Linux Professional Institute

LPIC-1

Version 5.0
Français

101

Table of Contents

Linux Professional Institute

Thème 101 : Architecture système

101.1 Détermination et configuration des paramètres du matériel

Référence aux objectifs de LPI

LPIC-1 v5, Exam 101, Objective 101.1

Valeur

2

Domaines de connaissance les plus importants

- Activer et désactiver les périphériques intégrés.
- Savoir différencier les types de périphériques de stockage de masse.
- Déterminer les ressources matérielles des périphériques.
- Outils et commandes permettant d'obtenir des informations sur les périphériques (par exemple lsusb, lspci, etc.).
- Outils et commandes permettant de manipuler les périphériques USB.
- Compréhension des concepts sysfs, udev et dbus.

Liste partielle de termes, fichiers et utilitaires utilisés pour cet objectif

- `/sys/`
- `/proc/`
- `/dev/`
- `modprobe`
- `lsmod`
- `lspci`
- `lsusb`

Linux Professional Institute

101.1 Leçon 1

Certification :	LPIC-1
Version :	5.0
Thème :	101 Architecture système
Objectif :	101.1 Détermination et configuration des paramètres du matériel
Leçon :	1 sur 1

Introduction

Depuis les débuts de l'informatique, les fabricants d'ordinateurs professionnels et personnels ont intégré dans leurs machines une variété de composants matériels qui devaient être pris en charge à leur tour par le système d'exploitation. Tâche difficile voire impossible du point de vue des développeurs de systèmes d'exploitation, à moins que l'industrie ne se mette d'accord sur les normes relatives aux jeux d'instructions et à la communication avec les périphériques. De la même manière que la couche d'abstraction standardisée fournie par le système d'exploitation à une application, ces normes facilitent le développement et la maintenance d'un système d'exploitation qui n'est pas lié à un type de matériel spécifique. Ceci étant dit, la complexité du matériel intégré sous le capot nécessite parfois certains ajustements quant à la manière dont les ressources matérielles sont exposées au système afin que celui-ci puisse être installé et fonctionner correctement.

Certains de ces réglages peuvent même être effectués en l'absence de système d'exploitation. La plupart des ordinateurs offrent un outil de configuration qui peut être exécuté à l'allumage de la machine. Jusque vers le début des années 2000, cet utilitaire de configuration était implémenté dans le BIOS (*Basic Input/Output System*, système élémentaire d'entrée/sortie), la norme pour le

firmware (micrologiciel) contenant les routines de configuration de base que l'on peut trouver sur les cartes mère x86. À partir de la fin de la première décennie des années 2000, les machines basées sur l'architecture x86 ont commencé à remplacer le BIOS par une nouvelle implémentation appelée UEFI (*Unified Extensible Firmware Interface*, interface micrologicielle extensible unifiée) dotée de fonctionnalités plus sophistiquées pour l'identification, les tests et la configuration matérielle ainsi que les mises à jour du *firmware*. Malgré cette évolution, il n'est pas rare de nos jours d'avoir recours à l'utilitaire de configuration BIOS, étant donné que les deux implémentations répondent au même besoin.

NOTE Les menus détails sur les points communs et les différences entre le BIOS et l'UEFI seront traités dans une leçon ultérieure.

Activation des périphériques

L'utilitaire de configuration du système s'affiche lorsqu'on appuie sur une touche spécifique lors de l'allumage de l'ordinateur. Cette touche peut varier d'un fabricant à l'autre, mais il s'agit généralement de la touche `Suppr` ou de l'une des touches de fonction comme `F2` ou `F12`. La combinaison de touches à utiliser est souvent affichée à l'écran juste après l'allumage.

Le BIOS permet de prendre en charge ou de désactiver les périphériques intégrés, d'activer la protection contre les erreurs et de modifier les paramètres matériels tels que les interruptions matérielles (IRQ) et l'accès direct à la mémoire (DMA). La modification de ces paramètres est rarement nécessaire sur les machines modernes, mais leur ajustement peut s'avérer nécessaire pour résoudre des problèmes spécifiques. Certaines technologies RAM, par exemple, sont compatibles avec des taux de transfert de données plus élevés que celui par défaut, il est donc recommandé de le remplacer par le taux de transfert spécifié par le fabricant. Certains processeurs offrent des fonctionnalités pas forcément requises pour une installation particulière, et qui peuvent être désactivées. La désactivation de ces fonctionnalités permet de réduire la consommation en énergie tout en augmentant la protection du système, étant donné que certaines fonctionnalités du CPU qui contiennent des bugs peuvent également être désactivées.

Lorsque la machine est équipée d'un certain nombre de périphériques de stockage, il est important de définir celui qui contient le bon chargeur de démarrage et qui doit apparaître en premier dans l'ordre d'amorçage des périphériques. Le système d'exploitation peut ne pas se lancer lorsqu'un périphérique incorrect apparaît en tête de liste dans les vérifications de démarrage du BIOS.

Inspection du matériel sous Linux

Une fois que les périphériques ont été correctement identifiés, il appartient au système de leur

associer les composants logiciels requis pour leur bon fonctionnement. Lorsqu'un matériel ne fonctionne pas comme prévu, il est important de savoir à quel niveau exactement se situe le problème. Lorsqu'un composant matériel n'est pas détecté par le système d'exploitation, il y a de fortes chances pour que la pièce - ou le port auquel elle est connectée - soit défectueuse. Lorsque la partie matérielle est correctement détectée mais qu'elle ne fonctionne pas comme elle devrait, le problème se situe probablement du côté du système d'exploitation. Par conséquent, l'une des premières étapes lorsqu'on traite les problèmes liés au matériel consiste à vérifier si le système d'exploitation détecte correctement le périphérique. Il existe deux méthodes de base pour identifier les ressources matérielles sur un système Linux : utiliser des commandes dédiées à cet effet ou lire des fichiers spécifiques dans des systèmes de fichiers spéciaux.

Commandes pour l'inspection

Les deux commandes essentielles pour identifier les périphériques connectés dans un système Linux sont :

`lspci`

Affiche tous les périphériques actuellement connectés au bus PCI (*Peripheral Component Interconnect*). Les périphériques PCI peuvent être intégrés à la carte mère, comme un contrôleur de disque, ou alors ils peuvent être connectés sur les ports d'extension de la carte mère, comme une carte graphique externe.

`lsusb`

Répertorie les périphériques USB (*Universal Serial Bus*) actuellement connectés à la machine. Même s'il existe des périphériques USB pour presque tous les usages imaginables, l'interface USB est utilisée principalement pour connecter des périphériques d'entrée - claviers, dispositifs de pointage - et des supports de stockage amovibles.

La sortie des commandes `lspci` et `lsusb` consiste en une liste de tous les périphériques PCI et USB identifiés par le système d'exploitation. Cependant, il se peut que le périphérique ne soit pas encore pleinement opérationnel, étant donné que chaque pièce matérielle requiert un composant logiciel pour contrôler le périphérique correspondant. Ce composant logiciel est appelé un *module du noyau* et il peut faire partie du noyau Linux officiel ou être ajouté depuis une source tierce.

Les modules du noyau Linux associés aux périphériques matériels sont également appelés pilotes ou *drivers*, comme dans d'autres systèmes d'exploitation. En revanche, les pilotes pour Linux ne sont pas toujours fournis par les fabricants de ces périphériques. Même si certains fabricants fournissent leurs propres pilotes binaires à installer séparément, de nombreux pilotes sont écrits par des développeurs indépendants. Historiquement, les composants qui fonctionnent sous Windows, par exemple, peuvent ne pas avoir de module de noyau correspondant pour Linux. De nos jours, les systèmes d'exploitation basés sur Linux offrent un support matériel important et la

plupart des appareils fonctionnent sans problème.

Les commandes directement liées au matériel requièrent souvent les droits root pour être exécutées ou affichent seulement des informations limitées lorsqu'elles sont invoquées par un utilisateur normal, il peut donc être nécessaire de se connecter en tant que root ou d'exécuter la commande avec `sudo`.

La sortie suivante de la commande `lspci`, par exemple, affiche quelques périphériques identifiés :

```
$ lspci
01:00.0 VGA compatible controller: NVIDIA Corporation GM107 [GeForce GTX 750 Ti] (rev a2)
04:02.0 Network controller: Ralink corp. RT2561/RT61 802.11g PCI
04:04.0 Multimedia audio controller: VIA Technologies Inc. ICE1712 [Envy24] PCI Multi-
Channel I/O Controller (rev 02)
04:0b.0 FireWire (IEEE 1394): LSI Corporation FW322/323 [TrueFire] 1394a Controller (rev 70)
```

La sortie de ces commandes peut compter des dizaines de lignes, les exemples précédents et suivants ne contiennent que les sections qui nous intéressent. Les nombres hexadécimaux au début de chaque ligne représentent l'adresse unique du périphérique PCI correspondant. La commande `lspci` affiche davantage de détails sur un périphérique donné lorsque son adresse est précisée par l'option `-s`, suivie de l'option `-v` :

```
$ lspci -s 04:02.0 -v
04:02.0 Network controller: Ralink corp. RT2561/RT61 802.11g PCI
    Subsystem: Linksys WMP54G v4.1
    Flags: bus master, slow devsel, latency 32, IRQ 21
    Memory at e3100000 (32-bit, non-prefetchable) [size=32K]
    Capabilities: [40] Power Management version 2
    kernel driver in use: rt61pci
```

La sortie affiche maintenant beaucoup plus de détails sur le périphérique à l'adresse `04:02.0`. Il s'agit d'une carte réseau dont le nom interne est `Ralink corp. RT2561/RT61 802.11g PCI`. L'entrée `Subsystem` est associée à la marque et au modèle de l'appareil — `Linksys WMP54G v4.1` — et peut servir à des fins de diagnostic.

Le module du noyau peut être identifié dans la ligne `kernel driver in use`, qui affiche le module `rt61pci`. D'après toutes les informations recueillies, il est correct de supposer que :

1. Le périphérique a été identifié.
2. Un module de noyau correspondant a été chargé.

3. Le périphérique devrait être prêt à l'emploi.

Une autre façon de vérifier quel module du noyau est utilisé pour un périphérique donné est fournie par l'option `-k`, disponible dans les versions plus récentes de `lspci` :

```
$ lspci -s 01:00.0 -k
01:00.0 VGA compatible controller: NVIDIA Corporation GM107 [GeForce GTX 750 Ti] (rev a2)
    kernel driver in use: nvidia
    kernel modules: nouveau, nvidia_drm, nvidia
```

Pour le périphérique en question, une carte graphique NVIDIA, `lspci` nous indique à la ligne `kernel driver in use: nvidia` que le module en question est nommé `nvidia`, et tous les modules correspondants du noyau sont listés à la ligne `kernel modules : nouveau, nvidia_drm, nvidia`.

La commande `lsusb` est similaire à `lspci`, mais liste exclusivement les informations USB :

```
$ lsusb
Bus 001 Device 029: ID 1781:0c9f Multiple Vendors USBtiny
Bus 001 Device 028: ID 093a:2521 Pixart Imaging, Inc. Optical Mouse
Bus 001 Device 020: ID 1131:1001 Integrated System Solution Corp. KY-BT100 Bluetooth Adapter
Bus 001 Device 011: ID 04f2:0402 Chicony Electronics Co., Ltd Genius LuxeMate i200 Keyboard
Bus 001 Device 007: ID 0424:7800 Standard Microsystems Corp.
Bus 001 Device 003: ID 0424:2514 Standard Microsystems Corp. USB 2.0 Hub
Bus 001 Device 002: ID 0424:2514 Standard Microsystems Corp. USB 2.0 Hub
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
```

La commande `lsusb` affiche les ports USB disponibles et les périphériques qui y sont connectés. Tout comme pour `lspci`, l'option `-v` affiche des informations plus détaillées. Un périphérique spécifique peut être sélectionné pour l'inspection en fournissant son ID à l'option `d` :

```
$ lsusb -v -d 1781:0c9f
Bus 001 Device 029: ID 1781:0c9f Multiple Vendors USBtiny
Device Descriptor:
  bLength                18
  bDescriptorType         1
  bcdUSB               1.01
  bDeviceClass          255 Vendor Specific Class
  bDeviceSubClass         0
  bDeviceProtocol         0
  bMaxPacketSize0         8
```

```
  idVendor           0x1781 Multiple Vendors
  idProduct          0x0c9f USBtiny
  bcdDevice            1.04
  iManufacturer           0
  iProduct                2 USBtiny
  iSerial                 0
  bNumConfigurations      1
```

