

# Chapitre 01 : INTRODUCTION GENERALE

## 1.1 Introduction

La définition du mot Linux dépend du contexte dans lequel il est utilisé. Techniquement parlant, Linux est le noyau du système, qui est le contrôleur central de tout ce qui se passe sur l'ordinateur. Les personnes qui disent que leur ordinateur « fonctionne sous Linux » font généralement référence au noyau et à la suite d'outils qui l'accompagnent (appelés distribution). Si quelqu'un dit avoir une expérience Linux, cela peut faire référence à la configuration de systèmes, à l'exécution de serveurs Web ou à un certain nombre d'autres services et programmes fonctionnant sous Linux. Au fil du temps, l'administration Linux a évolué pour englober à peu près toutes les tâches qu'une entreprise, un établissement d'enseignement ou un établissement gouvernemental moderne peut utiliser dans ses opérations quotidiennes.

**Et UNIX ?** UNIX était à l'origine un système d'exploitation développé par AT&T Bell Labs dans les années 1970. Il a été modifié et fork (c'est-à-dire que des personnes l'ont modifié et ces modifications ont servi de base à d'autres systèmes), de sorte qu'il existe désormais de nombreuses variantes différentes d'UNIX. Cependant, UNIX est désormais à la fois une marque commerciale et une spécification, propriété d'un consortium industriel appelé Open Group. Seuls les logiciels certifiés par l'Open Group peuvent s'appeler UNIX. Bien qu'il ait adopté la plupart, sinon la totalité, des exigences de la spécification UNIX, Linux n'a pas été certifié, donc Linux n'est vraiment pas UNIX ! C'est juste... de type UNIX.

### 1.1.1 Rôle du noyau

Les trois principaux composants d'un système d'exploitation sont le **noyau**, le **shell** et le **système de fichiers**. Le noyau du système d'exploitation est comme un contrôleur aérien dans un aéroport. Le noyau dicte quel programme obtient quelles parties de mémoire, il démarre et tue les programmes, il interprète les instructions qui lui sont données par l'utilisateur et il gère des tâches plus courantes et simples telles que l'affichage de texte sur un moniteur. Lorsqu'une application doit écrire sur le disque, elle doit demander au noyau de terminer l'opération d'écriture.

Le noyau gère également le changement d'application. Un ordinateur aura un ou plusieurs processeurs et une quantité limitée de mémoire. Le noyau se charge du déchargement des tâches et du chargement de nouvelles tâches, et peut gérer plusieurs tâches sur plusieurs processeurs. Lorsque la tâche en cours s'est exécutée pendant une durée suffisante, le processeur met la tâche en pause afin qu'une autre puisse s'exécuter. C'est ce qu'on appelle le multitâche préemptif. Le multitâche signifie que l'ordinateur effectue plusieurs tâches à la fois, et le mode préemptif signifie que le noyau décide quand basculer entre les tâches. Les tâches changeant si rapidement, il semble que l'ordinateur fasse plusieurs choses à la fois.

Chaque application peut penser qu'elle dispose d'un gros bloc de mémoire sur le système, mais c'est le noyau qui entretient cette illusion ; remapper des blocs de mémoire plus petits, partager des blocs de

mémoire avec d'autres applications ou même échanger des blocs qui n'ont pas été utilisés depuis un certain temps sur le disque.

Lorsque l'ordinateur démarre, il charge un petit morceau de code appelé chargeur de démarrage. Le travail du chargeur de démarrage est de vous donner un choix (si configuré) d'options pour charger une ou plusieurs versions de Linux, ou même d'autres systèmes d'exploitation, puis de charger le noyau de l'option choisie et de le démarrer. Si vous êtes plus familier avec les systèmes d'exploitation tels que Microsoft Windows ou OS X d'Apple, vous ne verrez probablement jamais le chargeur de démarrage, mais dans le monde UNIX, il est généralement visible afin que vous puissiez ajuster la façon dont votre ordinateur démarre.

Le chargeur de démarrage charge le noyau Linux puis transfère le contrôle. Linux continue ensuite à exécuter les programmes nécessaires pour rendre l'ordinateur utile, comme la connexion au réseau ou le démarrage d'un serveur Web.

