBox:~$ cd r2
youness@youness-VirtualBox:~/r2$ ls
f2 r3
youness@youness-VirtualBox:~/r2$ cd
youness@youness-VirtualBox:~$ mv f3 r2
youness@youness-VirtualBox:~$ ls r2
f2 f3 r3
youness@youness-VirtualBox:~$
```

1. Transfert du fichier dans le répertoire avec le même nom.

# III. 1. Systèmes: GNU/Linux « Initiation et utilisation »

## Systèmes de Fichiers

### La manipulation des fichiers

mv

2. Transfert de(s) fichier(s) cité(s) dans le répertoire avec le(s) même(s) nom(s) :  
*mv fichier(s) répertoire*

Exemple:

```
youness@youness-VirtualBox:~$ mkdir projet
youness@youness-VirtualBox:~$ cd projet/
youness@youness-VirtualBox:~/projet$ touch livrable
youness@youness-VirtualBox:~/projet$ ls
livrable
youness@youness-VirtualBox:~/projet$ cd
youness@youness-VirtualBox:~$ mkdir rep
youness@youness-VirtualBox:~$ ls
Bureau      Modèles    Public     Téléchargements
Documents  Musique    rep        test_v2
Images      projet     rep1       Vidéos
youness@youness-VirtualBox:~$ cd projet/
youness@youness-VirtualBox:~/projet$ ls
livrable
youness@youness-VirtualBox:~/projet$ mv livrable /home/youness/rep
youness@youness-VirtualBox:~/projet$ cd
youness@youness-VirtualBox:~$ cd rep
youness@youness-VirtualBox:~/rep$ ls
livrable
youness@youness-VirtualBox:~/rep$ █
```

# III. 1. Systèmes: GNU/Linux « Initiation et utilisation »

## Systèmes de Fichiers

### La manipulation des fichiers

rm

#### ***ReMove***

Supprime un (ou plusieurs) fichier(s) d'un répertoire :

*rm fichier(s)*

Exemple:

```
youness@youness-VirtualBox:~$ rm test
youness@youness-VirtualBox:~$ ls
Bureau      Images      Musique     rep1        test_v2
Documents  Modèles    Public      Téléchargements  Vidéos
```

# III. 1. Systèmes: GNU/Linux « Initiation et utilisation »

## Systèmes de Fichiers

### La manipulation des fichiers

grep

#### ***GREP***

Recherche, dans un ou plusieurs fichiers, de toutes les lignes contenant une chaîne donnée de caractères.

Exemple:

Recherche de la chaîne de caractères *bonjour* dans le fichier *test*.

```
youness@youness-VirtualBox:~$ grep Bonjour test
Bonjour tout le monde
youness@youness-VirtualBox:~$
```

# III. 1. Systèmes: GNU/Linux « Initiation et utilisation »

## Systèmes de Fichiers

### La manipulation des fichiers

WC

#### *Word Count*

Cette commande permet le dénombrement des *mots*, *lignes* et *caractères* dans un fichier. Un mot est défini comme une suite de caractères précédée et suivie par des espaces, des tabulations, le début ou la fin de la ligne.

#### Syntaxe

```
wc lwcL fichier
```

l	affiche le nombre de lignes
w	affiche le nombre de mots
c	affiche le nombre de caractères
L	affiche la longueur de la ligne la plus longue
<i>fichier</i>	liste de noms de fichiers à parcourir (ou entrée standard si vide)

# III. 1. Systèmes: GNU/Linux « Initiation et utilisation »

## Systèmes de Fichiers

### La manipulation des fichiers

WC

**Exemples :** L'option par défaut : `-lwc`

- 1) Créer le fichier `program.f` , le fichier contient le texte suivant :
- 2) Format
- 3) Text format
- 4) Test vi
- 5) Linux form
- 6) Format
- 7) Form
- 8) Formater
- 9) 2 )compter le nombre de mots du fichier `program.f`
- 10)
- 11)3) Rechercher le nombre de lignes contenant le mot `format` dans le fichier `porgram.f`

# III. 1. Systèmes: GNU/Linux « Initiation et utilisation »

## Systèmes de Fichiers

### La manipulation des répertoires

WC

#### ***Word Count***

Cette commande permet le dénombrement des *mots*, *lignes* et *caractères* dans un fichier. Un mot est défini comme une suite de caractères précédée et suivie par des espaces, des tabulations, le début ou la fin de la ligne.

Impression du nombre de mots dans le fichier *test*.

#### Exemple

```
youness@youness-VirtualBox:~$ wc -w test
11 test
youness@youness-VirtualBox:~$
```

# III. 1. Systèmes: GNU/Linux « Initiation et utilisation »

## Systèmes de Fichiers

### La manipulation des répertoires

In

#### *LiNK*

Permet de désigner un fichier par plusieurs noms différents.

Le fichier *test* existe, le fichier *test\_v2* est créé sans occupation disque et est lié au fichier *test*.

#### Exemple

```
youness@youness-VirtualBox:~$ ln test test_v2
youness@youness-VirtualBox:~$ ls
Bureau      Modèles  rep1      test_v2
Documents  Musique  Téléchargements  Vidéos
Images      Public   test
youness@youness-VirtualBox:~$
```

Le fichier (en tant qu'espace disque) porte les deux noms.

# III. 1. Systèmes: GNU/Linux « Initiation et utilisation »

## Systèmes de Fichiers

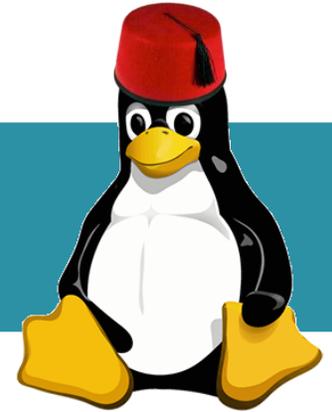
---

**TP 2: manipulation des  
commandes de base**

Chapitre

4

Systemes: GNU/Linux « Commandes avancées »



## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

124

### Edition de fichiers Les modes:

vim

<b>Touche</b>	<b>Mode</b>	<b>Usage</b>
Esc	normal	accéder à tout les autres modes
i	insertion	ajouter du texte
:	commande	entrer des commandes
v	visuel	
voir démo	sélection	
q	Ex	idem que pour le mode commande

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

125

### Edition de fichiers

### Quelques commandes

vim

<b>commande</b>	<b>usage</b>
A	ajouter en fin de ligne
u	annuler la dernière opération
ctrl+r	rétablir
yy	copier la ligne
dd	supprimer la ligne (couper)
p	coller
x	effacer le caractère
dw	effacer jusqu'à la fin du mot
diw	effacer le mot sous le curseur

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

126

### Edition de fichiers Vim tutor

vim