Avec l'option `-t`, la commande `lsusb` affiche le mappage en vigueur des périphériques USB sous forme d'une arborescence hiérarchique :

```
$ lsusb -t
/:  Bus 01.Port 1: Dev 1, Class=root_hub, Driver=dwc_otg/1p, 480M
    |__ Port 1: Dev 2, If 0, Class=Hub, Driver=hub/4p, 480M
        |__ Port 1: Dev 3, If 0, Class=Hub, Driver=hub/3p, 480M
            |__ Port 2: Dev 11, If 1, Class=Human Interface Device, Driver=usbhid, 1.5M
            |__ Port 2: Dev 11, If 0, Class=Human Interface Device, Driver=usbhid, 1.5M
            |__ Port 3: Dev 20, If 0, Class=Wireless, Driver=btusb, 12M
            |__ Port 3: Dev 20, If 1, Class=Wireless, Driver=btusb, 12M
            |__ Port 3: Dev 20, If 2, Class=Application Specific Interface, Driver=, 12M
            |__ Port 1: Dev 7, If 0, Class=Vendor Specific Class, Driver=lan78xx, 480M
        |__ Port 2: Dev 28, If 0, Class=Human Interface Device, Driver=usbhid, 1.5M
        |__ Port 3: Dev 29, If 0, Class=Vendor Specific Class, Driver=, 1.5M
```

Il est possible que tous les périphériques ne soient pas associés à un module correspondant. La communication avec certains périphériques peut être gérée directement par l'application, sans passer par l'intermédiaire d'un module. Néanmoins, il y a des informations importantes dans les résultats fournis par `lsusb -t`. Lorsqu'un module approprié existe, son nom apparaît à la fin de la ligne correspondant au périphérique, comme dans `Driver=btusb`. La `Class` du périphérique identifie la catégorie générale, comme `Human Interface Device`, `Wireless` ou `Mass Storage`, entre autres. Pour vérifier quel périphérique utilise le module `btusb` qui figure dans le listing ci-dessus, les numéros de `Bus` et de `Dev` doivent être fournis en argument à l'option `-s` de la commande `lsusb` :

```
$ lsusb -s 01:20
Bus 001 Device 020: ID 1131:1001 Integrated System Solution Corp. KY-BT100 Bluetooth Adapter
```

Il est courant de se retrouver avec un grand nombre de modules de noyau chargés à tout moment dans un système Linux standard. La meilleure façon d'interagir avec eux consiste à utiliser les commandes fournies par le paquet `kmod`, qui est un ensemble d'outils pour gérer les tâches

communes avec les modules du noyau Linux comme l'insertion, la suppression, la liste, la vérification des propriétés, la résolution des dépendances et des alias. La commande `lsmod`, par exemple, affiche tous les modules actuellement chargés :

```
$ lsmod
Module                  Size  Used by
kvm_intel             138528  0
kvm                   421021  1 kvm_intel
iTCO_wdt               13480  0
iTCO_vendor_support    13419  1 iTCO_wdt
snd_usb_audio         149112  2
snd_hda_codec_realtek  51465  1
snd_ice1712            75006  3
snd_hda_intel          44075  7
arc4                   12608  2
snd_cs8427             13978  1 snd_ice1712
snd_i2c                13828  2 snd_ice1712,snd_cs8427
snd_ice17xx_ak4xxx     13128  1 snd_ice1712
snd_ak4xxx_adda        18487  2 snd_ice1712,snd_ice17xx_ak4xxx
microcode              23527  0
snd_usbmidi_lib        24845  1 snd_usb_audio
gspca_pac7302          17481  0
gspca_main             36226  1 gspca_pac7302
videodev              132348  2 gspca_main,gspca_pac7302
rt61pci                32326  0
rt2x00pci              13083  1 rt61pci
media                  20840  1 videodev
rt2x00mmio             13322  1 rt61pci
hid_dr                 12776  0
snd_mpu401_uart        13992  1 snd_ice1712
rt2x00lib              67108  3 rt61pci,rt2x00pci,rt2x00mmio
snd_rawmidi            29394  2 snd_usbmidi_lib,snd_mpu401_uart
```

La sortie de la commande `lsmod` est divisée en trois colonnes :

`Module`

Le nom du module.

`Size`

La quantité de RAM occupée par le module, en octets.

`Used by`

Les modules dépendants.

Certains modules requièrent d'autres modules pour fonctionner correctement, comme c'est le cas pour les modules des périphériques audio :

```
$ lsmod | fgrep -i snd_hda_intel
snd_hda_intel          42658  5
snd_hda_codec         155748  3 snd_hda_codec_hdmi,snd_hda_codec_via,snd_hda_intel
snd_pcm                81999  3 snd_hda_codec_hdmi,snd_hda_codec,snd_hda_intel
snd_page_alloc         13852  2 snd_pcm,snd_hda_intel
snd                    59132  19
snd_hwdep,snd_timer,snd_hda_codec_hdmi,snd_hda_codec_via,snd_pcm,snd_seq,snd_hda_codec,snd_h
da_intel,snd_seq_device
```

La troisième colonne, `Used by`, indique les modules qui ont besoin du module figurant dans la première colonne pour fonctionner correctement. Beaucoup de modules de l'architecture audio de Linux, préfixés par `snd`, sont interdépendants. Lors de la recherche de problèmes pendant le diagnostic du système, il peut être utile de décharger certains modules spécifiques actuellement chargés. La commande `modprobe` peut être utilisée pour charger et décharger les modules du noyau : pour décharger un module et ses modules connexes tant qu'ils ne sont pas utilisés par un processus en cours, la commande `modprobe -r` doit être utilisée. Par exemple, pour décharger le module `snd-hda-intel` (le module pour un périphérique audio Intel HDA) et d'autres modules liés au système audio :

```
# modprobe -r snd-hda-intel
```

Outre le chargement et le déchargement des modules du noyau pendant le fonctionnement du système, il est possible de modifier les paramètres des modules lors du chargement du noyau, ce qui est comparable à la transmission d'options aux commandes. Chaque module accepte des paramètres spécifiques, mais la plupart du temps les valeurs par défaut sont recommandées et les paramètres supplémentaires ne sont pas nécessaires. Dans certains cas, il est toutefois nécessaire d'utiliser ces paramètres pour modifier le comportement d'un module de manière à ce qu'il fonctionne comme prévu.

En utilisant le nom du module comme seul argument, la commande `modinfo` affiche une description, le fichier, l'auteur, la licence, l'identification, les dépendances et les paramètres disponibles pour le module donné. Les paramètres personnalisés d'un module peuvent être rendus persistants en les incluant dans le fichier `/etc/modprobe.conf` ou dans des fichiers individuels avec l'extension `.conf` dans le répertoire `/etc/modprobe.d/`. L'option `-p` fera en

sorte que la commande `modinfo` affiche tous les paramètres disponibles en ignorant les autres informations :

```
# modinfo -p nouveau
vram_pushbuf:Create DMA push buffers in VRAM (int)
tv_norm:Default TV norm.
                Supported: PAL, PAL-M, PAL-N, PAL-Nc, NTSC-M, NTSC-J,
                        hd480i, hd480p, hd576i, hd576p, hd720p, hd1080i.
                Default: PAL
                NOTE Ignored for cards with external TV encoders. (charp)
nofbaccel:Disable fbcon acceleration (int)
fbcon_bpp:fbcon bits-per-pixel (default: auto) (int)
mst:Enable DisplayPort multi-stream (default: enabled) (int)
tv_disable:Disable TV-out detection (int)
ignorelid:Ignore ACPI lid status (int)
duallink:Allow dual-link TMDS (default: enabled) (int)
hdmimhz:Force a maximum HDMI pixel clock (in MHz) (int)
config:option string to pass to driver core (charp)
debug:debug string to pass to driver core (charp)
noaccel:disable kernel/abi16 acceleration (int)
modeset:enable driver (default: auto, 0 = disabled, 1 = enabled, 2 = headless) (int)
atomic:Expose atomic ioctl (default: disabled) (int)
runpm:disable (0), force enable (1), optimus only default (-1) (int)
```

L'exemple ci-dessus affiche tous les paramètres disponibles pour le module `nouveau`, un module de noyau fourni par le projet Nouveau comme alternative aux pilotes propriétaires pour les cartes graphiques NVIDIA. L'option `modeset`, par exemple, permet de contrôler si la résolution et la profondeur d'affichage sont définies dans l'espace noyau plutôt que dans l'espace utilisateur. L'ajout de `options nouveau modeset=0` au fichier `/etc/modprobe.d/nouveau.conf` désactivera la fonctionnalité `modeset` du noyau.

Lorsqu'un module pose problème, le fichier `/etc/modprobe.d/blacklist.conf` peut être utilisé pour bloquer le chargement du module. Par exemple, pour empêcher le chargement automatique du module `nouveau`, la ligne `blacklist nouveau` doit être ajoutée au fichier `/etc/modprobe.d/blacklist.conf`. Cette action est requise lorsque le module propriétaire `nvidia` est installé et que le module par défaut `nouveau` doit être ignoré.

NOTE

Rien ne vous empêche de modifier le fichier `/etc/modprobe.d/blacklist.conf` qui est déjà présent sur le système par défaut. Cependant, la méthode privilégiée consiste à créer un fichier de configuration séparé, `/etc/modprobe.d/<nom_du_module>.conf`, qui contiendra les paramètres spécifiques au module du noyau en question.

Fichiers d'informations et fichiers de périphériques

Les commandes `lspci`, `lsusb` et `lsmod` agissent comme des frontaux pour lire les informations relatives au matériel stockées par le système d'exploitation. Ce type d'information est conservé dans des fichiers spéciaux dans les répertoires `/proc` et `/sys`. Ces répertoires sont des points de montage vers des systèmes de fichiers non présents dans une partition de périphérique, mais uniquement dans l'espace RAM utilisé par le noyau pour stocker la configuration d'exécution et les informations sur les processus en cours. Ces systèmes de fichiers ne sont pas destinés au stockage conventionnel de fichiers, ils sont donc appelés pseudo-systèmes de fichiers et n'existent que lorsque le système est en cours d'exécution. Le répertoire `/proc` contient des fichiers avec des informations concernant les processus en cours et les ressources hardware. Parmi les fichiers importants dans `/proc` pour l'inspection du matériel, on trouve :

`/proc/cpuinfo`
: Liste des informations détaillées sur le(s) processeur(s) trouvé(s) par le système d'exploitation.

`/proc/interrupts`
: Une liste des numéros des interruptions par dispositif d'entrée/sortie pour chaque CPU.

`/proc/ioports`
: Liste les plages de ports d'entrée/sortie actuellement enregistrées et utilisées.

`/proc/dma`
: Liste les canaux DMA (accès direct à la mémoire) enregistrés en cours d'utilisation.

Les fichiers du répertoire `/sys` ont un rôle similaire à ceux du répertoire `/proc`. Cependant, le répertoire `/sys` a pour vocation spécifique de stocker les informations sur les périphériques et les données du noyau relatives au hardware, tandis que `/proc` contient également des informations sur les différentes structures de données du noyau, y compris les processus en cours d'exécution et la configuration.

Un autre répertoire directement lié aux périphériques dans un système Linux standard est `/dev`. Chaque fichier à l'intérieur de `/dev` est associé à un périphérique système, en particulier les périphériques de stockage. Un ancien disque dur IDE, par exemple, lorsqu'il est connecté au premier canal IDE de la carte mère, sera représenté par le fichier `/dev/hda`. Chaque partition de ce disque sera identifiée par `/dev/hda1`, `/dev/hda2` jusqu'à la dernière partition trouvée.

Les périphériques amovibles sont gérés par le sous-système *udev*, qui crée les périphériques correspondants dans `/dev`. Le noyau Linux capture l'événement de détection du hardware et le transmet au processus udev, qui identifie alors le périphérique et crée dynamiquement les fichiers correspondants dans `/dev`, en utilisant un jeu de règles prédéfinies.

Dans les distributions Linux actuelles, udev se charge de l'identification et de la configuration des périphériques déjà présents lors de la mise sous tension de la machine (*détection coldplug*) et des périphériques identifiés lorsque le système est en cours d'exécution (*détection hotplug*). Udev s'appuie sur *SysFS*, le pseudo-système de fichiers pour les informations relatives au hardware monté dans `/sys`.

NOTE

Hotplug est le terme utilisé pour désigner la détection et la configuration d'un périphérique lorsque le système est en cours d'exécution, par exemple lorsqu'un périphérique USB est inséré. Le noyau Linux supporte les fonctionnalités hotplug depuis la version 2.6, permettant à la plupart des bus système (PCI, USB, etc.) de déclencher des événements hotplug lorsqu'un périphérique est connecté ou déconnecté.

Au fur et à mesure que de nouveaux périphériques sont détectés, udev recherche une règle correspondante dans les règles prédéfinies stockées dans le répertoire `/etc/udev/rules.d/`. Les règles les plus importantes sont fournies par la distribution, mais de nouvelles règles peuvent être ajoutées pour des cas de figure spécifiques.

Périphériques de stockage

Sous Linux, les périphériques de stockage sont appelés périphériques bloc de manière générique, étant donné que les données de ces périphériques sont lues et écrites en blocs de données bufferisées de tailles et de positions différentes. Chaque périphérique bloc est identifié par un fichier dans le répertoire `/dev`, le nom du fichier dépendant du type de périphérique (IDE, SATA, SCSI, etc.) et de ses partitions. Les lecteurs CD/DVD et les disquettes floppy, par exemple, auront leur nom attribué en conséquence dans `/dev` : un lecteur CD/DVD connecté au second canal IDE sera identifié comme `/dev/hdc` (`/dev/hda` et `/dev/hdb` sont réservés aux périphériques maître et esclave sur le premier canal IDE) et un ancien lecteur de disquettes floppy sera identifié comme `/dev/fd0`, `/dev/fd1`, etc.