### **1.1.2 Demandes**

Comme un contrôleur aérien, le noyau n'est pas utile sans quelque chose à contrôler. Si le noyau est la tour, les applications sont les avions. Les applications envoient des requêtes au noyau et reçoivent en retour des ressources, telles que la mémoire, le processeur et le disque. Le noyau extrait également les détails compliqués de l'application. L'application ne sait pas si le bloc d'un disque se trouve sur un disque SSD du fabricant A, un disque dur en métal en rotation du fabricant B ou même un partage de fichiers réseau. Les applications suivent simplement l'interface de programmation d'application (API) du noyau et n'ont pas à se soucier des détails d'implémentation.

Lorsque nous, utilisateurs, pensons aux applications, nous avons tendance à penser aux traitements de texte, aux navigateurs Web et aux clients de messagerie. Le noyau ne se soucie pas de savoir s'il exécute quelque chose destiné à l'utilisateur, un service réseau qui communique avec un ordinateur distant ou une tâche interne. Ainsi, à partir de là, nous obtenons une abstraction appelée processus. Un processus n'est qu'une tâche chargée et suivie par le noyau. Une application peut même avoir besoin de plusieurs processus pour fonctionner, de sorte que le noyau se charge d'exécuter les processus, de les démarrer et de les arrêter comme demandé, et de distribuer les ressources système.

### **1.1.3 Rôle de l'Open Source**

Les projets logiciels prennent la forme d'un code source, qui est un ensemble d'instructions informatiques lisibles par l'homme. Le code source peut être écrit dans des centaines de langages de programmation différents. Le noyau Linux est principalement écrit en C, un langage qui partage l'histoire avec l'UNIX d'origine.

Le code source n'est pas compris directement par l'ordinateur, il doit donc être compilé en instructions machine par un compilateur. Le compilateur rassemble tous les fichiers sources et génère quelque chose qui peut être exécuté sur l'ordinateur, comme le noyau Linux.

Historiquement, la plupart des logiciels ont été publiés sous une licence à source fermée, ce qui signifie que vous avez le droit d'utiliser le code machine, mais que vous ne pouvez pas voir le code source. Souvent, la licence indique spécifiquement que vous n'essayeriez pas de procéder à une ingénierie inverse du code machine jusqu'au code source pour comprendre ce qu'il fait !

L'Open Source adopte une vision centrée sur la source du logiciel. La philosophie open source est que vous avez le droit d'obtenir le logiciel et de le modifier pour votre propre usage. Linux a adopté cette philosophie avec beaucoup de succès.

En 1991, Linux a démarré comme un projet de loisir de Linus Torvalds. Il a rendu la source disponible gratuitement, permettant à d'autres de se joindre à lui et de façonner ce nouveau système d'exploitation. Ce n'était pas le premier système développé par un groupe de bénévoles, mais comme il a été construit à partir de zéro, les premiers utilisateurs ont pu influencer l'orientation du projet. Les gens ont pris la source, ont apporté des modifications et les ont partagées avec le reste du groupe, accélérant considérablement le rythme de développement et garantissant que les erreurs d'autres systèmes d'exploitation ne se reproduisaient pas.

Le noyau Linux est sous licence GNU Public License (GPL), ce qui vous oblige à rendre les modifications disponibles. Cela garantit que ceux qui utilisent le code contribueront également au bien commun en rendant ces changements accessibles à tous.

À côté de cela, il y avait le projet GNU (GNU, pas UNIX). Alors que GNU (prononcé « guh-noo ») construisait son propre système d'exploitation, ils réussissaient beaucoup mieux à créer les outils qui accompagnent un système d'exploitation UNIX, tels que les compilateurs et les interfaces utilisateur. Les sources étaient toutes disponibles gratuitement, Linux a donc pu cibler ses outils et fournir un système complet. En tant que tel, la plupart des outils qui font partie du système Linux proviennent de ces outils GNU.

Il existe de nombreuses variantes différentes sur l'open source. Cependant, tous conviennent que vous devriez avoir accès au code source, mais ils diffèrent sur la manière dont vous pouvez, ou dans certains cas, devez, redistribuer les modifications.

#### **1.1.4 Distribution Linux**

Prenez le noyau Linux et les outils GNU, ajoutez d'autres applications destinées aux utilisateurs comme un client de messagerie, des traitements de texte et d'autres programmes et vous obtenez un système Linux complet. Les gens ont commencé à regrouper tous ces logiciels dans une distribution presque dès que Linux est devenu utilisable. La distribution s'occupe de la configuration du stockage, de l'installation du noyau et de l'installation du reste du logiciel. Les distributions complètes incluent également des outils pour gérer le système et un gestionnaire de packages pour vous aider à ajouter et supprimer des logiciels une fois l'installation terminée.