```
$ vim tutor
```

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

127

### Commandes avancées

#### Find, Locate et Which

- **Locate:** La commande `locate` permet de trouver très rapidement un fichier. Contrairement à ce que l'on pourrait penser **locate** ne va pas chercher le fichier au sein de l'arborescence, mais au sein d'une base de données contenant la liste des fichiers existants.
- >> mettre à jour la base: **updatedb**
- **Find:** La commande *find*, contrairement à *locate* va chercher le fichier au sein de l'arborescence
- **Which:** La commande *which* permet de localiser une commande dans le système.

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

128

### Commandes avancées

Find

#### *find*

la commande *find* permet de chercher un fichier ou dossier au sein d'une arborescence, si vous voulez par exemple chercher un fichier dont le nom est "*mon\_fichier*" dans le répertoire */home*, la commande *find* va effectuer une recherche au sein de ce répertoire et dans tous ses sous répertoires :

**La syntaxe :** *find chemin .... Expression*

#### Options :

- name** fichier : recherche fichier
- type** [b|c|d|p|f|l|s]: cherche les répertoires on peut chercher les fichiers , les liens ....par type
- perm mode** : par permissions
- links** : cherche tous les fichiers qui ont n liens
- user** nom : recherche les fichiers dont le propriétaire est nom
- mtime** n : cherche les fichiers qui ont été modifiés depuis n jours
- maxdepth levels** : limiter le nombre de sous-répertoires analysés
- size** [c|k|M|G] : cherche les fichiers dont la taille est n blocs ( un blocs =512 octets)

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

129

### Commandes avancées Retrouver un fichier ("find") :

*trouver un fichier portant un nom donné*

```
find / -name nom-de-fichier -print
```

**"/** indique que nous voulons chercher à partir de la racine de notre fichier.

**"-name "** est l'option qui indique ici que nous voulons spécifier le nom d'un fichier.

**"-print"** demande à **find** d'afficher le résultat.

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

130

### Commandes avancées Retrouver un fichier ("find") :

*trouver un fichier portant un nom donné* et supprimer le

```
find / -name fichier1 -exec rm {} \;
```

"/" indique que nous voulons chercher à partir de la racine de notre fichier.

"-name " est l'option qui indique ici que nous voulons spécifier le nom d'un fichier.

"-exec » demande à **find** d'exécuter la commande rm (suppression).

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

131

### Commandes avancées Opérateurs conditionnels:

- -a : ET

```
$ find . \( -perm -g=w -a -group root \)
```

- -o : OU

```
$ find . \( -name '*.tex' -o -name '*.dvi' \)
```

- ! : négation

```
$ find . ! -user root
```

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

132

### Commandes avancées

#### Exercices:

1. Rechercher les fichiers de l'utilisateur **test** qui ont été modifiés y a plus de 20 jours
2. Rechercher , dans toute l'arborescence tous les répertoires de taille supérieure à 300 blocs , puis afficher à l'écran le résultat de la recherche.
3. Rechercher tous les fichiers sous le répertoire **/etc** dont les noms commencent par rc,
4. Rechercher tous les sous-répertoires de */etc*,
5. Rechercher tous les fichiers réguliers se trouvant sous votre répertoire d'accueil et qui n'ont pas été modifiés dans les 10 derniers jours

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

133

### Commandes avancées

locate

**Retrouver un fichier (La commande locate ):**

La commande **locate** a la même mission que **find**

**Find** cherche dans l'arborescence

**Locate** cherche plus rapidement dans une base de données des fichiers existants : **/usr/lib/** se nomme **locatedb**

La syntaxe de la commande est simple:

**\$ locate nom\_du\_fichier.**

**locate** est un complément de **find**.

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

134

### Commandes avancées **Autres Implémentations:**

- mlocate
  - plus moderne
  - fonctionnement identique
- slocate
  - sécurisé
  - empêche les utilisateurs de voir les noms de fichiers auxquels ils n'ont pas accès
- le plus souvent, locate est un lien symbolique vers un de ces programmes

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

135

### Commandes avancées

locate

**Retrouver un fichier (La commande locate ):**

**\$ locate nom\_du\_fichier.**

**locate** est un complément de **find**.

- Options :
  - -i : ignorer la casse
  - -r *regex* : utiliser une expression régulière
  - -c : n'afficher que le nombre de fichiers correspondant

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

136

### Commandes avancées

whereis

```
$ whereis [-bms] filename.
```

- Options :
  - -b : seulement les fichiers binaires
  - -m : seulement les pages man
  - -s : seulement les sources

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

137

### Commandes avancées

Which

La commande **which** affiche le chemin complet des commandes (shell)

- La commande **which** peut prendre un ou plusieurs arguments.
- Pour chacun d'entre eux elle affiche sur la sortie standard (stdout) le chemin complet des exécutable qui auraient pu être saisis directement depuis le prompt Shell.
- La commande **which** recherche l'exécutable dans les différents répertoires.

- **Exemple:**

```
#Which ls  
#Which ls du mv
```

- Afficher tous les chemins vers la commande **ls**

```
#Which -a ls
```

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

138

### Commandes avancées

Type

```
$ type commande
```

- affiche comment une commande est interprétée
- Options :
  - -a : pour toutes les correspondance
  - -t : sortie raccourcie

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

139

### Commandes avancées

grep

*Trouver du texte dans un fichier*

- La commande **grep** cherche une expression rationnelle dans un ou plusieurs fichiers;
- La commande **grep** affiche la ligne qui contient le mot recherché;
- La commande **grep** est souvent utilisé en tube avec d autres commande;

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

140

### Commandes avancées

grep