À partir de la version 2.4 du noyau Linux, la plupart des périphériques de stockage sont désormais identifiés comme s'il s'agissait de périphériques SCSI, quel que soit leur type de matériel. Les périphériques bloc IDE, SSD et USB seront préfixés par `sd`. Pour les disques IDE, le préfixe `sd` sera utilisé, mais la troisième lettre sera choisie selon que le lecteur est un maître ou un esclave (dans le premier canal IDE, le maître sera `sda` et l'esclave sera `sdb`). Les partitions sont listées par ordre numérique. Les chemins `/dev/sda1`, `/dev/sda2`, etc. sont utilisés pour la première et la deuxième partition du périphérique bloc identifié en premier et `/dev/sdb1`, `/dev/sdb2`, etc. sont utilisés pour identifier la première et la deuxième partition du périphérique bloc identifié en second. L'exception à ce schéma intervient avec les cartes mémoire (cartes SD) et les périphériques NVMe (SSD connectés au bus PCI Express). Pour les cartes SD, les chemins `/dev/mmcblk0p1`, `/dev/mmcblk0p2`, etc. sont utilisés pour la première et la deuxième partition

du périphérique identifié en premier et `/dev/mmcblk1p1`, `/dev/mmcblk1p2`, etc. sont utilisés pour identifier la première et la deuxième partition du périphérique identifié en second. Les périphériques NVMe reçoivent le préfixe `nvme`, comme dans `/dev/nvme0n1p1` et `/dev/nvme0n1p2`.

Exercices guidés

1. Admettons qu'un système d'exploitation soit incapable de démarrer après l'ajout d'un deuxième disque SATA au système. Sachant que tous les composants ne sont pas défectueux, quelle pourrait être la cause possible de cette défaillance ?

2. Imaginons que vous vouliez vous assurer que la carte graphique externe connectée au bus PCI de votre nouvel ordinateur de bureau est bien celle annoncée par le fabricant, mais l'ouverture du boîtier du PC annulera la garantie. Quelle commande pourrait être utilisée pour lister les détails de la carte graphique tels qu'ils ont été détectés par le système d'exploitation ?

3. La ligne suivante est un extrait de la sortie générée par la commande `lspci` :

   ```
   03:00.0 RAID bus controller: LSI Logic / Symbios Logic MegaRAID SAS 2208 [Thunderbolt]
   (rev 05)
   ```

   Quelle commande devez-vous exécuter pour identifier le module du noyau utilisé pour ce périphérique spécifique ?

4. Un administrateur système veut essayer différents paramètres pour le module de noyau `bluetooth` sans redémarrer le système. Cependant, toute tentative de déchargement du module avec `modprobe -r bluetooth` entraîne l'erreur suivante :

   ```
   modprobe: FATAL: Module bluetooth is in use.
   ```

   Quelle est la cause possible de cette erreur ?

Exercices d'approfondissement

1. Il n'est pas rare de tomber sur des machines anciennes dans des environnements de production, comme certains équipements qui utilisent une connectique obsolète pour communiquer avec l'ordinateur de contrôle, d'où la nécessité d'accorder une attention particulière à certaines particularités de ces machines anciennes. Certains serveurs x86 avec un ancien firmware BIOS, par exemple, ne démarreront pas si le clavier n'est pas détecté. Comment éviter ce problème particulier ?

2. Les systèmes d'exploitation construits autour du noyau Linux sont également disponibles pour une grande variété d'architectures informatiques autres que x86, comme dans les ordinateurs monocartes basés sur l'architecture ARM. Un utilisateur attentif remarquera l'absence de la commande `lspci` sur de telles machines, comme le Raspberry Pi. Quelle différence avec les machines x86 justifie cette absence ?

3. De nombreux routeurs disposent d'un port USB permettant la connexion d'un périphérique externe, comme un disque dur USB. Comme la plupart d'entre eux utilisent un système d'exploitation basé sur Linux, comment un disque dur USB externe sera-t-il nommé dans le répertoire `/dev/`, en supposant qu'aucun autre périphérique bloc conventionnel ne soit présent dans le routeur ?

4. En 2018, la vulnérabilité matérielle connue sous le nom de *Meltdown* a été découverte. Elle concerne presque tous les processeurs des différentes architectures. Les versions récentes du noyau Linux peuvent indiquer si le système actuel est vulnérable. Comment obtenir ces informations ?

Résumé

Cette leçon couvre les concepts fondamentaux sur la manière dont le noyau Linux gère les ressources matérielles, notamment dans l'architecture x86. La leçon aborde les sujets suivants :

- Comment les paramètres définis dans les utilitaires de configuration BIOS ou UEFI peuvent affecter la manière dont le système d'exploitation interagit avec le matériel.
- Comment utiliser les outils fournis par un système Linux standard pour obtenir des informations sur le hardware.
- Comment identifier les périphériques de stockage fixes et amovibles dans le système de fichiers.

Voici les procédures et les commandes abordées :

- Commandes pour inspecter le matériel détecté : `lspci` et `lsusb`.
- Commandes pour gérer les modules du noyau : `lsmod` et `modprobe`.
- Fichiers spéciaux liés au matériel, soit les fichiers trouvés dans le répertoire `/dev/` ou dans les pseudo-systèmes de fichiers dans `/proc/` et `/sys/`.

Réponses aux exercices guidés

1. Admettons qu'un système d'exploitation soit incapable de démarrer après l'ajout d'un deuxième disque SATA au système. Sachant que tous les composants ne sont pas défectueux, quelle pourrait être la cause possible de cette défaillance ?

   L'ordre des périphériques d'amorçage doit être configuré dans l'utilitaire de configuration BIOS, faute de quoi le BIOS risque de ne pas pouvoir exécuter le chargeur de démarrage.

2. Imaginons que vous vouliez vous assurer que la carte graphique externe connectée au bus PCI de votre nouvel ordinateur de bureau est bien celle annoncée par le fabricant, mais l'ouverture du boîtier du PC annulera la garantie. Quelle commande pourrait être utilisée pour lister les détails de la carte graphique tels qu'ils ont été détectés par le système d'exploitation ?

   La commande `lspci` fournira des informations détaillées sur tous les périphériques actuellement connectés au bus PCI.

3. La ligne suivante est un extrait de la sortie générée par la commande `lspci` :

   ```
   03:00.0 RAID bus controller: LSI Logic / Symbios Logic MegaRAID SAS 2208 [Thunderbolt]
   (rev 05)
   ```

   Quelle commande devez-vous exécuter pour identifier le module du noyau utilisé pour ce périphérique spécifique ?

   La commande `lspci -s 03:00.0 -v` ou `lspci -s 03:00.0 -k`

4. Un administrateur système veut essayer différents paramètres pour le module de noyau `bluetooth` sans redémarrer le système. Cependant, toute tentative de déchargement du module avec `modprobe -r bluetooth` entraîne l'erreur suivante :

   ```
   modprobe: FATAL: Module bluetooth is in use.
   ```

   Quelle est la cause possible de cette erreur ?

   Le module `bluetooth` est utilisé par un processus en cours.

Réponses aux exercices d'approfondissement

1. Il n'est pas rare de tomber sur des machines anciennes dans des environnements de production, comme certains équipements qui utilisent une connectique obsolète pour communiquer avec l'ordinateur de contrôle, d'où la nécessité d'accorder une attention particulière à certaines particularités de ces machines anciennes. Certains serveurs x86 avec un ancien firmware BIOS, par exemple, ne démarreront pas si le clavier n'est pas détecté. Comment éviter ce problème particulier ?

   L'utilitaire de configuration BIOS permet de désactiver le verrouillage de l'ordinateur en cas d'absence de clavier.

2. Les systèmes d'exploitation construits autour du noyau Linux sont également disponibles pour une grande variété d'architectures informatiques autres que x86, comme dans les ordinateurs monocartes basés sur l'architecture ARM. Un utilisateur attentif remarquera l'absence de la commande `lspci` sur de telles machines, comme le Raspberry Pi. Quelle différence avec les machines x86 justifie cette absence ?

   Contrairement à la plupart des machines x86, un ordinateur basé sur ARM comme le Raspberry Pi n'a pas de bus PCI, de sorte que la commande `lspci` ne sert à rien.

3. De nombreux routeurs disposent d'un port USB permettant la connexion d'un périphérique externe, comme un disque dur USB. Comme la plupart d'entre eux utilisent un système d'exploitation basé sur Linux, comment un disque dur USB externe sera-t-il nommé dans le répertoire `/dev/`, en supposant qu'aucun autre périphérique bloc conventionnel ne soit présent dans le routeur ?

   Les noyaux Linux modernes identifient les disques durs USB comme des périphériques SATA, le fichier correspondant sera donc `/dev/sda` puisqu'il n'existe aucun autre périphérique bloc conventionnel dans le système.

4. En 2018, la vulnérabilité matérielle connue sous le nom de *Meltdown* a été découverte. Elle concerne presque tous les processeurs des différentes architectures. Les versions récentes du noyau Linux peuvent indiquer si le système actuel est vulnérable. Comment obtenir ces informations ?

   Le fichier `/proc/cpuinfo` a une ligne qui affiche les bugs connus pour le CPU en question, comme `bugs : cpu_meltdown`.

101.2 Démarrage du système

Référence aux objectifs de LPI

LPIC-1 v5, Exam 101, Objective 101.2

Valeur

3

Key knowledge areas

- Passage de commandes au chargeur de démarrage et passage de paramètres d'amorçage au noyau.
- Démontrer sa connaissance des séquences d'amorçage depuis le BIOS / UEFI jusqu'à l'achèvement des séquences de démarrage.
- Compréhension de l'init SysV et de systemd.
- Sensibilisation à Upstart.
- Consulter les événements de la phase de démarrage dans les journaux (logs).

Domaines de connaissance les plus importants

- `dmesg`
- `journalctl`
- BIOS
- UEFI
- bootloader
- kernel
- `initramfs`
- `init`

- SysVinit
- systemd

101.2 Leçon 1

Certification :	LPIC-1
Version :	5.0
Thème :	101 Architecture système
Objectif :	101.2 Démarrer le système
Leçon :	1 sur 1

Introduction

Afin de contrôler la machine, le composant principal du système d'exploitation — le noyau — doit être chargé par un programme appelé le *chargeur de démarrage*, qui est lui-même chargé par un micrologiciel (*firmware*) préinstallé comme le BIOS ou l'UEFI. Le chargeur de démarrage peut être personnalisé pour transmettre des paramètres au noyau, par exemple la partition qui contient le système de fichiers racine ou le mode dans lequel le système d'exploitation doit s'exécuter. Une fois chargé, le noyau poursuit le processus de démarrage en identifiant et en configurant le matériel. Au terme de ce processus, le noyau lance l'utilitaire chargé de démarrer et de gérer les services du système.

NOTE Sur certaines distributions Linux, les commandes exécutées dans cette leçon peuvent nécessiter les droits root.

BIOS ou UEFI

Les procédures exécutées par les machines x86 pour lancer le chargeur de démarrage diffèrent en fonction de l'utilisation du BIOS ou de l'UEFI. Le BIOS, abréviation de *Basic Input/Output System*, est un programme stocké dans une puce de mémoire non volatile attachée à la carte mère, exécuté à chaque fois que l'ordinateur est mis sous tension. Ce type de programme est appelé *firmware* et son emplacement de stockage est distinct des autres périphériques de stockage que le système peut avoir. Le BIOS part du principe que les 440 premiers octets du premier périphérique de stockage — suivant l'ordre défini dans l'interface de configuration du BIOS — constituent la première phase du chargeur de démarrage (également appelée *bootstrap* ou amorçage). Les 512 premiers octets d'un périphérique de stockage sont appelés le MBR (*Master Boot Record*) pour les périphériques de stockage qui utilisent le schéma de partition DOS standard et, en plus de la première phase du chargeur de démarrage, contiennent la table de partitions. Si le MBR ne contient pas de données correctes, le système ne pourra pas démarrer, à moins qu'une méthode alternative ne soit utilisée.

D'une manière générale, les étapes préopératoires pour amorcer un système équipé d'un BIOS sont les suivantes :

1. L'auto-test d'allumage POST (*Power-On Self-Test*) est exécuté pour identifier les défaillances matérielles élémentaires dès que la machine est mise sous tension.
2. Le BIOS active les composants basiques pour charger le système, comme la sortie vidéo, le clavier et les médias de stockage.
3. Le BIOS exécute la première phase du chargeur de démarrage à partir du MBR (les 440 premiers octets du premier périphérique, tel qu'il est défini dans l'interface de configuration du BIOS).
4. La première phase du chargeur de démarrage appelle la deuxième phase du chargeur de démarrage, chargée de présenter les options de démarrage et de charger le noyau.