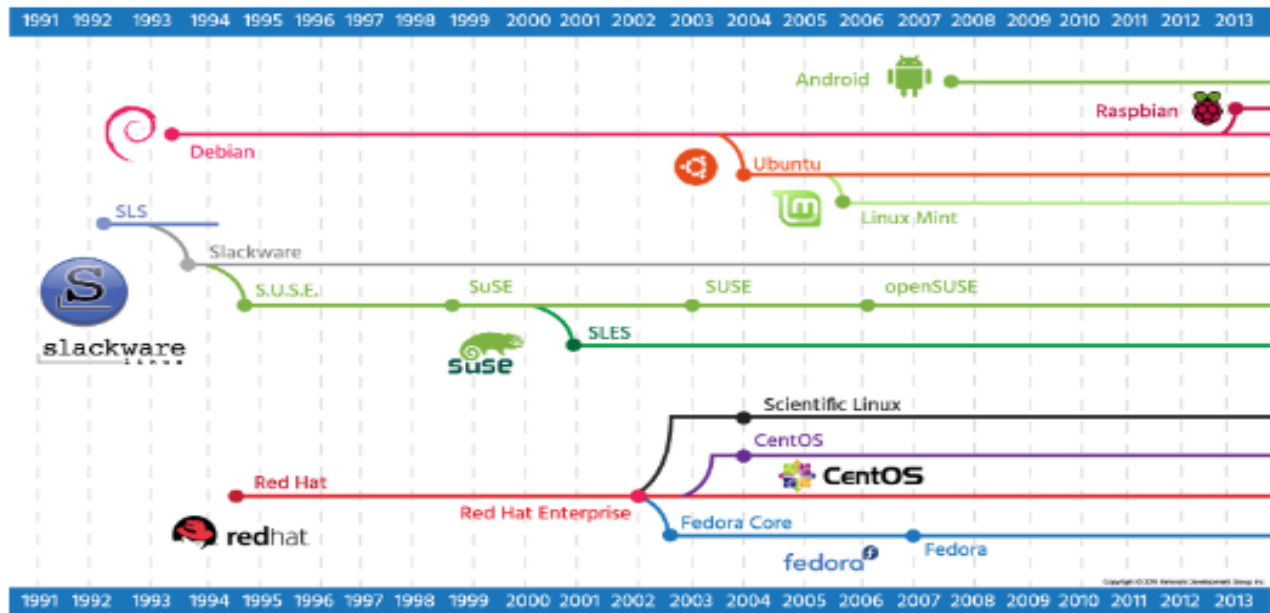
Comme UNIX, il existe de nombreuses versions différentes de distributions. De nos jours, il existe des distributions qui se concentrent sur l'exécution de serveurs, d'ordinateurs de bureau ou même d'outils spécifiques à un secteur comme la conception électronique ou le calcul statistique. Les principaux acteurs du marché remontent soit à **Red Hat**, soit à **Debian**. La différence la plus visible est le gestionnaire de progiciels, bien que vous puissiez trouver d'autres différences sur tout, depuis l'emplacement des fichiers jusqu'aux philosophies politiques.

**Red Hat** a commencé comme une simple distribution qui a introduit le Red Hat Package Manager (RPM) basé sur le format de fichier **.rpm**. Le développeur a finalement formé une entreprise autour de lui, qui a tenté de commercialiser un ordinateur de bureau Linux pour les entreprises. Au fil du temps, Red Hat a commencé à se concentrer davantage sur les applications serveur telles que le service Web et de fichiers et a publié Red Hat Enterprise Linux, qui était un service payant avec un long cycle de publication. Le cycle de publication dicte la fréquence à laquelle le logiciel est mis à niveau. Une entreprise peut apprécier la stabilité et souhaiter des cycles de publication longs, tandis qu'un amateur ou une startup peut souhaiter le logiciel le plus récent et opter pour un cycle de publication plus court. Pour satisfaire ce dernier groupe, Red Hat sponsorise le projet Fedora qui réalise un bureau personnel comprenant les derniers logiciels mais toujours construit sur les mêmes bases que la version entreprise.