```
$ grep [options] regexp [fichier...]
```

#### Les options :

- c Imprime le nombre de lignes comprenant la chaîne de caractères dans le fichier.
- h Cette option ne considère pas le nom du fichier comme valide. Par exemple, si le nom du fichier contient la chaîne de caractères, il ne sera pas affiché à l'écran.
- i Ignore la distinction entre les majuscules et les minuscules.
- l Imprime que le nom du fichier contenant la chaîne de caractères. Il n'y a pas de répétitions.
- n Chaque ligne trouvée est précédée par son numéro de ligne correspondant.
- v Imprime les lignes du fichier qui ne contiennent pas la chaîne de caractères.

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

141

### Commandes avancées

grep

**Remarque : dans tous les cas combinez grep à ls par mécanisme de tube**

#### Exercice

1. Créez un répertoire tpgrep
2. Dans ce répertoire , créez les fichiers suivant : grep1 , GREP1 , grep2 , GREP2 , grep3 , GREP3 , grep4 et GREP4
3. Trouver le nombre de fichier grep1 ?
4. Afficher tous les fichiers grep1 ( sans tenir compte de la casse )
5. Afficher tous les fichier commençant par gr ou GR

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

142

### Commandes avancées

gzip

**Le compactage et le décompactage des fichiers au format .gz**

Pour compacter un fichier, taper la commande suivante :

**gzip non\_du\_fichier**

Pour décompacter un fichier, taper la commande suivante :

**gzip -d non\_du\_fichier.gz**

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

143

### Commandes avancées

Head, tail & less

- Afficher les premières lignes d'un fichier

```
$ head [options] [fichier...]
```

- Afficher les dernières lignes d'un fichier

```
$ tail [options] [fichier...]
```

- Afficher un fichier page par page

```
$ less [fichier...]
```

#### Exercice

Afficher les 3 premières lignes du fichier fichiertest

Afficher les 3 dernières lignes du fichier fichiertest

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

144

### Commandes avancées

sort

lire et trier le contenu du fichier

**Syntaxe : sort [-option] [+c1 [-c2]] fichier**

C1 et c2 sont des numéros de champ:

+C1 : permet d'exclure du tri les champs de 1 à c1

-C2 : permet d'exclure du tri les champs qui suivent le champ c2

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

145

### Commandes avancées

sort

-c : test si l'entrée est déjà triée

-r : tri inverse

#### Les options :

-u : élimine les lignes identiques

-n : tri numérique

-d : ordre alphabétique

-o : résultat du tri dans un autre fichier

-tx : choix du séparateur de champ x au lieu de la valeur par défaut ( espace ou tabulation )

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

146

### Commandes avancées

sort

#### Exemples:

1. Tri du fichier `essai` et mettre le fichier trié dans le fichier `essai.tri` ( il faut créer `essai` si il n'existe pas )
2. Tri du fichier `/etc/passwd` sur le numéro d'utilisateur (UID) qui est le 3eme champ de chaque ligne du fichier , le résultat du tri sera dans le fichier `/etc/passwd.tri`

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

147

### Commandes avancées

sort

1. **Sort -o essai.tri essai**
2. **Sort -t: +2n -3 -o /etc/passwd.tri /etc/passwd**

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

148

### Commandes avancées

cal

**affiche le calendrier de l'année spécifiée en paramètre**

***syntaxe : cal mois année***

**Exempe :**

cal 2010 : affiche le calendrier de tous les mois de l'année 2010

cal 12 3000 : affiche le calendrier du mois de décembre de l'année 3000

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

149

### Commandes avancées

date

*1. Retourne la date courante*

*2. Modifie la date : date MMDDhhmmYY.ss*

MM est le mois,  
DD le jour,  
hh l'heure,  
mm les minutes,  
YY l'année  
ss les secondes, tous sur deux chiffres.

Exemple : date 0830145008.00

Tue Aug 30 14:50:00 CEST 2008

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

150

### Commandes avancées

echo

*commande d'affichage*

id

*retourne le nom de login, l'UID, le groupe et le GID*

finger

retourne des informations de l'utilisateur passé en argument.

uname

Affiche, dans l'ordre, le nom du système d'exploitation, le nom et la release du système de la machine.

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

151

### Les méta-caractères \* ? et []

Les méta-caractères permettent de faire une sélection de fichiers suivant certains critères.

\*

- \* est Le méta caractère le plus fréquemment utilisé , il remplace une chaîne de longueur non définie.

**Exemple :**

**\***, tous les fichiers

**a\*** : tous les fichiers commençant par **a**

**\*a** : tous les fichiers se terminant par **a**

**\*a\*** : fichiers qui ont le caractère **a** dans leur nom, quelque soit sa place.

**rm \*.txt** : supprimer tout les fichiers finissant par txt

**ls \*11\*** : l'étoile en début et en fin de chaîne indique à l'interpréteur de lister tous les fichiers et répertoires **comportant** la chaîne "11".

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

152

### Les méta-caractères

#### Exemples :

- Lister tous les fichiers qui commencent par la lettre a
- Supprimer tous les fichiers qui se terminent par rmp
- Déplacer tous les fichiers qui ont l'extension .htm

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

153

### Les méta-caractères \* ? et []



**?** : Extraire un caractère quelconque non vide ; remplace un caractère unique

Le point d'interrogation ? permet d'extraire d'un flux de données n'importe quel caractère non vide, mais un seul

#### Exemples

**a??** : sélectionner les fichiers dont le nom commence par **a**, mais qui contiennent au total trois caractères, exactement.

**rm ?** : supprimer **tous** les dossiers et fichiers du répertoires dont le nom contient qu'**un seul caractère**, quelqu'il soit

**rm ???*.txt*** : supprimer tous fichiers **finissant** par la chaîne "*.txt*", ne contenant que trois caractères avant l'extension

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

154

### Les méta-caractères \* ? et []

[]

[] représente une série de caractères

- Le critère **[aA]\*** permet la sélection des fichiers dont le nom commence par un **a** ou **A** (minuscule ou majuscule) .
- Le critère **[a-d]\*** fait la sélection des fichiers dont le nom commence par **a** jusqu'à **d**.
- Le critère **\*[de]** fait la sélection des fichiers dont le nom se termine par **d** ou **e**.

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

155

### Les méta-caractères \* ? et []

#### Résumé

- Ils permettent de créer des noms génériques;
- \*Correspond à une chaîne de caractères quelconque (même vide);
- ?Correspond à un caractère quelconque;
- [**abc...**] Correspond à un des caractères spécifiés dans la liste. Il est possible de décrire des listes de caractères en ne précisant que le premier caractère et le dernier caractère de la liste séparé par un moins (par exemple : a-z, I-M, 0-9);
- [**!abc...**]Correspond à tout caractère non spécifié dans la liste.

#### Exemples

- **ls toto\*** Liste les fichiers **toto**, **toto.c** et **toto.txt**.
- **cat prog?.c** Affiche le contenu du fichier **prog2.c**
- **ls [A-D]\*** Liste les fichiers commençant par **A**, **B**, **C** ou **D**.

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

156

### Les méta-caractères \* ? et []

**Exercice :** Créer dans un répertoire rep1 les fichiers suivants : *fich1*, *fich2*, *fich11*, *fich12*, *fich1a*, *ficha1*, *.fich1*, *.fich2*, *toto*, *afich*.

- Lister les fichiers :
  1. dont les noms commencent par *fich*,
  2. dont les noms commencent par *fich* suivi d'un seul caractère,
  3. dont les noms commencent par *fich* suivi d'un chiffre,
  4. dont les noms commencent par point,
  5. dont les noms ne commencent pas par *f*,
  6. dont les noms contiennent *fich*.

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

157

### Les méta-caractères \* ? et []

#### Corrigé :