L'UEFI, abréviation de *Unified Extensible Firmware Interface*, se distingue du BIOS sur plusieurs points essentiels. Tout comme le BIOS, l'UEFI est également un micrologiciel, mais il est capable d'identifier des partitions et de lire un grand nombre de systèmes de fichiers qui s'y trouvent. L'UEFI ne s'appuie pas sur le MBR et ne prend en compte que les seuls paramètres stockés dans sa mémoire non volatile (*NVRAM*) attachée à la carte mère. Ces définitions indiquent l'emplacement des programmes compatibles UEFI, appelés *applications EFI*, qui seront exécutés automatiquement ou invoqués à partir d'un menu de démarrage. Les applications EFI peuvent être des chargeurs d'amorçage, des sélecteurs de systèmes d'exploitation, des outils de diagnostic et de réparation système, etc. Ils doivent figurer dans une partition classique de périphérique de stockage et dans un système de fichiers compatible. Les systèmes de fichiers standard compatibles

sont le FAT12, le FAT16 et le FAT32 pour les dispositifs en bloc et l'ISO-9660 pour les supports optiques. Cette approche permet le déploiement d'outils beaucoup plus sophistiqués que ceux qui sont possibles avec le BIOS.

La partition qui contient les applications EFI est appelée *EFI System Partition* ou simplement ESP. Cette partition ne doit en aucun cas être partagée avec d'autres systèmes de fichiers du système comme le système de fichiers racine ou les systèmes de fichiers contenant les données des utilisateurs. Le répertoire EFI dans la partition ESP contient les applications référencées par les entrées enregistrées dans la NVRAM.

D'une manière générale, les étapes préopératoires pour amorcer un système avec UEFI sont les suivantes :

1. L'auto-test d'allumage POST (*Power-On Self-Test*) est exécuté pour identifier les défaillances matérielles élémentaires dès que la machine est mise sous tension.
2. L'UEFI active les composants basiques pour charger le système, comme la sortie vidéo, le clavier et les médias de stockage.
3. Le micrologiciel de l'UEFI lit les paramètres stockés dans la NVRAM pour exécuter l'application EFI prédéfinie stockée dans le système de fichiers de la partition ESP. En règle générale, cette application EFI prédéfinie est un chargeur de démarrage.
4. Si l'application EFI prédéfinie est un chargeur de démarrage, celui-ci chargera le noyau pour démarrer le système d'exploitation.

La norme UEFI comprend également une option appelée *Secure Boot*, qui autorise uniquement l'exécution des applications EFI signées, c'est-à-dire des applications EFI autorisées par le fabricant du matériel. Cette fonctionnalité renforce la protection contre les malwares, mais peut entraver l'installation de systèmes d'exploitation non couverts par la garantie du fabricant.

Le chargeur de démarrage

Le chargeur de démarrage le plus populaire pour Linux dans l'architecture x86 est GRUB (*Grand Unified Bootloader*). Dès qu'il est appelé par le BIOS ou par l'UEFI, GRUB affiche une liste de systèmes d'exploitation disponibles au démarrage. Il peut arriver que la liste n'apparaisse pas automatiquement, mais elle peut être invoquée en appuyant sur `Maj` pendant que GRUB est appelé par le BIOS. Avec les systèmes UEFI, la touche `Échap` doit être utilisée à la place.

Depuis le menu GRUB, il est possible de choisir lequel des noyaux installés doit être chargé ainsi que de transmettre de nouveaux paramètres à celui-ci. La plupart des paramètres du noyau suivent le schéma `option=valeur`. Voici quelques-uns des paramètres les plus pertinents du noyau :

`acpi`

Active/désactive le support de l'ACPI. `acpi=off` désactivera le support de l'ACPI.

`init`

Définit un initialiseur système alternatif. À titre d'exemple, `init=/bin/bash` définira le shell Bash comme initialiseur. Cela signifie qu'une session shell sera lancée juste après le processus de démarrage du noyau.

`systemd.unit`

Définit la cible *systemd* à activer. Par exemple, `systemd.unit=graphical.target`. Systemd accepte également les niveaux d'exécution numériques tels que définis pour *SysV*. Pour activer le niveau d'exécution 1, par exemple, il suffit d'inclure le chiffre `1` ou la lettre `S` (pour "single") comme paramètre du noyau.

`mem`

Définit la quantité de RAM disponible pour le système. Ce paramètre est utile pour les machines virtuelles afin de limiter la quantité de RAM disponible pour chaque système invité. L'utilisation de `mem=512M` limitera à 512 mégaoctets la quantité de RAM disponible pour un système invité donné.

`maxcpus`

Limite le nombre de processeurs (ou cœurs de processeur) visibles pour le système dans les machines à multiprocesseurs symétriques. C'est également valable pour les machines virtuelles. Une valeur de `0` désactive le support des machines multiprocesseurs et a le même effet que le paramètre de noyau `nosmp`. Le paramètre `maxcpus=2` limitera à deux le nombre de processeurs disponibles pour le système d'exploitation.

`quiet`

Masque la plupart des messages de démarrage.

`vga`

Sélectionne un mode vidéo. Le paramètre `vga=ask` affichera une liste des modes disponibles au choix.

`root`

Définit la partition racine, distincte de celle pré-configurée dans le chargeur de démarrage. Par exemple, `root=/dev/sda3`.

`rootflags`

Options de montage pour le système de fichiers racine.

`ro`

Effectue le montage initial du système de fichiers racine en lecture seule.

`rw`

Permet l'écriture dans le système de fichiers racine lors du montage initial.

La modification des paramètres du noyau n'est pas nécessaire en général, mais elle peut s'avérer utile pour détecter et résoudre les problèmes liés au système d'exploitation. Les paramètres du noyau doivent être ajoutés au fichier `/etc/default/grub` à la ligne `GRUB_CMDLINE_LINUX` pour les rendre persistants après chaque redémarrage. Un nouveau fichier de configuration pour le chargeur d'amorçage doit être généré à chaque fois que `/etc/default/grub` change, ce qui est effectué par la commande `grub-mkconfig -o /boot/grub/grub.cfg`. Une fois que le système d'exploitation tourne, les paramètres du noyau utilisés pour le chargement de la session en cours sont disponibles en lecture dans le fichier `/proc/cmdline`.

NOTE La configuration de GRUB sera abordée plus en détail dans une leçon ultérieure.

Initialisation du système

En dehors du noyau, le système d'exploitation dépend d'autres composants qui fournissent les fonctionnalités voulues. Un grand nombre de ces composants sont chargés au cours du processus d'initialisation du système et vont du simple script shell au programme de service plus complexe. Les scripts sont souvent utilisés pour effectuer des tâches éphémères qui s'exécutent et se terminent pendant le processus d'initialisation du système. Les services, également connus sous le nom de *démons*, sont susceptibles d'être actifs en permanence car ils peuvent être responsables des aspects intrinsèques du système d'exploitation.

La variété des procédés permettant d'intégrer dans une distribution Linux des scripts de démarrage et des démons présentant les caractéristiques les plus diverses est énorme, ce qui a historiquement entravé le développement d'une solution unique répondant aux attentes des mainteneurs et des utilisateurs de toutes les distributions Linux. Ceci étant dit, n'importe quel outil choisi par les mainteneurs des distributions pour remplir cette fonction sera au moins capable de démarrer, d'arrêter et de redémarrer les services du système. Ces actions sont souvent effectuées par le système lui-même après une mise à jour logicielle, par exemple, mais l'administrateur du système devra presque toujours redémarrer manuellement le service après avoir apporté des modifications à son fichier de configuration.

Il est également pratique pour un administrateur système de pouvoir activer un ensemble particulier de démons en fonction du contexte. Il devrait être possible, par exemple, de ne faire tourner que le strict minimum de services pour effectuer des tâches de maintenance du système.

NOTE

À proprement parler, le système d'exploitation n'est constitué que du noyau et des composants qui contrôlent le matériel et gèrent tous les processus. Il est cependant courant d'utiliser le terme "système d'exploitation" de manière plus souple, pour désigner tout un groupe de programmes distincts qui constituent l'environnement logiciel dans lequel un utilisateur peut effectuer les tâches informatiques de base.

L'initialisation du système d'exploitation commence lorsque le chargeur de démarrage charge le noyau dans la RAM. Ensuite, le noyau prend en charge le CPU et commence à détecter et à configurer les aspects fondamentaux du système d'exploitation comme la configuration matérielle de base et l'adressage de la mémoire.

Le noyau ouvre alors le disque mémoire initial (*initial RAM filesystem* ou *initramfs*). L'initramfs est une archive qui contient un système de fichiers utilisé comme système de fichiers racine temporaire lors du processus de démarrage. Le rôle principal d'un fichier initramfs est de fournir les modules nécessaires pour que le noyau puisse accéder au "vrai" système de fichiers racine du système d'exploitation.

Dès que le système de fichiers racine est disponible, le noyau monte tous les systèmes de fichiers configurés dans `/etc/fstab` et exécute ensuite le premier programme, un utilitaire nommé `init`. Le programme `init` est responsable de l'exécution de tous les scripts d'initialisation et des démons du système. Il existe des implémentations distinctes de ces initialiseurs de système en dehors de l'init traditionnel, comme *systemd* et *Upstart*. Une fois que le programme `init` est chargé, l'initramfs est retiré de la RAM.

SysV init

Un gestionnaire de services basé sur la norme SysV init contrôle les démons et les ressources qui seront disponibles en se basant sur le concept de niveaux d'exécution ou *runlevels*. Les niveaux d'exécution sont numérotés de 0 à 6 et définis par les mainteneurs des distributions pour répondre à des objectifs spécifiques. Les seules définitions de niveau d'exécution communes à toutes les distributions sont les niveaux d'exécution 0, 1 et 6.

systemd

Systemd est un gestionnaire de système et de services moderne avec une couche de compatibilité pour les commandes et les niveaux d'exécution SysV. systemd a une structure parallèle, utilise les sockets et D-Bus pour l'activation des services, l'exécution de démons à la demande, la surveillance des processus avec *cgroups*, le *support des instantanés*, la récupération des sessions système, le contrôle des points de montage et un contrôle des services basé sur les dépendances. Ces dernières années, la plupart des grandes distributions Linux ont progressivement adopté systemd comme gestionnaire de système par défaut.

Upstart

Comme systemd, Upstart est un remplaçant d'init. Le principal objectif d'Upstart est d'accélérer le processus de démarrage en parallélisant le processus de chargement des services du système. Upstart était utilisé par les distributions basées sur Ubuntu dans les versions précédentes, mais aujourd'hui il a cédé la place à systemd.

Inspection de l'initialisation

Il peut y avoir des erreurs au cours du processus de démarrage, mais elles ne sont pas forcément critiques au point de provoquer l'arrêt complet du système d'exploitation. Néanmoins, ces erreurs peuvent compromettre le comportement attendu du système. Toutes les erreurs génèrent des messages qui peuvent être utilisés pour de futures investigations, car ils contiennent des informations précieuses sur le moment et la manière dont l'erreur s'est produite. Même lorsqu'aucun message d'erreur n'est généré, les informations recueillies pendant le processus de démarrage peuvent être utiles à des fins de réglage et de configuration.

L'espace mémoire où le noyau stocke ses messages, y compris les messages de démarrage, est appelé le tampon circulaire du noyau (*kernel ring buffer*). Les messages sont conservés dans le tampon circulaire même lorsqu'ils ne sont pas affichés pendant le processus d'initialisation, comme lorsqu'une animation est affichée à la place. Cependant, le tampon circulaire du noyau perd tous les messages lorsque le système est éteint ou lorsque la commande `dmesg --clear` est exécutée. Invoquée sans options, la commande `dmesg` affiche les messages actuels dans le tampon circulaire du noyau :

```
$ dmesg
[    5.262389] EXT4-fs (sda1): mounted filesystem with ordered data mode. Opts: (null)
[    5.449712] ip_tables: (C) 2000-2006 Netfilter Core Team
[    5.460286] systemd[1]: systemd 237 running in system mode.
[    5.480138] systemd[1]: Detected architecture x86-64.
[    5.481767] systemd[1]: Set hostname to <torre>.
[    5.636607] systemd[1]: Reached target User and Group Name Lookups.
[    5.636866] systemd[1]: Created slice System Slice.
[    5.637000] systemd[1]: Listening on Journal Audit Socket.
[    5.637085] systemd[1]: Listening on Journal Socket.
[    5.637827] systemd[1]: Mounting POSIX Message Queue File System...
[    5.638639] systemd[1]: Started Read required files in advance.
[    5.641661] systemd[1]: Starting Load Kernel Modules...
[    5.661672] EXT4-fs (sda1): re-mounted. Opts: errors=remount-ro
[    5.694322] lp: driver loaded but no devices found
[    5.702609] ppdev: user-space parallel port driver
[    5.705384] parport_pc 00:02: reported by Plug and Play ACPI
```

```
[    5.705468] parport0: PC-style at 0x378 (0x778), irq 7, dma 3
[PCSPP,TRISTATE,COMPAT,EPP,ECP,DMA]
[    5.800146] lp0: using parport0 (interrupt-driven).
[    5.897421] systemd-journald[352]: Received request to flush runtime journal from PID 1
```

La sortie de `dmesg` peut compter des centaines de lignes, et le listing ci-dessus ne contient donc que l'extrait qui montre le noyau en train d'appeler le gestionnaire de services systemd. Les valeurs au début de chaque ligne correspondent au nombre de secondes par rapport à l'instant où le noyau a commencé à être chargé.