Parce que tout dans Red Hat Enterprise Linux est open source, un projet appelé CentOS a vu le jour, qui a recompilé tous les packages RHEL et les a distribués gratuitement. CentOS et d'autres similaires (tels que Scientific Linux) sont largement compatibles avec RHEL et intègrent des logiciels plus récents, mais n'offrent pas le support payant de Red Hat.

Debian est davantage un effort communautaire et, en tant que tel, promeut également l'utilisation de logiciels open source et le respect des normes. Debian a proposé son propre système de gestion de paquets basé sur le format de fichier **.deb**. Alors que Red Hat laisse la prise en charge des plates-formes non Intel et AMD aux projets dérivés, Debian prend directement en charge bon nombre de ces plates-formes.

**Ubuntu** est la distribution dérivée de Debian la plus populaire. Il s'agit de la création de Canonical, une entreprise qui a été créée pour favoriser la croissance d'Ubuntu et qui gagne de l'argent en fournissant un support.



## 1.2 Plateformes matérielles

Linux a commencé comme quelque chose qui ne pouvait fonctionner que sur un ordinateur comme celui de Linus : un 386 avec un contrôleur de disque dur spécifique. La gamme de support s'est élargie à mesure que la prise en charge d'autres matériels a été créée. Finalement, Linux a commencé à prendre en charge d'autres chipsets, y compris du matériel conçu pour exécuter des systèmes d'exploitation concurrents !

Les types de matériel sont passés de la modeste puce Intel aux superordinateurs. Des puces Linux de plus petite taille ont finalement été développées pour s'adapter aux appareils grand public (appelés appareils intégrés). La prise en charge de Linux est devenue omniprésente, de sorte qu'il est souvent plus facile de créer du matériel pour prendre en charge Linux, puis d'utiliser Linux comme tremplin pour votre logiciel personnalisé, plutôt que de créer le matériel et les logiciels personnalisés à partir de zéro.

Finalement, les téléphones portables et les tablettes ont adopté Linux. Une société, rachetée plus tard par Google, a mis au point la plate-forme Android, qui regroupe Linux et les logiciels nécessaires au fonctionnement d'un téléphone ou d'une tablette. Cela signifie que les efforts nécessaires pour commercialiser un téléphone sont nettement moindres. Au lieu d'un long développement sur un nouveau système d'exploitation, les entreprises peuvent consacrer leur temps à innover sur les logiciels destinés aux utilisateurs. Android est désormais l'un des leaders du marché dans le domaine des téléphones et des tablettes.

Outre les téléphones et les tablettes, Linux est présent sur de nombreux appareils grand public. Les routeurs sans fil fonctionnent souvent sous Linux car ils disposent d'un riche ensemble de fonctionnalités réseau. Le **TiVo** est un enregistreur vidéo numérique grand public construit sous Linux. Même si ces appareils reposent sur Linux, les utilisateurs finaux n'ont pas besoin de savoir comment utiliser Linux. Le logiciel personnalisé interagit avec l'utilisateur et Linux fournit la plate-forme stable.

## 1.3 Coque

Un système d'exploitation fournit au moins un shell qui permet à l'utilisateur de communiquer avec le système d'exploitation. Un shell est parfois appelé interpréteur car il prend les commandes émises par un utilisateur et les interprète sous une forme que le noyau peut ensuite exécuter sur le matériel de l'ordinateur. Les deux types de shells les plus courants sont l'interface utilisateur graphique (GUI) et l'interface de ligne de commande (CLI).

**Microsoft Windows™** utilise généralement un shell GUI, utilisant principalement la souris pour indiquer ce que vous souhaitez accomplir. Bien que l'utilisation d'un système d'exploitation de cette manière puisse être considérée comme facile, l'utilisation d'une CLI présente de nombreux avantages, notamment :

Répétition de commandes : dans un shell GUI, il n'existe pas de moyen simple de répéter une commande précédente. Dans une CLI, il existe un moyen simple de répéter (et également de modifier) une commande précédente.

Flexibilité des commandes : le shell de l'interface graphique offre une flexibilité limitée dans la manière dont la commande s'exécute. Dans une CLI, les options sont spécifiées avec des commandes pour fournir une interface plus flexible et plus puissante.

**Ressources** : un shell GUI utilise généralement une quantité relativement importante de ressources (RAM, CPU, etc.). En effet, l'affichage des graphiques nécessite beaucoup de puissance de traitement et de mémoire. En revanche, une CLI utilise très peu de ressources système, ce qui permet à davantage de ces ressources d'être disponibles pour d'autres programmes.