```
mkdir rep1; cd rep1; touch fich1 fich2 fich11 fich12 fich1a ficha1 .fich1 .fich2 toto afich
```

- Lister les fichiers :
  - 1/ dont les noms commencent par fich, • **ls fich\***
  - 2/ dont les noms commencent par fich suivi d'un seul caractère, • **ls fich?**
  - 3/ dont les noms commencent par fich suivi d'un chiffre, • **ls fich[0-9]**
  - 4/ dont les noms commencent par point, • **ls .??\*** (ne pas prendre . Et ..)
  - 5/ dont les noms ne commencent pas par f, • **ls [!f]\***
  - 6/ dont les noms contiennent fich. • **ls \*fich\***

# IV. 1. Systèmes: GNU/Linux « Commandes avancées »

158

## Commandes avancées

### 1. La ligne de commandes séquentielles

- ❑ Il est possible de taper plusieurs commandes sur la même ligne en les séparant par des **points-virgules**.
- ❑ Les commandes sont **exécutées séquentiellement**, de façon totalement indépendante, la première **n'influençant pas la seconde** et ainsi de suite.

Exemple :

```
youness@Ubuntu:~/Bureau$ pwd; who; ls /var/log
/home/youness/Bureau
youness :0      2022-04-21 11:08 (:0)
alternatives.log  dpkg.log      syslog
alternatives.log.1 dpkg.log.1    syslog.1
apt               faillog       syslog.2.gz
auth.log          fontconfig.log syslog.3.gz
auth.log.1        gdm3          syslog.4.gz
auth.log.2.gz     gpu-manager.log ubuntu-advantage.log
auth.log.3.gz     hp            ubuntu-advantage-timer.log
boot.log          installer     ubuntu-advantage-timer.log.1
boot.log.1        journal       unattended-upgrades
bootstrap.log     kern.log      vboxadd-install.log
btm               kern.log.1    vboxadd-setup.log
btm.1             kern.log.2.gz vboxadd-setup.log.1
cups              kern.log.3.gz vboxadd-setup.log.2
dist-upgrade      lastlog       vboxadd-setup.log.3
dmesg             openvpn       wtmp
dmesg.0           private
dmesg.1.gz        speech-dispatcher
youness@Ubuntu:~/Bureau$
```

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

159

### Commandes avancées

#### 2. Les séparateurs conditionnels de commandes

Il est possible de contrôler la séquence d'exécution de commandes en utilisant des séparateurs conditionnels.

- ❑ Le séparateur `&&` permet d'exécuter la commande qui le suit **si et seulement si** la commande qui le précède **a été exécutée sans erreur**.
- ❑ Le séparateur `||` permet d'exécuter la commande qui le suit **si et seulement si la commande qui le précède a été exécutée avec erreur** (code retour du processus différent de 0).

Exemple :