Dans les systèmes basés sur systemd, la commande `journalctl` affichera les messages d'initialisation avec les options `-b`, `--boot`, `-k` ou `--dmesg`. La commande `journalctl --list-boots` affiche une liste de numéros de démarrage relatifs au démarrage en cours, leur empreinte d'identification et l'horodatage du premier et du dernier message correspondant :

```
$ journalctl --list-boots
 -4 9e5b3eb4952845208b841ad4dbefa1a6 Thu 2019-10-03 13:39:23 -03—Thu 2019-10-03 13:40:30 -03
 -3 9e3d79955535430aa43baa17758f40fa Thu 2019-10-03 13:41:15 -03—Thu 2019-10-03 14:56:19 -03
 -2 17672d8851694e6c9bb102df7355452c Thu 2019-10-03 14:56:57 -03—Thu 2019-10-03 19:27:16 -03
 -1 55c0d9439bfb4e85a20a62776d0dbb4d Thu 2019-10-03 19:27:53 -03—Fri 2019-10-04 00:28:47 -03
  0 08fbbebd9f964a74b8a02bb27b200622 Fri 2019-10-04 00:31:01 -03—Fri 2019-10-04 10:17:01 -03
```

Les journaux d'initialisation précédents sont également conservés dans les systèmes basés sur systemd, de sorte que les messages des sessions précédentes du système d'exploitation peuvent toujours être inspectés. Si les options `-b 0` ou `--boot=0` sont fournies, alors les messages pour le démarrage en cours seront affichés. Les options `-b -1` ou `--boot=-1` afficheront les messages de la précédente initialisation. Les options `-b -2` ou `--boot=-2` montreront les messages de l'avant-dernière initialisation, et ainsi de suite. L'extrait suivant montre le noyau en train d'appeler le gestionnaire de services systemd pour le dernier processus d'initialisation :

```
$ journalctl -b 0
oct 04 00:31:01 ubuntu-host kernel: EXT4-fs (sda1): mounted filesystem with ordered data
mode. Opts: (null)
oct 04 00:31:01 ubuntu-host kernel: ip_tables: (C) 2000-2006 Netfilter Core Team
oct 04 00:31:01 ubuntu-host systemd[1]: systemd 237 running in system mode.
oct 04 00:31:01 ubuntu-host systemd[1]: Detected architecture x86-64.
oct 04 00:31:01 ubuntu-host systemd[1]: Set hostname to <torre>.
oct 04 00:31:01 ubuntu-host systemd[1]: Reached target User and Group Name Lookups.
oct 04 00:31:01 ubuntu-host systemd[1]: Created slice System Slice.
oct 04 00:31:01 ubuntu-host systemd[1]: Listening on Journal Audit Socket.
oct 04 00:31:01 ubuntu-host systemd[1]: Listening on Journal Socket.
```

```
oct 04 00:31:01 ubuntu-host systemd[1]: Mounting POSIX Message Queue File System...
oct 04 00:31:01 ubuntu-host systemd[1]: Started Read required files in advance.
oct 04 00:31:01 ubuntu-host systemd[1]: Starting Load Kernel Modules...
oct 04 00:31:01 ubuntu-host kernel: EXT4-fs (sda1): re-mounted. Opts:
commit=300,barrier=0,errors=remount-ro
oct 04 00:31:01 ubuntu-host kernel: lp: driver loaded but no devices found
oct 04 00:31:01 ubuntu-host kernel: ppdev: user-space parallel port driver
oct 04 00:31:01 ubuntu-host kernel: parport_pc 00:02: reported by Plug and Play ACPI
oct 04 00:31:01 ubuntu-host kernel: parport0: PC-style at 0x378 (0x778), irq 7, dma 3
[PCSPP,TRISTATE,COMPAT,EPP,ECP,DMA]
oct 04 00:31:01 ubuntu-host kernel: lp0: using parport0 (interrupt-driven).
oct 04 00:31:01 ubuntu-host systemd-journald[352]: Journal started
oct 04 00:31:01 ubuntu-host systemd-journald[352]: Runtime journal
(/run/log/journal/abb765408f3741ae9519ab3b96063a15) is 4.9M, max 39.4M, 34.5M free.
oct 04 00:31:01 ubuntu-host systemd-modules-load[335]: Inserted module 'lp'
oct 04 00:31:01 ubuntu-host systemd-modules-load[335]: Inserted module 'ppdev'
oct 04 00:31:01 ubuntu-host systemd-modules-load[335]: Inserted module 'parport_pc'
oct 04 00:31:01 ubuntu-host systemd[1]: Starting Flush Journal to Persistent Storage...
```

Les messages d'initialisation ainsi que les autres messages émis par le système d'exploitation sont stockés dans des fichiers à l'intérieur du répertoire `/var/log/`. Si une erreur critique se produit et que le système d'exploitation est incapable de poursuivre le processus d'initialisation après le chargement du noyau et de l'initramfs, un support de démarrage alternatif pourrait être utilisé pour démarrer le système et accéder au système de fichiers correspondant. Ensuite, les fichiers en dessous de `/var/log/` peuvent être examinés pour déterminer les raisons possibles de l'interruption du processus de démarrage. Les options `-D` ou `--directory` de la commande `journalctl` peuvent être utilisées pour lire les logs dans des répertoires autres que `/var/log/journal/`, qui est l'emplacement par défaut des messages de journalisation de systemd. Étant donné que les messages du journal de systemd ne sont pas stockés au format texte brut, la commande `journalctl` est indispensable pour les lire.

Exercices guidés

1. Sur une machine équipée d'un firmware BIOS, où se situe le binaire d'amorçage ?

2. Le firmware UEFI gère les fonctionnalités étendues fournies par des programmes externes, les applications EFI. Toutefois, ces applications disposent de leur propre emplacement spécifique. À quel endroit du système les applications EFI se trouvent-elles ?

3. Les chargeurs de démarrage permettent de passer des paramètres personnalisés au noyau avant de le charger. Admettons que le système soit incapable de démarrer parce que l'emplacement du système de fichiers racine est mal renseigné. Comment le système de fichiers racine approprié, situé dans `/dev/sda3`, serait-il fourni comme paramètre au noyau ?

4. Le processus de démarrage d'une machine sous Linux se termine par le message suivant :

   ```
   ALERT! /dev/sda3 does not exist. Dropping to a shell!
   ```

   Quelle est la cause probable de ce problème ?

Exercices d'approfondissement

1. Le chargeur de démarrage présente une liste de systèmes d'exploitation au choix lorsque plusieurs systèmes d'exploitation sont installés sur la machine. Cependant, il arrive qu'un système d'exploitation nouvellement installé écrase le MBR du disque dur, ce qui supprime la première phase du chargeur de démarrage en rendant l'autre système d'exploitation inaccessible. Pourquoi cela ne serait-il pas le cas sur une machine équipée d'un firmware UEFI ?

2. Quelle est la conséquence habituelle de l'installation d'un noyau personnalisé sans fournir une image initramfs appropriée ?

3. Le journal d'initialisation compte des centaines de lignes, de sorte que le résultat de la commande `dmesg` est souvent transmis à une commande de pagination — comme la commande `less` — pour faciliter la lecture. Quelle option de `dmesg` va automatiquement paginer sa sortie, en éliminant la nécessité d'utiliser explicitement une commande de pagination ?

4. Un disque dur contenant tout le système de fichiers d'une machine hors ligne a été retiré et relié à une machine en état de marche en tant que disque secondaire. En supposant que son point de montage est `/mnt/hd`, comment `journalctl` serait-il invoqué pour inspecter le contenu des fichiers de journalisation situés dans `/mnt/hd/var/log/journal/` ?

Résumé

Cette leçon couvre la séquence de démarrage dans un système Linux standard. Une bonne connaissance du fonctionnement du processus de démarrage d'un système Linux permet d'éviter les erreurs qui peuvent rendre le système inaccessible. La leçon aborde les sujets suivants :

- Les différences entre les modes de démarrage du BIOS et de l'UEFI.
- Les phases typiques de l'initialisation du système.
- La récupération des messages de démarrage.

Voici les procédures et les commandes abordées :

- Les paramètres du noyau les plus courants.
- Les commandes pour lire les messages de démarrage : `dmesg` et `journalctl`.

Réponses aux exercices guidés

1. Sur une machine équipée d'un firmware BIOS, où se situe le binaire d'amorçage ?

   Dans le MBR du premier périphérique de stockage, tel que défini dans l'utilitaire de configuration du BIOS.

2. Le firmware UEFI gère les fonctionnalités étendues fournies par des programmes externes, les applications EFI. Toutefois, ces applications disposent de leur propre emplacement spécifique. À quel endroit du système les applications EFI se trouvent-elles ?

   Les applications EFI sont stockées dans la partition système EFI (ESP ou *EFI System Partition*), située dans n'importe quel bloc de stockage disponible doté d'un système de fichiers compatible (généralement un système de fichiers FAT32).

3. Les chargeurs de démarrage permettent de passer des paramètres personnalisés au noyau avant de le charger. Admettons que le système soit incapable de démarrer parce que l'emplacement du système de fichiers racine est mal renseigné. Comment le système de fichiers racine approprié, situé dans `/dev/sda3`, serait-il fourni comme paramètre au noyau ?

   Le paramètre `root` doit être utilisé, comme dans `root=/dev/sda3`.

4. Le processus de démarrage d'une machine sous Linux se termine par le message suivant :

   ```
   ALERT! /dev/sda3 does not exist. Dropping to a shell!
   ```

   Quelle est la cause probable de ce problème ?

   Le noyau n'a pas pu trouver le périphérique `/dev/sda3`, renseigné comme système de fichiers racine.

Answers to Explorational Exercises

1. Le chargeur de démarrage présente une liste de systèmes d'exploitation au choix lorsque plusieurs systèmes d'exploitation sont installés sur la machine. Cependant, il arrive qu'un système d'exploitation nouvellement installé écrase le MBR du disque dur, ce qui supprime la première phase du chargeur de démarrage en rendant l'autre système d'exploitation inaccessible. Pourquoi cela ne serait-il pas le cas sur une machine équipée d'un firmware UEFI ?

   Les machines UEFI n'utilisent pas le MBR du disque dur pour stocker la première phase du chargeur de démarrage.

2. Quelle est la conséquence habituelle de l'installation d'un noyau personnalisé sans fournir une image initramfs appropriée ?

   Le système de fichiers racine peut être inaccessible si son type a été compilé comme un module de noyau externe.

3. Le journal d'initialisation compte des centaines de lignes, de sorte que le résultat de la commande `dmesg` est souvent transmis à une commande de pagination — comme la commande `less` — pour faciliter la lecture. Quelle option de `dmesg` va automatiquement paginer sa sortie, en éliminant la nécessité d'utiliser explicitement une commande de pagination ?

   Les commandes `dmesg -H` ou `dmesg --human` activeront la pagination par défaut.

4. Un disque dur contenant tout le système de fichiers d'une machine hors ligne a été retiré et relié à une machine en état de marche en tant que disque secondaire. En supposant que son point de montage est `/mnt/hd`, comment `journalctl` serait-il invoqué pour inspecter le contenu des fichiers de journalisation situés dans `/mnt/hd/var/log/journal/` ?

   Avec les options `journalctl -D /mnt/hd/var/log/journal` ou `journalctl --directory=/mnt/hd/var/log/journal`

101.3 Changement de niveaux d'exécution / des cibles de démarrage de systemd et arrêt ou redémarrage du système

Référence aux objectifs de LPI

LPIC-1 v5, Exam 101, Objective 101.3

Valeur

3

Domaines de connaissance les plus importants

- Paramétrage du niveau d'exécution ou de la cible systemd par défaut.
- Passage d'un niveau d'exécution / d'une cible systemd à un(e) autre, y compris en mode mono-utilisateur.
- Arrêt et redémarrage du système en ligne de commande.
- Avertissement des utilisateurs avant un changement de niveau d'exécution / de cible systemd ou pour d'autres événements système importants.
- Terminer les processus correctement.
- Connaissance de base de acpid.

Liste partielle de termes, fichiers et utilitaires utilisés pour cet objectif

- `/etc/inittab`
- `shutdown`
- `init`
- `/etc/init.d/`
- `telinit`
- `systemd`

- `systemctl`
- `/etc/systemd/`
- `/usr/lib/systemd/`
- `wall`

101.3 Leçon 1

Certification :	LPIC-1
Version :	5.0
Thème :	101 Architecture système
Objectif :	101.3 Changement de niveaux d'exécution / des cibles de démarrage de systemd et arrêt ou redémarrage du système
Leçon :	1 sur 1

Introduction

Un point caractéristique commun aux systèmes d'exploitation qui suivent les principes de conception d'Unix est l'utilisation de processus séparés pour contrôler les différentes fonctions du système. Ces processus appelés *démons* (ou, plus généralement, *services*) sont également chargés des fonctionnalités étendues sous-jacentes au système d'exploitation, comme les services d'application réseau (serveur HTTP, partage de fichiers, courrier électronique, etc.), les bases de données, la configuration à la demande, etc. Même si Linux utilise un noyau monolithique, de nombreux aspects bas niveau du système d'exploitation sont affectés par les démons, comme l'équilibrage de charge ou la configuration du pare-feu.