**Scripts** : dans un shell d'interface graphique, l'exécution de plusieurs tâches nécessite souvent plusieurs clics de souris. Avec une CLI, un script peut être créé pour exécuter de nombreuses opérations complexes en tapant simplement le nom du script. Un script est une série de commandes placées dans un seul fichier. Une fois exécuté, le script exécute toutes les commandes du fichier.

**Accès à distance** : bien qu'il soit possible d'exécuter des commandes dans un shell GUI à distance, cette fonctionnalité n'est généralement pas configurée par défaut. Avec un shell CLI, accéder à une machine distante est facile et généralement disponible par défaut.

**Développement** : Normalement, la création d'un programme basé sur une interface graphique prend plus de temps aux développeurs que celle des programmes basés sur la CLI. En conséquence, il existe généralement des milliers de programmes CLI sur un système d'exploitation Linux classique, contre seulement quelques centaines de programmes dans un système d'exploitation principalement basé sur une interface graphique comme Microsoft Windows. Plus de programmes signifie plus de puissance et de flexibilité.

Le système d'exploitation Microsoft Windows a été conçu pour utiliser principalement l'interface GUI en raison de sa simplicité, bien que plusieurs interfaces CLI soient également disponibles. Pour les commandes simples, il existe la boîte de dialogue Exécuter, dans laquelle vous pouvez saisir ou parcourir les commandes que vous souhaitez exécuter. Si vous souhaitez taper plusieurs commandes ou si vous souhaitez voir le résultat de la commande, vous pouvez utiliser l'invite de commande, également appelée shell DOS. Récemment, Microsoft a réalisé à quel point il est important de disposer d'un environnement de ligne de commande puissant et a introduit Powershell.

Comme Windows, Linux dispose également d'une CLI et d'une interface graphique. Contrairement à Windows, Linux vous permet de modifier facilement le shell GUI (également appelé environnement de bureau) que vous souhaitez utiliser. Les deux environnements de bureau les plus courants pour Linux sont GNOME et KDE ; cependant, il existe de nombreux autres shells GUI disponibles.

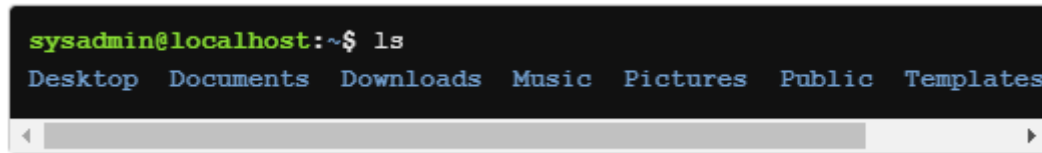
Pour accéder à la CLI depuis l'interface graphique sur un système d'exploitation Linux, l'utilisateur peut ouvrir un logiciel appelé terminal. Linux peut également être configuré uniquement pour exécuter la CLI sans l'interface graphique ; cela se fait généralement sur des serveurs qui ne nécessitent pas d'interface graphique, principalement pour libérer des ressources système.

### 1.3.1 Shell Bash

Non seulement le système d'exploitation Linux fournit plusieurs shells GUI, mais plusieurs shells CLI sont également disponibles. Normalement, ces shells sont dérivés de l'un des deux anciens shells UNIX : le Bourne Shell et le C Shell. En fait, le shell Bash, un shell CLI par défaut utilisé dans les systèmes d'exploitation Linux modernes, tire son nom du Bourne Shell : Bourne Again SHell. Dans ce cours, vous vous concentrerez sur l'apprentissage de l'utilisation de la CLI pour Linux avec le shell Bash, sans doute la CLI la plus populaire sous Linux.

Les utilisateurs interagissent avec un système en exécutant des commandes qui sont interprétées par le shell et transformées en actions par le noyau. Ces actions peuvent ou non renvoyer des informations à la

ligne de commande en fonction de la commande émise et de son résultat. Par exemple, lorsque la commande `ls` est saisie dans la console, elle renvoie le contenu du répertoire dans lequel se trouve actuellement l'utilisateur.



```
sysadmin@localhost:~$ ls
Desktop Documents Downloads Music Pictures Public Templates
```

Une commande peut être suivie d'options qui modifient la façon dont la commande est exécutée, et d'arguments, qui sont généralement les fichiers sur lesquels opérer :

```
command [options] [arguments]
```

Les commandes saisies sont considérées comme une entrée standard (stdin), qu'elles soient saisies par un opérateur, saisies par un script ou le résultat d'une autre commande. Le texte renvoyé à la console peut être soit une sortie standard (stdout), soit une erreur standard (stderr).