```
youness@Ubuntu:~$ cd rep1/ && rm *
youness@Ubuntu:~/rep1$ ls
youness@Ubuntu:~/rep1$
```

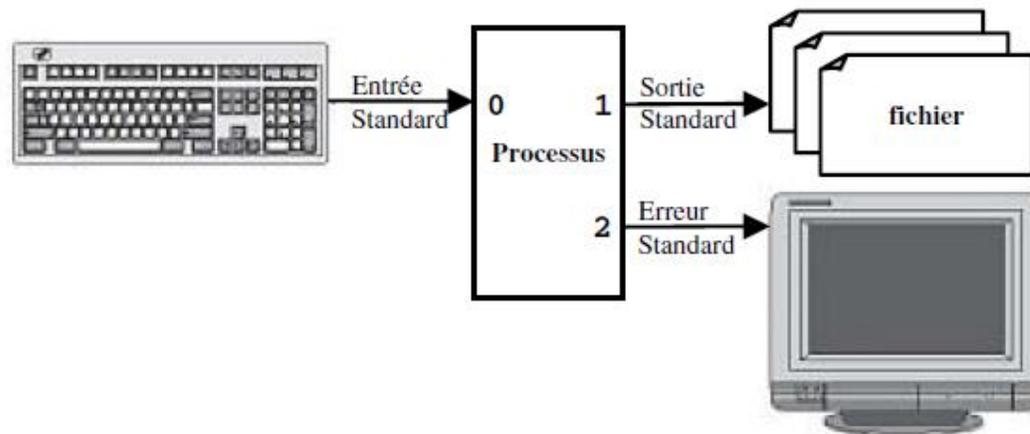
```
youness@Ubuntu:~/rep1$ cd project1 || rm *
bash: cd: project1: Aucun fichier ou dossier de ce type
rm: impossible de supprimer '*': Aucun fichier ou dossier de ce type
```

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

160

### 3. La redirection des entrées-sorties

- ❑ On appelle processus, ou tâche, l'exécution d'un programme exécutable.
- ❑ Au lancement de chaque processus, l'interpréteur de commandes ouvre d'office une entrée standard (**par défaut le clavier**), une sortie standard (**par défaut l'écran**) et la sortie d'erreur standard (**par défaut l'écran**).
- ❑ Ces entrées-sorties standard peuvent être redirigées vers un *fichier*, un *tube*, un *périphérique*.
- ❑ La redirection de la sortie standard consiste à renvoyer le texte qui apparaît à l'écran vers un fichier.



# IV. 1. Systèmes: GNU/Linux « Commandes avancées »

## 3. La redirection des entrées-sorties

La redirection des sorties peut être réalisée par :

- Effacement et création du fichier;
  - Ou par ajout à la fin du fichier si ce dernier existe;
  - Dans le cas contraire, un nouveau fichier sera créé;
  - Dans le cas de la redirection de l'entrée, le fichier doit exister.
- Le caractère **<** suivi du nom d'un fichier indique la redirection de l'entrée standard à partir de ce fichier : `<fe` Définition de *fe* comme fichier d'entrée standard.
  - Le caractère **>** suivi du nom d'un fichier indique la redirection de la sortie standard vers ce fichier :

<b>&gt;</b>	<b>stdout vers nouveau fichier</b>
<b>&gt;&gt;</b>	stdout à la suite d'un fichier
<b>2&gt;</b>	stderr vers nouveau fichier
<b>2&gt;&gt;</b>	stderr à la suite d'un fichier
<b>&amp;&gt;</b>	stdout + stderr
<b>&lt;</b>	stdin depuis un fichier
<b>&lt;&lt;</b>	stdin à partir d'une chaîne de caractères
<b>&lt;&gt;</b>	stdin et stdout vers et depuis le même fichier



**stdin (le clavier), stdout (l'écran) et stderr (la sortie des messages d'erreur vers l'écran)**

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

162

### Commandes avancées

**tee**

**Copier l'entrée standard sur la sortie standard et dans un fichier.**

- ❑ En utilisant la commande **Tee** Linux, nous recevons un résultat dans le terminal, que nous pouvons diriger vers une autre commande pour le traiter.
- ❑ La commande suivante listera les fichiers à l'intérieur du dossier et en utilisant la première pipe, le résultat sera écrit dans le fichier test.txt et le même résultat sera transmis à la troisième commande – **grep** pour identifier les fichiers contenant la chaîne de caractères **ch** :

#### Exemple

```
ls | tee test.txt | grep 'ch'
```

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

163

### Commandes avancées

#### 3.1 La commande `cat` et les redirections

- La commande ***cat*** est une commande **multi-usage** qui permet d'afficher, de créer, de copier et de concaténer des fichiers.
- Elle utilise pleinement les mécanismes de redirection.
- Elle lit l'entrée standard si aucun fichier n'est spécifié.
- Ce qui est lu est affiché sur la sortie standard.

##### *a. Lecture sur clavier et écriture sur écran*

```
youness@Ubuntu:~/rep1$ cat
Bonjour tout le monde ;)
Bonjour tout le monde ;)

youness@Ubuntu:~/rep1$ █
```

*Le texte « bonjour tout le monde ☺, » est lu du clavier et est affiché à l'écran.  
La combinaison de touches <ctrl d> interrompt la saisie.*

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

164

### 3.1 La commande cat et les redirections

#### *b. Copie d'un fichier*

```
youness@Ubuntu:~/rep1$ cat >f1
Hi
youness@Ubuntu:~/rep1$ cat >f2
Hello
youness@Ubuntu:~/rep1$ cat f1 >f2
youness@Ubuntu:~/rep1$ cat f2
Hi
youness@Ubuntu:~/rep1$ cat f1
Hi
youness@Ubuntu:~/rep1$
```

*Le fichier f1 est copié dans f2, et le contenu de f2 est écrasé.*

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

165

### 3.1 La commande cat et les redirections

#### *c. Concaténation des fichiers*

```
youness@Ubuntu:~/rep1$ cat >f1
Hi
youness@Ubuntu:~/rep1$ cat >f2
Hello
youness@Ubuntu:~/rep1$ cat >f3
Bonjour
youness@Ubuntu:~/rep1$ cat f1 f2 f3 >f123
youness@Ubuntu:~/rep1$ cat f123
Hi
Hello
Bonjour
youness@Ubuntu:~/rep1$ cat f1
Hi
youness@Ubuntu:~/rep1$ cat f2
Hello
youness@Ubuntu:~/rep1$ cat f3
Bonjour
youness@Ubuntu:~/rep1$
```

*Le fichier f123 contiendra la concaténation des fichiers f1, f2 et f3, dans cet ordre.*

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

166

### 3.1 La commande cat et les redirections

#### *d. Ajout d'un fichier*

```
youness@Ubuntu:~/rep1$ cat >f4
Salut tout le monde
youness@Ubuntu:~/rep1$ cat f123
Hi
Hello
Bonjour
youness@Ubuntu:~/rep1$ cat f4>>f123
youness@Ubuntu:~/rep1$ cat f123
Hi
Hello
Bonjour
Salut tout le monde
youness@Ubuntu:~/rep1$
```

*Le fichier f4 est concaténé à la suite du fichier f123. f123 est créé s'il n'existe pas.*

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

167

### Commandes avancées

#### 3.1 La commande cat et les redirections

*d. Création d'un fichier par saisie au clavier*