Le choix des démons à activer dépend de la vocation du système. Le jeu de démons actifs doit également être modifiable en cours d'exécution, de sorte que les services puissent être démarrés et arrêtés sans avoir à redémarrer l'ensemble du système. Pour résoudre ce problème, toutes les grandes distributions Linux proposent une forme d'outil de gestion des services pour contrôler le système.

Les services peuvent être contrôlés par des scripts shell ou par un programme et ses fichiers de configuration. La première méthode est implémentée par la norme *SysVinit*, également connue sous le nom de *System V* ou simplement *SysV*. La deuxième méthode est implémentée par *systemd* et *Upstart*. Historiquement, les gestionnaires de services basés sur SysV étaient les plus utilisés par les distributions Linux. De nos jours, les gestionnaires de services basés sur systemd sont plus courants dans la plupart des distributions Linux. Le gestionnaire de services est le premier programme lancé par le noyau au cours du processus de démarrage, son PID (numéro d'identification du processus) est donc toujours `1`.

SysVinit

Un gestionnaire de services basé sur la norme SysVinit fournira un ensemble prédéfini d'états du système, appelés *runlevels* (niveaux d'exécution), et les fichiers de scripts du service correspondants à exécuter. Les *runlevels* sont numérotés de `0` à `6`, et ils sont généralement affectés aux objectifs suivants :

Niveau 0

Arrêt du système.

Niveau 1, s ou *single*

Mode mono-utilisateur, sans réseau et autres fonctionnalités non-essentielles (pour la maintenance).

Niveaux 2, 3 ou 4

Mode multi-utilisateur. Les utilisateurs peuvent se connecter par la console ou par le réseau. Les niveaux 2 et 4 ne sont pas souvent utilisés.

Niveau 5

Mode multi-utilisateur. Il est équivalent au niveau 3, avec la connexion en mode graphique.

Niveau 6

Redémarrage du système.

Le programme chargé de la gestion des niveaux d'exécution et des démons/ressources associés est `/sbin/init`. Lors de l'initialisation du système, le programme `init` identifie le niveau d'exécution requis, défini par un paramètre du noyau ou dans le fichier `/etc/inittab`, et charge les scripts correspondants qui y sont référencés pour le niveau d'exécution donné. Chaque niveau d'exécution peut être associé à une série de fichiers de services, généralement des scripts dans le répertoire `/etc/init.d/`. Comme tous les niveaux d'exécution ne se valent pas selon les différentes distributions Linux, une description succincte de la finalité du niveau d'exécution peut

également être trouvée dans les distributions basées sur SysV.

La syntaxe du fichier `/etc/inittab` utilise ce format :

```
id:niveaux:action:processus
```

L'`id` est un nom générique comportant jusqu'à quatre caractères, utilisé pour identifier l'entrée. L'entrée `niveaux` est une liste de numéros de niveaux d'exécution pour lesquels une action spécifiée doit être exécutée. Le terme `action` définit comment `init` va exécuter le processus indiqué par le terme `processus`. Les actions disponibles sont les suivantes :

`boot`
: Le processus sera exécuté lors de l'initialisation du système. Le champ `niveaux` est ignoré.

`bootwait`
: Le processus sera exécuté pendant l'initialisation du système et `init` attendra qu'il se termine pour continuer. Le champ `niveaux` est ignoré.

`sysinit`
: Le processus sera exécuté après l'initialisation du système, quel que soit le niveau d'exécution. Le champ `niveaux` est ignoré.

`wait`
: Le processus sera exécuté pour les niveaux d'exécution donnés et `init` attendra qu'il se termine pour continuer.

`respawn`
: Le processus sera relancé s'il est interrompu.

`ctrlaltdel`
: Le processus sera exécuté lorsque le processus `init` reçoit le signal `SIGINT`, déclenché lorsque la séquence de touches `Ctrl` + `Alt` + `Del` est actionnée.

Le niveau d'exécution par défaut — celui qui sera choisi si aucun autre n'est fourni comme paramètre du noyau — est également défini dans `/etc/inittab`, dans l'entrée `id:x:initdefault`. Le `x` est le nombre correspondant au niveau d'exécution par défaut. Ce nombre ne doit jamais être égal à `0` ou `6`, étant donné que cela entraînerait l'arrêt ou le redémarrage du système dès la fin du processus de démarrage. Un fichier `/etc/inittab` typique est présenté ci-dessous :

```
# Niveau d'exécution par défaut
id:3:initdefault:

# Script de configuration exécuté au démarrage
si::sysinit:/etc/init.d/rcS

# Action effectuée au niveau d'exécution S (mono-utilisateur)
~:S:wait:/sbin/sulogin

# Configuration pour chaque niveau d'exécution
l0:0:wait:/etc/init.d/rc 0
l1:1:wait:/etc/init.d/rc 1
l2:2:wait:/etc/init.d/rc 2
l3:3:wait:/etc/init.d/rc 3
l4:4:wait:/etc/init.d/rc 4
l5:5:wait:/etc/init.d/rc 5
l6:6:wait:/etc/init.d/rc 6

# Action effectuée à la combinaison de touches ctrl+alt+del
ca::ctrlaltdel:/sbin/shutdown -r now

# Activer les consoles pour les niveaux d'exécution 2 et 3
1:23:respawn:/sbin/getty tty1 VC linux
2:23:respawn:/sbin/getty tty2 VC linux
3:23:respawn:/sbin/getty tty3 VC linux
4:23:respawn:/sbin/getty tty4 VC linux

# Activer le port série en plus pour le niveau d'exécution 3
# terminals ttyS0 and ttyS1 (modem) consoles
S0:3:respawn:/sbin/getty -L 9600 ttyS0 vt320
S1:3:respawn:/sbin/mgetty -x0 -D ttyS1
```

La commande `telinit q` devra être exécutée après chaque modification du fichier `/etc/inittab`. L'argument `q` (ou `Q`) indique à `init` de recharger sa configuration. Cette étape est cruciale pour éviter un arrêt du système suite à une mauvaise configuration dans `/etc/inittab`.

Les scripts utilisés par `init` pour configurer chaque niveau d'exécution sont rangés dans le répertoire `/etc/init.d/`. Chaque niveau d'exécution dispose d'un répertoire associé dans `/etc/`, nommé `/etc/rc0.d/`, `/etc/rc1.d/`, `/etc/rc2.d/`, etc., avec les scripts censés être exécutés au démarrage du niveau d'exécution correspondant. Étant donné qu'un même script peut être utilisé par différents niveaux d'exécution, les fichiers contenus dans ces répertoires ne sont que des liens symboliques vers les scripts réels dans `/etc/init.d/`. Par ailleurs, la première lettre du nom du

lien dans le répertoire du niveau d'exécution indique si le service doit être démarré ou arrêté pour le niveau d'exécution correspondant. Un nom de lien commençant par la lettre `K` détermine que le service sera arrêté à l'entrée du niveau d'exécution (*kill*). S'il commence par la lettre `S`, le service sera démarré lors de l'entrée dans le niveau d`exécution (*start*). Le répertoire `/etc/rc1.d/`, par exemple, comportera de nombreux liens vers des scripts réseau commençant par la lettre `K`, étant donné que le niveau d'exécution 1 est le niveau d'exécution mono-utilisateur, sans connectivité réseau.

La commande `runlevel` indique le niveau d'exécution en cours pour le système. Cette commande affiche deux valeurs, la première est le niveau d'exécution précédent et la seconde correspond au niveau d'exécution actuel :

```
$ runlevel
N 3
```

La lettre `N` dans la sortie montre que le niveau d'exécution n'a pas changé depuis le dernier démarrage. Dans l'exemple, `runlevel 3` correspond au niveau d'exécution en cours du système.

Le même programme `init` peut être utilisé pour basculer à chaud entre les niveaux d'exécution d'un système, sans qu'il soit nécessaire de redémarrer. La commande `telinit` peut également être utilisée pour basculer d'un niveau à l'autre. Par exemple, les commandes `telinit 1`, `telinit s` ou `telinit S` feront passer le système au niveau d'exécution 1.

systemd

Actuellement, systemd constitue la boîte à outils la plus utilisée pour gérer les ressources et les services du système, qui sont désignés sous le nom d'unités (*units) par systemd. Une unité est composée d'un nom, d'un type et d'un fichier de configuration correspondant. À titre d'exemple, l'unité pour un processus de serveur _httpd* (comme le serveur web Apache) sera `httpd.service` sur les distributions de la famille Red Hat et son fichier de configuration sera également appelé `httpd.service` (sur les distributions de la famille Debian, cette unité est appelée `apache2.service`).

On distingue sept types d'unités systemd :

`service`

Le type d'unité le plus courant, pour les ressources actives du système qui peuvent être initiées, interrompues et rechargées.

`socket`

Le type d'unité `socket` peut être un socket de système de fichiers ou un socket réseau. Toutes les unités `socket` ont une unité `service` correspondante, chargée lorsque le socket est sollicité.

`device`

Une unité `device` est associée à un périphérique matériel identifié par le noyau. Un périphérique ne sera considéré comme une unité systemd que s'il existe une règle udev à cet effet. Une unité `device` peut être utilisée pour résoudre les dépendances de configuration lorsque certains composants matériels sont détectés, étant donné que les propriétés de la règle udev peuvent être utilisées comme paramètres pour l'unité `device`.

`mount`

Une unité `mount` est une définition de point de montage dans le système de fichiers, similaire à une entrée dans `/etc/fstab`.

`automount`

Une unité `automount` est également une définition de point de montage dans le système de fichiers, mais montée automatiquement. Chaque unité `automount` dispose d'une unité `mount` correspondante, qui est lancée lors de l'accès au point de montage `automount`.

`target`

Une unité `target` (cible) est un regroupement d'autres unités, gérées comme une seule unité.

`snapshot`

Une unité `snapshot` est un état sauvegardé du gestionnaire systemd (non disponible sur toutes les distributions Linux).

La commande principale pour contrôler les unités systemd est `systemctl`. La commande `systemctl` est utilisée pour exécuter toutes les tâches liées à l'activation, la désactivation, l'exécution, l'interruption, la surveillance des unités, etc. Pour une unité fictive appelée `unit.service`, par exemple, les opérations `systemctl` les plus courantes seront :

`systemctl start unit.service`

Démarre `unit`.

`systemctl stop unit.service`

Arrête `unit`.

`systemctl restart unit.service`
Relance `unit`.

`systemctl status unit.service`
Affiche l'état de `unit`, y compris l'état d'activation.

`systemctl is-active unit.service`
Affiche *active* si `unit`est en état de marche ou *inactive* dans le cas contraire.

`systemctl enable unit.service`
Active `unit`, c'est-à-dire que `unit` se chargera lors de l'initialisation du système.

`systemctl disable unit.service`
`unit` ne démarrera pas avec le système.

`systemctl is-enabled unit.service`
Vérifie si `unit` démarre le système. La réponse est enregistrée dans la variable `$?`. La valeur `0` indique que `unit` démarre avec le système et la valeur `1` indique que `unit` ne démarre pas avec le système.

NOTE

Les installations plus récentes de systemd indiqueront plutôt la configuration d'une unité pour le démarrage. Par exemple :

```
$ systemctl is-enabled apparmor.service
enabled
```

Si aucune autre unité du même nom n'existe dans le système, le suffixe après le point peut être omis. Si, par exemple, il n'y a qu'une seule unité `httpd` de type `service`, alors un simple `httpd` est suffisant comme paramètre d'unité pour `systemctl`.

La commande `systemctl` peut également contrôler les cibles système (*system targets*). L'unité `multi-user.target`, par exemple, combine toutes les unités requises par l'environnement système multi-utilisateurs. Il est similaire au niveau d'exécution 3 dans un système basé sur SysV.

La commande `systemctl isolate` bascule entre différentes cibles. Ainsi, pour basculer manuellement vers la cible `multi-user` :

```
# systemctl isolate multi-user.target
```

Il existe des cibles correspondantes aux niveaux d'exécution SysV, en allant de

`runlevel0.target` jusqu'à `runlevel6.target`. En revanche, systemd n'utilise pas le fichier `/etc/inittab`. Pour modifier la cible système par défaut, l'option `systemd.unit` peut être ajoutée à la liste des paramètres du noyau. Par exemple, pour utiliser `multi-user.target` comme cible standard, le paramètre du noyau sera `systemd.unit=multi-user.target`. Tous les paramètres du noyau peuvent être rendus persistants en modifiant la configuration du chargeur d'amorçage.