Cette méthode de communication d'une simplicité trompeuse avec le noyau Linux constitue la base de presque toutes les interactions qu'un administrateur Linux a avec ses systèmes. Cela peut être déroutant au début pour les utilisateurs qui n'ont qu'une expérience des interfaces GUI, mais en fin de compte, cela donne à l'opérateur expérimenté bien plus de puissance que n'importe quelle interface graphique.

Le shell Bash possède de nombreuses commandes et fonctionnalités intégrées que vous apprendrez, notamment :

Alias : donnez à une commande un nom différent ou plus court pour rendre le travail avec le shell plus efficace.

Réexécution des commandes : pour éviter de retaper de longues lignes de commande.

Correspondance de caractères génériques : utilise des caractères spéciaux tels que `?`, `*` et `[]` pour sélectionner un ou plusieurs fichiers en tant que groupe à traiter.

Redirection d'entrée/sortie : utilise des caractères spéciaux pour rediriger l'entrée, `<` ou `<<`, et la sortie, `>`.



Pipes : Utilisé pour connecter une ou plusieurs commandes simples pour effectuer des opérations plus complexes.

Traitement en arrière-plan : permet aux programmes et aux commandes de s'exécuter en arrière-plan pendant que l'utilisateur continue d'interagir avec le shell pour effectuer d'autres tâches.

Le shell que votre compte utilisateur utilise par défaut est défini au moment de la création de votre compte utilisateur. Par défaut, de nombreuses distributions Linux utilisent Bash pour le shell d'un nouvel utilisateur. En règle générale, un utilisateur apprend un shell et s'en tient à ce shell ; cependant, après avoir appris les bases de Linux, vous souhaitez peut-être explorer les fonctionnalités d'autres shells.

### 1.3.2 Accéder au shell

La manière dont vous accédez au shell de ligne de commande dépend si votre système fournit une connexion GUI ou CLI :

**Systèmes basés sur une interface graphique** : si le système est configuré pour présenter une interface graphique, vous devrez alors trouver une application logicielle appelée terminal. Dans l'environnement de bureau GNOME, l'application de terminal peut être démarrée en cliquant sur le menu Applications, puis sur le menu Outils système et sur l'icône Terminal.

**Systèmes basés sur CLI** : de nombreux systèmes Linux, en particulier les serveurs, ne sont pas configurés pour fournir une interface graphique par défaut, ils présentent donc une interface CLI à la place. Si le système est configuré pour présenter une CLI, il exécute automatiquement une application de terminal après votre connexion.

Aux débuts de l'informatique, les terminaux étaient de grandes machines qui permettaient aux utilisateurs de saisir des données via un clavier et d'afficher les résultats en les imprimant sur papier. Au fil du temps, les terminaux ont évolué et leur taille a été réduite à quelque chose qui ressemblait à un ordinateur de bureau avec un moniteur d'affichage vidéo pour la sortie et un clavier pour la saisie.

Finalement, avec l'introduction des ordinateurs personnels, les terminaux sont devenus des émulateurs logiciels du matériel réel. Tout ce que vous tapez dans le terminal est interprété par votre shell et traduit sous une forme qui peut ensuite être exécutée par le noyau du système d'exploitation.

Si vous vous trouvez dans un endroit distant, des connexions de pseudo-terminaux peuvent également être établies sur le réseau à l'aide de plusieurs techniques. Des connexions non sécurisées peuvent être établies à l'aide de protocoles tels que telnet et de programmes tels que rlogin, tandis que des

connexions sécurisées peuvent être établies à l'aide de programmes tels que putty et de protocoles tels que ssh.

## 1.4 Systèmes de fichiers

Outre le noyau et le shell, l'autre composant majeur de tout système d'exploitation est le système de fichiers. Pour l'utilisateur, un système de fichiers est une hiérarchie de répertoires et de fichiers avec la racine / répertoire en haut de l'arborescence des répertoires. Pour le système d'exploitation, un système de fichiers est une structure créée sur une partition de disque constituée de tables définissant les emplacements des répertoires et des fichiers.