```
Youness> cat >f1  
Bonjour  
Ctrl+D
```

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

168

### Commandes avancées

#### 4. Les tubes de communication et les filtres

##### *a. Les tubes*

- Un **tube** (**pipe** en anglais) est un flot de données qui permet de relier la sortie standard d'une commande à l'entrée standard d'une autre commande sans passer par un fichier temporaire.

Dans une ligne de commandes, le tube est formalisé par la barre verticale **|**, que l'on place entre deux commandes :

P1 | P2 | P3

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

169

### Commandes avancées

#### 4. Les tubes de communication et les filtres

##### a. Les tubes

Affichage page par page du contenu du répertoire courant :

```
ls -l | less
```

Dans cet exemple, le résultat de la commande **ls -l** n'apparaît pas à l'écran :

la sortie standard est redirigée vers l'entrée standard de la commande **less** qui, quant à elle, affichera son entrée standard page par page. La commande **less**, contrairement à la commande **more**, permet à l'aide des touches claviers, (flèche en bas et celui en haut) , <page-up> et <page-down> de monter et descendre dans le flot de données obtenu dans un tube.

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

170

### Commandes avancées

#### 4. Les tubes de communication et les filtres

##### *a. Les tubes*

```
youness@Ubuntu:~$ ls -l |grep rwxr-xr-x |cat
drwxr-xr-x 2 youness youness 4096 avril 20 23:37 Bureau
drwxr-xr-x 2 youness youness 4096 avril 20 23:37 Documents
drwxr-xr-x 2 youness youness 4096 avril 20 23:37 Images
drwxr-xr-x 2 youness youness 4096 avril 20 23:37 Modèles
drwxr-xr-x 2 youness youness 4096 avril 20 23:37 Musique
drwxr-xr-x 2 youness youness 4096 avril 20 23:37 Public
drwxr-xr-x 2 youness youness 4096 avril 20 23:37 Téléchargements
drwxr-xr-x 2 youness youness 4096 avril 20 23:37 Vidéos
youness@Ubuntu:~$ █
```

*Affichage à l'écran du contenu du répertoire courant dont les protections sont **rwxr-xr-x***

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

171

### Commandes avancées

#### 4. Les tubes de communication et les filtres

##### *a. Les tubes*

```
youness@Ubuntu:~$ who | wc -l  
1  
youness@Ubuntu:~$ ls | wc -w  
19  
youness@Ubuntu:~$
```

*La première commande indique le nombre de personnes connectées au système.*

*La deuxième indique le nombre de fichiers dans le répertoire courant*

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

172

### Commandes avancées

#### 4. Les tubes de communication et les filtres

##### *b. Les filtres*

- Dans les exemples précédents, nous voyons apparaître, à travers les commandes **WC**, **less** et **grep**, une famille particulièrement importante de commandes Linux : les **filtres**.
- Un filtre est une commande qui lit les données sur l'entrée standard, les traite et les écrit sur la sortie standard.
- Le concept de tube, avec sa simplicité, devient un outil très puissant dans Linux qui propose un choix très vaste de filtres. Les filtres les plus utilisés sont les suivants :
  - **grep** recherche les occurrences d'une chaîne de caractères.
  - **wc** compte le nombre de caractères (ou octets), mots et lignes.
  - **less** affiche son entrée standard page par page.
  - **sort** filtre de tri.

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

173

### Commandes avancées

#### 5. La substitution de commandes

- La substitution de commande ou *backquoting* permet d'utiliser le résultat d'une commande comme argument d'une autre commande.
- Pour utiliser cette fonctionnalité, il faut entourer la commande soit d'accents graves, ou « backquotes » ( `commande` ), soit \$(commande). La forme **\$(commande)** est plus récente et doit être préférée.
- La commande placée entre **\$(cmd)** est exécutée en premier, avant l'exécution de la ligne de commandes dont elle fait partie.
- Son résultat, c'est-à-dire la sortie standard, est entièrement intégré à la ligne de commandes en remplacement de la commande « backquotée ».
- La ligne de commandes est alors exécutée avec ces nouveaux arguments.

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

174

### Commandes avancées

#### 5. La substitution de commandes

La commande ***echo*** affiche à l'écran la chaîne de caractères qui la suit.

Exemple 1:

```
youness@Ubuntu:~$ echo pwd
pwd
youness@Ubuntu:~$ echo bonjour SMIS6
bonjour SMIS6
youness@Ubuntu:~$ █
```

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

175

### Commandes avancées

#### 5. La substitution de commandes

Exemple 2:

```
youness@Ubuntu:~$ echo pwd
pwd
youness@Ubuntu:~$ echo $(pwd)
/home/youness
youness@Ubuntu:~$
```

Dans cet exemple, la commande *pwd* qui indique le répertoire courant est exécutée.

Son résultat devient l'argument de la commande *echo* (*echo /home/youness*).

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

176

### Commandes avancées

#### 5. La substitution de commandes

Exemple 3:

```
youness@Ubuntu:~$ echo il y a $(who | wc -l) user connecté
il y a 1 user connecté
youness@Ubuntu:~$ echo il y a $(who | wc -l) user connecté qui est $(pwd)
il y a 1 user connecté qui est /home/youness
youness@Ubuntu:~$
```

Cette commande nous indique le nombre d'utilisateurs connectés puis il les affiche.

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

177

### Commandes avancées

#### 6. Les commandes groupées

- La commande groupée est une succession de commandes séparées par le caractère `;` et considérées comme un ensemble.
- Cet ensemble, repéré par des parenthèses (...), est exécuté par un nouveau processus **shell**.
- Les commandes seront exécutées séquentiellement, sans influence les unes sur les autres.
- Le résultat d'une commande groupée est cependant différent de celui qu'auraient les mêmes commandes réalisées séquentiellement.

Exemple :