Une autre manière de changer la cible par défaut consiste à modifier le lien symbolique `/etc/systemd/system/default.target` de manière à ce qu'il pointe vers la cible souhaitée. La redéfinition du lien peut être effectuée par le biais de la commande `systemctl` :

```
# systemctl set-default multi-user.target
```

De même, on peut déterminer la cible de démarrage par défaut du système avec la commande suivante :

```
$ systemctl get-default
graphical.target
```

Comme pour les systèmes avec SysV, la cible par défaut ne doit jamais pointer vers `shutdown.target`, puisqu'elle correspond au niveau d'exécution 0 (arrêt).

Les fichiers de configuration associés à chaque unité se trouvent dans le répertoire `/lib/systemd/system/`. La commande `systemctl list-unit-files` affiche la liste de toutes les unités disponibles et indique si elles sont activées au démarrage du système. L'option `--type` sélectionnera uniquement les unités pour un certain type, comme dans `systemctl list-unit-files --type=service` et `systemctl list-unit-files --type=target`.

Les unités actives ou ayant été actives pendant la session système en cours peuvent être listées avec la commande `systemctl list-units`. Comme pour l'option `list-unit-files`, la commande `systemctl list-units --type=service` sélectionnera uniquement les unités de type `service` et la commande `systemctl list-units --type=target` sélectionnera uniquement les unités de type `target`.

Systemd est également chargé du déclenchement et de la réponse aux événements liés à l'alimentation. La commande `systemctl suspend` mettra le système en mode économique, en gardant les données actuelles en mémoire. La commande `systemctl hibernate` va copier toutes les données en mémoire sur le disque, de sorte que l'état actuel du système peut être récupéré après son extinction. Les actions associées à ces événements sont définies dans le fichier `/etc/systemd/logind.conf` ou dans des fichiers individuels à l'intérieur du répertoire

`/etc/systemd/logind.conf.d/`. Cependant, cette fonctionnalité de systemd ne peut être utilisée que lorsqu'il n'y a aucun autre gestionnaire d'alimentation en cours d'exécution dans le système, comme le démon `acpid`. Le démon `acpid` est le principal gestionnaire d'énergie pour Linux et permet un ajustement plus fin des actions consécutives à des événements liés à l'alimentation, comme la fermeture du couvercle du portable, une batterie faible ou le niveau de charge de la batterie.

Upstart

Les scripts d'initialisation utilisés par Upstart se trouvent dans le répertoire `/etc/init/`. Les services du système peuvent être affichés avec la commande `initctl list`, qui indique également l'état actuel des services et, le cas échéant, leur PID.

```
# initctl list
avahi-cups-reload stop/waiting
avahi-daemon start/running, process 1123
mountall-net stop/waiting
mountnfs-bootclean.sh start/running
nmbd start/running, process 3085
passwd stop/waiting
rc stop/waiting
rsyslog start/running, process 1095
tty4 start/running, process 1761
udev start/running, process 1073
upstart-udev-bridge start/running, process 1066
console-setup stop/waiting
irqbalance start/running, process 1842
plymouth-log stop/waiting
smbd start/running, process 1457
tty5 start/running, process 1764
failsafe stop/waiting
```

Chaque action Upstart dispose de sa propre commande dédiée. Par exemple, la commande `start` peut être utilisée pour lancer un sixième terminal virtuel :

```
# start tty6
```

L'état actuel d'une ressource peut être vérifié avec la commande `status` :

```
# status tty6
```

```
tty6 start/running, process 3282
```

Quant à l'interruption d'un service, elle se fait avec la commande `stop` :

```
# stop tty6
```

Upstart n'utilise pas le fichier `/etc/inittab` pour définir les niveaux d'exécution, par contre les commandes traditionnelles `runlevel` et `telinit` permettent toujours de vérifier les niveaux d'exécution et d'alterner entre eux.

NOTE Upstart a été développé pour la distribution Ubuntu Linux afin de faciliter le démarrage parallèle des processus. Ubuntu a cessé d'utiliser Upstart en 2015, date à laquelle la distribution est passée de Upstart à systemd.

Arrêt et redémarrage

Une commande très classique utilisée pour arrêter ou redémarrer le système est appelée `shutdown`, comme on peut s'y attendre. La commande `shutdown` ajoute des fonctions supplémentaires au processus de mise hors tension : elle envoie automatiquement un avertissement à tous les utilisateurs connectés à leurs sessions shell et empêche toute nouvelle connexion. La commande `shutdown` agit comme un intermédiaire aux procédures SysV ou systemd, c'est-à-dire qu'elle exécute l'action requise en appelant l'action correspondante dans le gestionnaire de services utilisé par le système.

Après l'exécution de `shutdown`, tous les processus reçoivent le signal `SIGTERM`, suivi du signal `SIGKILL`, puis le système s'arrête ou change de niveau d'exécution. Par défaut, lorsqu'aucune des options `-h` ou `-r` n'est utilisée, le système passe au niveau d'exécution 1, c'est-à-dire au mode mono-utilisateur. Pour modifier les options par défaut pour `shutdown`, la commande devra être exécutée avec la syntaxe suivante :

```
$ shutdown [option] time [message]
```

Seul le paramètre `time` est requis. Le paramètre `time` définit le moment où l'action requise sera exécutée, en acceptant les formats suivants :

`hh:mm`
: Ce format spécifie le moment de l'exécution en heures et minutes.

`+m`

Ce format précise le nombre de minutes à attendre avant l'exécution.

`now` or `+0`

Ce format définit l'exécution immédiate.

Le paramètre `message` est le texte de l'avertissement envoyé dans toutes les sessions de terminal des utilisateurs connectés.

L'implémentation SysV permet de limiter les utilisateurs qui pourront redémarrer la machine en appuyant sur `Ctrl` + `Alt` + `Del`. Cela est possible en ajoutant l'option `-a` pour la commande `shutdown` à la ligne relative à `ctrlaltdel` dans le fichier `/etc/inittab`. En procédant ainsi, seuls les utilisateurs dont le nom d'utilisateur figure dans le fichier `/etc/shutdown.allow` pourront redémarrer le système grâce à la combinaison de touches `Ctrl` + `Alt` + `Del`.

La commande `systemctl` peut également être utilisée pour arrêter ou redémarrer la machine sur les systèmes basés sur systemd. Pour redémarrer le système, la commande `systemctl reboot` doit être utilisée. Pour arrêter le système, utilisez la commande `systemctl poweroff`. Les deux commandes nécessitent les droits root, étant donné que les utilisateurs normaux ne peuvent pas effectuer ce genre d'opération.

NOTE

Certaines distributions Linux associent `poweroff` et `reboot` à `systemctl` sous forme de commandes individuelles. Par exemple :

```
$ sudo which poweroff
/usr/sbin/poweroff
$ sudo ls -l /usr/sbin/poweroff
lrwxrwxrwx 1 root root 14 Aug 20 07:50 /usr/sbin/poweroff -> /bin/systemctl
```

Toutes les activités de maintenance ne requièrent pas l'arrêt ou le redémarrage du système. En revanche, lorsqu'il s'avère nécessaire de faire passer le système en mode mono-utilisateur, il est important d'avertir les utilisateurs connectés afin qu'ils ne soient pas affectés par une interruption brutale de leurs activités.

Tout comme la commande `shutdown` lors de l'arrêt ou du redémarrage du système, la commande `wall` est capable d'envoyer un message aux sessions de terminal de tous les utilisateurs connectés. Pour ce faire, il suffit à l'administrateur système de fournir un fichier ou d'écrire le message directement comme paramètre de la commande `wall`.

Exercices guidés

1. Comment la commande `telinit` peut-elle être utilisée pour redémarrer le système ?

1. Que deviendront les services liés au fichier `/etc/rc1.d/K90network` lorsque le système passe au niveau d'exécution 1 ?

2. En utilisant la commande `systemctl`, comment un utilisateur pourrait-il vérifier si l'unité `sshd.service` est en cours d'exécution ?

3. Sur un système basé sur systemd, quelle commande doit être exécutée pour permettre le lancement de l'unité `sshd.service` lors de l'initialisation du système ?

Exercices d'approfondissement

1. Sur un système basé sur SysV, supposons que le niveau d'exécution par défaut défini dans `/etc/inittab` est 3, mais le système démarre toujours au niveau d'exécution 1. Quelle est la cause probable de cette situation ?

2. Bien que le fichier `/sbin/init` soit présent dans les systèmes basés sur systemd, il n'est qu'un lien symbolique vers un autre fichier exécutable. Dans de tels systèmes, quel est le fichier pointé par `/sbin/init` ?

3. Comment peut-on vérifier la cible par défaut du système dans un système basé sur systemd ?

4. Comment peut-on annuler un redémarrage du système programmé avec la commande `shutdown` ?

Résumé

Cette leçon couvre les principaux outils utilisés pour gérer les services des distributions Linux. Les outils SysVinit, systemd et Upstart ont chacun leur propre approche pour contrôler les services et les états du système. La leçon aborde les sujets suivants :

- Les services du système et leur rôle au sein du système d'exploitation.
- Concepts et utilisation de base des commandes SysVinit, systemd et Upstart.
- Démarrer, arrêter et redémarrer correctement les services du système et le système lui-même.

Voici les procédures et les commandes abordées :

- Les commandes et les fichiers liés à SysVinit, comme `init`, `/etc/inittab` et `telinit`.
- La commande principale de systemd : `systemctl`.
- Les commandes d'Upstart : `initctl`, `status`, `start`, `stop`.
- Les commandes traditionnelles de gestion de l'alimentation comme `shutdown`, et la commande `wall`.

Réponses aux exercices guidés

1. Comment la commande `telinit` peut-elle être utilisée pour redémarrer le système ?

   La commande `telinit 6` va basculer vers le niveau d'exécution 6 et donc redémarrer le système.

2. Que deviendront les services liés au fichier `/etc/rc1.d/K90network` lorsque le système passe au niveau d'exécution 1 ?

   Du fait de la lettre `K` au début du nom du fichier, les services correspondants seront arrêtés.

3. En utilisant la commande `systemctl`, comment un utilisateur pourrait-il vérifier si l'unité `sshd.service` est en cours d'exécution ?

   Avec la commande `systemctl status sshd.service` ou `systemctl is-active sshd.service`.

4. Sur un système basé sur systemd, quelle commande doit être exécutée pour permettre le lancement de l'unité `sshd.service` lors de l'initialisation du système ?

   La commande `systemctl enable sshd.service`, exécutée par root.

Réponses aux exercices d'approfondissement

1. Sur un système basé sur SysV, supposons que le niveau d'exécution par défaut défini dans `/etc/inittab` est 3, mais le système démarre toujours au niveau d'exécution 1. Quelle est la cause probable de cette situation ?

   Les paramètres `1` ou `S` peuvent figurer dans la liste des paramètres du noyau.

2. Bien que le fichier `/sbin/init` soit présent dans les systèmes basés sur systemd, il n'est qu'un lien symbolique vers un autre fichier exécutable. Dans de tels systèmes, quel est le fichier pointé par `/sbin/init` ?

   Le binaire systemd principal : `/lib/systemd/systemd`.

3. Comment peut-on vérifier la cible par défaut du système dans un système basé sur systemd ?

   Le lien symbolique `/etc/systemd/system/default.target` pointera vers le fichier unité défini comme cible par défaut. La commande `systemctl get-default` peut également être utilisée.

4. Comment peut-on annuler un redémarrage du système programmé avec la commande `shutdown` ?

   La commande `shutdown -c` doit être utilisée.

Linux
Professional
Institute

Thème 102 : Installation de Linux et gestion de paquetages

102.1 Conception du schéma de partitionnement

Référence aux objectifs de LPI

LPIC-1 v5, Exam 101, Objective 102.1

Valeur

2

Domaines de connaissance les plus importants

- Répartition des systèmes de fichiers et de l'espace d'échange (swap) sur des partitions ou des disques séparés.
- Ajustement du schéma de partitionnement en fonction de l'usage prévu du système.
- Vérification que la partition /boot est conforme aux besoins de l'architecture matérielle pour le démarrage.
- Connaissance des caractéristiques de base de LVM.

Liste partielle de termes, fichiers et utilitaires utilisés pour cet objectif

- Système de fichiers racine `/`
- Système de fichiers `/var`
- Système de fichiers `/home`
- Système de fichiers `/boot`
- Partition système EFI - EFI System Partition (ESP)
- Espace d'échange swap
- Points de montage
- Partitions

102.1 Leçon 1

Certification :	LPIC-1
Version :	5.0
Thème :	102 Installation de Linux et gestion des paquets
Objectif :	102.1 Conception du schéma de partitionnement
Leçon :	1 of 1

Introduction

Pour répondre à cet objectif, vous devez comprendre le rapport entre les *disques*, les *partitions*, les *systèmes de fichiers* et les *volumes*.

Imaginez un disque (ou un *dispositif de stockage*, puisque les appareils modernes ne contiennent pas de "disques" à proprement parler) comme un "conteneur physique" pour vos données.

Avant qu'un disque ne puisse être utilisé par un ordinateur, il doit être partitionné. Une partition est un sous-ensemble logique du disque physique, comme une "clôture" logique. Le partitionnement est un moyen de "compartimenter" les informations stockées sur le disque, en séparant par exemple les données du système d'exploitation des données des utilisateurs.