```
youness@Ubuntu:~$ (cd rep1; rm f1)
youness@Ubuntu:~$ ls ./rep1
f123  f2  f3  f4
youness@Ubuntu:~$
```

Suppression du fichier *f1* dans le répertoire *rep1*.

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

178

### Commandes avancées

#### Archivage et compression

- **tar** (tape archiver) est un outil de manipulation d'archives sous les systèmes Unix et les dérivés dont Linux.
- Il ne compresse pas les fichiers, mais les concatène au sein d'une seule et même archive.
- La majorité des programmes linux utilisent ce système d'archivage.

#### Trois fonction principales:

1. Création de l'archive avec l'option `-c`
2. Lister le contenu de l'archive avec l'option `-t`
3. Extraire l'arborescence de l'archive avec l'option `-x`

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

179

### Commandes avancées

#### Archivage et compression

#### Compresser des fichiers

Pour compresser un **répertoire entier** ou bien un **seul fichier**, entrez la commande suivante :

```
tar -czvf nom-de-l-archive.tar.gz /chemin/vers/répertoire-ou-fichier
```

- c** : crée un archive.
- z** : compresser l'archive avec gzip.
- v** : mode verbose, affiche la progression.
- f** : permet de spécifier le nom du fichier d'archive.

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

180

### Commandes avancées

#### Archivage et compression

#### Compresser plusieurs fichiers à la fois

La commande ***tar*** peut également être utilisée pour compresser plusieurs dossiers et fichiers **en même temps**. Il suffit simplement de saisir une liste de fichiers et de dossiers à compresser au lieu d'un seul. Par exemple, pour compresser le dossier /home/youness/reptp1, le fichier ~/Téléchargements/photo.jpg et le fichier ~/Documents/sample.pdt, vous devez saisir la commande suivante :

```
tar -czvf archive.tar.gz /home/youness/reptp1 ~/Téléchargements/photo.jpg ~/Documents/sample.odt
```

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

181

### Commandes avancées

#### Archivage et compression    Exclure des dossiers/fichiers

Pour compresser un répertoire **en excluant** certains fichiers ou dossiers, vous pouvez utiliser l'option ***--exclude*** pour chaque dossier et fichier à exclure.

Par exemple, pour compresser le répertoire /home/youness/distributions mais **sans les sous-dossiers** /home/mzaydi/Ubuntu et /home/youness/Fedora , utilisez la commande suivante :

```
tar -czvf archive.tar.gz /home/youness/distributions --exclude=//home/youness/distributions /Ubuntu --exclude=/home/mzaydi/distributions/Fedora
```

## Commandes avancées

### Archivage et compression    Décompresser une archive

La commande ***tar*** est aussi capable d'**extraire** (décompresser) une archive.

Par exemple, pour extraire le contenu du fichier ***archive.tar.gz*** dans le **répertoire courant**, entrez la commande suivante :

```
tar -xzvf archive.tar.gz
```

**-x** qui indique à **tar** d'extraire une archive au lieu d'en créer une (-c).

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

183

### Commandes avancées

#### Archivage et compression

1. **tar** est un logiciel d'archivage qui permet de combiner plusieurs fichiers en un seul.
2. **gzip** est un logiciel de compression utilisé pour réduire la taille d'un fichier.
3. **tar** et **gzip** sont utilisés ensemble pour créer des archives compressées.
4. **.tar** : fichier d'archive non compressé.
5. **.gz** : fichier (archive ou non) compressé avec gzip.
6. **.tar.gz** : fichier d'archive compressé avec gzip.

## IV. 1. Systèmes: GNU/Linux « Commandes avancées »

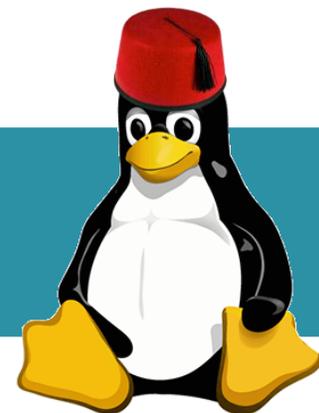
184

### **TP 3: Manipulation des commandes avancées**

Chapitre

5

Programmation Shell



# V. Programmation Shell

## Scripting

- ❑ Le *shell* est plus qu'un interpréteur de commande , c'est un puissant langage de programmation avec notamment une gestion des variables, des tests et des boucles, des opérations sur variables, des fonctions...
- ❑ Sous linux, un fichier contenant des commandes est appelé **script**

**Dans le reste de ce chapitre on utilisera :**

- ❑ **Script** : Un fichier contenant des commandes pour le shell est appelé un script
- ❑ **Shell**: l'interpréteur de commande + le langage correspondant

# V. Programmation Shell

## Variables d'environnement

Comme tout langage de programmation, le **shell** comporte des instructions et des variables

Tout jeu d'instructions comporte :

- Toutes les commandes Linux;
- L'invocation de programmes exécutables (ou de scripts) avec passage de paramètres;
- Des instructions d'assignation de variables;
- Des instructions conditionnelles et itératives;
- Des instructions d'entrée – sortie;

**NB: le mécanisme de redirection et des tubes sont utilisés dans les scripts**

# V. Programmation Shell

## Remarques :

1. Le Shell est un langage interprété ( pas besoin de recompiler les scripts à chaque modification )
2. Possibilité d'utiliser plusieurs **Shell** en même temps ( écrire le jeu d'instructions d'un script dans un **Shell** , et exécuter en un autre script sur un autre **Shell** )
3. Si un script commence par la ligne :

***#!/bin/xxx*** ( chemin d'accès du shell xxx qui doit interpréter ce script )

Il est interprété par le Shell /bin/xxx

# V. Programmation Shell

## La programmation de base en shell

- ❑ Par convention les shells scripts se terminent généralement (pas obligatoirement) par « **.sh** » pour le Bourne Shell et le Bourne Again Shell, par « **.ksh** » pour le Korn Shell et par « **.csh** » pour le C Shell.
- ❑ Dans ce qui suit on suppose que l'environnement utilisateur est le bash
- ❑ Les bases de programmation exposées sont communes à tous les interpréteurs de commande

**Attention** : toujours commencer un shell script par la ligne :

```
#!/bin/bash
```

# V. Programmation Shell

## Les variables

### Variables d'environnement

- ❑ **Une variable** est un emplacement de stockage de valeur qui peut être un nom de fichier, numéro, texte ou toutes autres données.
- ❑ Elle est désignée par son nom symbolique lors de sa création;
- ❑ La valeur de la variable peut être affichée, supprimée, éditée et/ou enregistrée.
- ❑ **Une variable d'environnement:**
  - Sont des variables ont des valeurs dynamiques qui affectent les processus ou les programmes sur un ordinateur.
  - Existent dans tous les OS;
  - Permettent de créer, d'éditer et de supprimer des valeurs;
  - Donnent des informations sur le comportement du système.

# V. Programmation Shell

## Les variables : **Variables d'environnement**

Pour afficher le contenu de la variable on utilise la commande: **echo \$NOM\_VARIABLE**

Variables	description
<b>HOME</b>	contient le répertoire d'utilisateur
<b>USER</b>	contient le login d'utilisateur
<b>PWD</b>	contient le répertoire courant
<b>SHELL</b>	contient le nom du shell de connexion
<b>PATH</b>	contient la liste des répertoires où se trouvent les commandes que l'utilisateur peut exécuter
<b>HOSTNAME</b>	contient le nom de la machine
<b>HISTSIZE</b>	contient la taille maximale des commandes exécutées contenues dans le fichier historique
<b>PS1</b>	contient les paramètres d'affichage de l'invite de commande (le prompt)

**printenv** : permet d'afficher toutes les variables d'environnement disponibles.

# V. Programmation Shell

Combinaison	Effet
\u	nom de l'utilisateur courant
\h	nom de la machine
\W	nom du répertoire courant
\S	privilège de l'utilisateur courant
\d	date courante (au format lun. janv. 1)
\w	chemin complet du répertoire de travail
\A	heure format 24h sans les secondes heure format 24h avec les secondes
\@	heure format 12h sans les secondes
\T	heure format 12h avec les secondes
\D{%d-%m-%Y %H:%M:%S%z}	Date et heure dans un format personnalisable
\j	nombre de tâches en cours dans le terminal
\#	numéro de la commande dans l'historique
\v	version de bash
	nouvelle ligne

## Variable PS1

- Pour modifier le contenu de la variable \$PS1, en choisissant un format

**Nom\_d'utilisateur@Nom\_de\_la\_machine heure>**

- 

```
export PS1="\u@\h \A> "
```

- La modification est temporaire;
- Pour que la modification soit permanente:

➔ Modifier le fichier **/etc/bash.bashrc** et repérez la ligne commençant par PS1=...

## V. Programmation Shell

### Les alias :

Les alias sont des substitutions abrégées de commandes répétitives et/ou longues à taper dans la console.

Il est possible de définir les alias dans deux fichiers cachés qui se trouvent dans le *Dossier Personnel*:

- dans le fichier **bash.bashrc** juste après la ligne "*some more ls aliases*"
- dans un fichier **bash.bash\_aliases**. Et si ce dernier n'existe pas, il faut le créer.

# V. Programmation Shell

## Les alias: exemple

- Pour afficher le contenu d'un dossier, y compris les fichiers cachés, avec un format des noms de fichiers long et rendu facilement compréhensible pour les humains, il faut utiliser la commande suivante:

```
ls -alh /dossier
```

- Créer un alias appelé llf (pour list long format):

```
alias llf='ls -alh'
```

- Désormais, lorsque nous saisissons **llf** à l'invite de commande, bash comprendra qu'il s'agit d'un alias qui pointe vers **ls -alh**.
- Les **aliases** ne sont valables que pour la session courante.
- Il suffit de les ajouter au **bash.bashrc**.

```
youness@Ubuntu:~$ alias llh='ls -alh'
youness@Ubuntu:~$ llh
total 140K
drwxr-xr-x 20 youness youness 4,0K mai 25 00:26 .
drwxr-xr-x 3 root root 4,0K avril 20 23:17 ..
-rw-rw-r-- 1 youness youness 0 mai 19 10:49 abc
-rw-rw-r-- 1 youness youness 2,6K mai 25 01:16 test
```

# V. Programmation Shell

## Les variables :

### Créer un script

1-Créer le fichier script.sh

**#vim script.sh**

2-Dans le fichier, définir le shell utilisé **#!/bin/bash**

3-Mettre des commandes shell ou un commentaire

**#Afficher les dossiers**

**Ls**

```
#!/bin/bash
#Afficher les dossiers
ls
```

L'extention **.sh** c'est juste par convention

**#!** C'est le **shebang** (en-tête d'un fichier texte **qui** indique au système d'exploitation (de type Unix) que ce fichier n'est pas un fichier binaire mais un **script**).

(une ligne commençant par **#** )

commentaire

affiche le dossier

# V. Programmation Shell

## Créer un script :

Créer avec l'éditeur **vi** , **nano** ou **gedit** le fichier **premierscript.sh**

**Echo** est une commande qui permet d'afficher tout ce qui suit la commande

Exécuter le script

```
#!/bin/bash
```

```
Echo bonjour tout le monde , ceci est un premier script
```

# V. Programmation Shell

## Exécuter un script :

- Un script peut être exécuté en mode mise au point grâce à la commande sh :

- **sh -x premierscript.sh** mode trace
- **sh -v premierscript.sh** mode verbose

```
youness@Ubuntu:~$ chmod +x script.sh
youness@Ubuntu:~$ ./script.sh
abc      essai.tri  Public  script1.sh  TEST1    test4
Bureau   Images    r1      script.sh   TEST2    TEST4
Documents Modèles   r2      sortie     test2.gz test.txt
E        Musique   rep1    Téléchargements test3    Vidéos
essai    passwd.tri rep2    test1      TEST3
```

- demander au shell de **tracer** le déroulement du script ou de le **commenter**.
- **Bash premierscript.sh** (spécifier le shell utilisé, dans ce cas le bash)
- Un script peut être exécuter en tant que commande : **premierscript.sh**
- Doit avoir le droit d'exécution **x** pour le propriétaire
- **chmod u+x premierscript.sh** (chmod pour la gestion des droits utilisateurs)