Chaque disque a besoin d'au moins une partition mais peut en avoir plusieurs si nécessaire, les informations correspondantes sont stockées dans une table de partition. Cette table comprend les informations sur le premier et le dernier secteur de la partition ainsi que le type de partition. Elle contient également des détails supplémentaires sur chaque partition.

A l'intérieur de chaque partition se trouve un système de fichiers. Le système de fichiers décrit la

manière dont les informations sont effectivement stockées sur le disque. Ces informations comprennent la façon dont les répertoires sont organisés, les relations entre eux, l'emplacement des données pour chaque fichier, etc.

Les partitions ne peuvent pas s'étendre sur plusieurs disques. Or, en utilisant le *Logical Volume Manager* (LVM), plusieurs partitions peuvent être combinées, même d'un disque à l'autre, pour former un seul volume logique.

Les volumes logiques font abstraction des contraintes des périphériques physiques et vous permettent de travailler avec des *pools* d'espace disque qui peuvent être combinés ou distribués de manière beaucoup plus souple que les partitions traditionnelles. LVM est utile dans les situations où il vous faut ajouter de l'espace à une partition sans pour autant être obligé de migrer les données vers un périphérique plus spacieux.

Dans cet objectif, vous allez apprendre à élaborer un schéma de partitionnement de disque pour un système Linux, en allouant des systèmes de fichiers et l'espace d'échange à des partitions ou des disques séparés lorsque cela est nécessaire.

La *création* et la *gestion* des partitions et des systèmes de fichiers seront abordées dans des leçons ultérieures.

Les points de montage

Avant de pouvoir accéder à un système de fichiers sous Linux, il faut le *monter*. Cela revient à rattacher le système de fichiers à un point spécifique de l'arborescence de votre système, nommé *point de montage*.

Une fois monté, le contenu du système de fichiers sera disponible sous le point de montage. Par exemple, imaginez que vous ayez une partition avec les données personnelles de vos utilisateurs (leurs répertoires personnels), contenant les répertoires `/john`, `/jack` et `/carol`. Une fois monté sous `/home`, le contenu de ces répertoires sera disponible sous `/home/john`, `/home/jack` et `/home/carol`.

Le point de montage doit exister avant le montage du système de fichiers. Vous ne pouvez pas monter une partition sous `/mnt/userdata` si ce répertoire n'existe pas. En revanche, si le répertoire existe et qu'il contient des fichiers, ces fichiers ne seront pas disponibles tant que vous n'aurez pas démonté le système de fichiers. Si vous affichez le contenu du répertoire, vous verrez les fichiers stockés sur le système de fichiers monté, et non pas le contenu initial du répertoire.

Les systèmes de fichiers peuvent être montés où vous voulez. Toutefois, il existe une série de bonnes pratiques à suivre pour faciliter l'administration du système.

Traditionnellement, `/mnt` était le répertoire sous lequel tous les périphériques externes étaient montés, et un certain nombre de *points d'ancrage* pré-configurés pour les périphériques courants comme les lecteurs de CD-ROM (`/mnt/cdrom`) et les disquettes floppy (`/mnt/floppy`) existaient dans ce répertoire.

Il a été remplacé par `/media`, qui est maintenant le point de montage par défaut pour tous les supports amovibles (par exemple les disques externes, les clés USB, les lecteurs de cartes mémoire, les disques optiques, etc.) connectés au système.

Sur la plupart des distributions Linux et des environnements de bureau modernes, les périphériques amovibles sont automatiquement montés sous `/media/USER/LABEL` lorsqu'ils sont connectés au système, où `USER` est le nom d'utilisateur et `LABEL` l'étiquette du périphérique. Par exemple, une clé USB étiquetée `FlashDrive` et insérée par l'utilisateur `john` sera montée sous `/media/john/FlashDrive/`. La manière dont cela est géré peut varier en fonction de l'environnement de bureau.

Ceci étant dit, lorsque vous avez besoin de monter *manuellement* un système de fichiers, il est recommandé de le monter sous `/mnt`. Les commandes spécifiques pour contrôler le montage et le démontage des systèmes de fichiers sous Linux seront abordées dans une autre leçon.

Chaque chose à sa place

Sous Linux, il y a certains répertoires que vous devriez envisager de stocker sur des partitions distinctes. Il y a plusieurs raisons à cela : par exemple, en conservant les fichiers liés au chargeur de démarrage (stockés sur `/boot`) sur une *partition de démarrage*, vous êtes sûr que votre système pourra toujours démarrer en cas de plantage du système de fichiers racine.

Le fait de conserver les répertoires personnels des utilisateurs (sous `/home`) sur une partition séparée facilite la réinstallation du système sans risquer de toucher par inadvertance aux données des utilisateurs. En conservant les données relatives à un serveur web ou une base de données (généralement sous `/var`) sur une partition séparée (ou même un disque séparé), l'administration du système est facilitée si vous devez ajouter de l'espace disque supplémentaire pour ces cas de figure.

Il peut même y avoir des raisons de performance pour garder certains répertoires sur des partitions séparées. Vous pouvez conserver le système de fichiers racine (`/`) sur un disque SSD rapide, et des répertoires plus volumineux comme `/home` et `/var` sur des disques durs plus lents qui offrent beaucoup plus d'espace pour une fraction du coût.

La partition de démarrage (`/boot`)

La partition de démarrage contient les fichiers utilisés par le chargeur de démarrage pour charger le système d'exploitation. Sur les systèmes Linux, le chargeur de démarrage est généralement GRUB2 ou, sur les systèmes plus anciens, GRUB Legacy. La partition est généralement montée sous `/boot` et ses fichiers sont stockés dans `/boot/grub`.

D'un point de vue technique, une partition de démarrage n'est pas nécessaire, puisque dans la plupart des cas, GRUB est capable de monter la partition racine (`/`) et de charger les fichiers depuis un répertoire `/boot` séparé.

Toutefois, une partition de démarrage séparée peut être souhaitable pour des raisons de sécurité (afin de garantir que le système démarre même en cas de plantage du système de fichiers racine), ou si vous souhaitez utiliser un système de fichiers que le chargeur de démarrage ne peut pas gérer dans la partition racine, ou s'il utilise une méthode de chiffrement ou de compression qui n'est pas prise en charge.

La partition de démarrage est généralement la première partition du disque. En effet, le BIOS des PC IBM d'origine gérait les disques en utilisant des *cylindres*, des *têtes* et des *secteurs* (CHS pour *Cylinder/Head/Sector*), avec un maximum de 1024 cylindres, 256 têtes et 63 secteurs, ce qui donne une taille de disque maximale de 528 Mo (504 Mo sous MS-DOS). Cela signifie que tout ce qui dépasse cette marque ne serait pas accessible sur les anciens systèmes, à moins d'utiliser un schéma d'adressage de disque différent (comme l'adressage par bloc logique ou LBA pour *Logical Block Addressing*).

Ainsi, pour garantir une compatibilité maximale, la partition de démarrage est généralement située au début du disque et se termine avant le cylindre 1024 (528 Mo), ce qui garantit que, quoi qu'il arrive, la machine sera toujours capable de charger le noyau.

Comme la partition de démarrage ne stocke que les fichiers nécessaires au chargeur de démarrage, le disque mémoire initial et les images du noyau, elle peut être assez petite selon les normes actuelles. Une bonne taille se situe aux alentours de 300 Mo.

La partition système EFI (ESP)

La partition système EFI (ESP pour *EFI System Partition*) est utilisée par les machines basées sur le *Unified Extensible Firmware Interface* (UEFI) pour stocker les chargeurs de démarrage et les images de noyau pour les systèmes d'exploitation installés.

Cette partition est formatée avec un système de fichiers basé sur FAT. Sur un disque partitionné avec une table de partitions GUID, il possède un identificateur global unique de `C12A7328-F81F-11D2-BA4B-00A0C93EC93B`. Si le disque a été formaté selon le schéma de partitionnement MBR,

l'ID de la partition est `0xEF`.

Sur les machines qui tournent sous Microsoft Windows, cette partition est généralement la première sur le disque, même si cela n'est pas obligatoire. La partition EFI est créée (ou approvisionnée) par le système d'exploitation lors de l'installation, et sur un système Linux elle est montée sous `/boot/efi`.

La partition `/home`

Chaque utilisateur du système dispose d'un répertoire d'accueil pour stocker ses fichiers personnels et ses préférences, la plupart d'entre eux se trouvent sous `/home`. En règle générale, le répertoire personnel correspond au nom d'utilisateur, de sorte que l'utilisateur John aura son répertoire sous `/home/john`.

Il y a cependant des exceptions. Par exemple, le répertoire personnel de l'utilisateur root est `/root`, et certains services système peuvent avoir associé des utilisateurs à d'autres répertoires personnels.

Il n'y a pas de règle pour déterminer la taille d'une partition pour le répertoire `/home` (la partition `home`). Vous devez prendre en compte le nombre d'utilisateurs du système et la manière dont il sera utilisé. Un utilisateur qui ne fait que de la navigation web et du traitement de texte aura besoin de moins d'espace que celui qui fait du montage vidéo, par exemple.

Les données variables (`/var`)

Ce répertoire contient des "données variables", c'est-à-dire des fichiers et des répertoires dans lesquels le système doit pouvoir écrire pendant son fonctionnement. Cela inclut les journaux système (dans `/var/log`), les fichiers temporaires (`/var/tmp`) et les données d'application mises en cache (dans `/var/cache`).

`/var/www/html` est également le répertoire par défaut des fichiers de données pour le serveur Web Apache et `/var/lib/mysql` est l'emplacement par défaut des fichiers de bases de données pour le serveur MySQL. Toutefois, ces deux emplacements peuvent être modifiés.

Une bonne raison de prévoir une partition séparée pour `/var` est la stabilité. Un grand nombre d'applications et de processus écrivent dans `/var` et ses sous-répertoires, comme `/var/log` ou `/var/tmp`. Un processus défaillant peut écrire des données jusqu'à ce qu'il n'y ait plus d'espace libre sur le système de fichiers.

Si `/var` est sous `/` cela peut déclencher un *kernel panic* et une dégradation du système de fichiers, ce qui entraîne une situation difficile à récupérer. Mais si `/var` est conservé dans une partition séparée, le système de fichiers racine ne sera pas affecté.

Comme pour `/home`, il n'y a pas de règle universelle pour déterminer la taille d'une partition pour `/var`, car elle varie selon la manière dont le système est utilisé. Sur un ordinateur familial, elle peut ne prendre que quelques gigaoctets. Mais sur un serveur de bases de données ou un serveur web, il faut sans doute beaucoup plus d'espace. Dans ce type de scénario, il peut être judicieux de mettre `/var` sur une partition d'un disque distinct de la partition racine en ajoutant une protection supplémentaire contre la défaillance physique du disque.

Le swap

La partition d'échange est utilisée pour basculer des pages de mémoire de la RAM vers le disque si cela est nécessaire. Cette partition doit être d'un type spécifique, et elle doit être configurée avec un utilitaire approprié appelé `mkswap` avant de pouvoir être utilisée.

La partition swap ne peut pas être montée comme les autres, ce qui veut dire que vous ne pouvez pas y accéder comme à un répertoire normal et voir son contenu.

Un système peut avoir plusieurs partitions d'échange (bien que cela soit peu courant) et Linux supporte également l'utilisation de *fichiers* d'échange au lieu de partitions, ce qui peut être utile pour augmenter rapidement l'espace de swap en cas de besoin.

La taille de la partition d'échange est une question controversée. L'ancienne règle des débuts de Linux ("deux fois la quantité de mémoire vive") peut ne plus s'appliquer selon la manière dont le système est utilisé et la quantité de mémoire vive physique installée.

Dans la documentation de Red Hat Enterprise Linux 7, Red Hat recommande ceci :

Quantité de RAM	Taille de swap recommandée	Taille de swap recommandée avec l'hibernation
< 2 Go de RAM	2x la quantité de RAM	3x la quantité de RAM
2-8 Go de RAM	Égale à la quantité de RAM	2x la quantité de RAM
8-64 Go de RAM	Au moins 4 Go	1.5x la quantité de RAM
> 64 Go de RAM	Au moins 4 Go	Non recommandé

Bien évidemment, la quantité de swap peut dépendre de la charge de travail. Si la machine fait tourner un service critique, tel qu'une base de données, un serveur web ou un serveur SAP, il est conseillé de consulter la documentation pour ces services (ou l'éditeur de votre logiciel) pour avoir une recommandation.

NOTE Pour en savoir plus sur la création et l'activation des partitions d'échange et des fichiers swap, reportez-vous à l'objectif 104.1 du LPIC-1.

LVM

Nous avons déjà évoqué l'organisation des disques en une ou plusieurs partitions, chaque partition contenant un système de fichiers qui décrit comment sont stockés les fichiers et les métadonnées associées. L'un des inconvénients du partitionnement est que l'administrateur système doit décider à l'avance de la répartition de l'espace disque disponible sur un périphérique. Cela peut poser des problèmes par la suite, si une partition nécessite plus d'espace que prévu initialement. Bien sûr, les partitions peuvent être redimensionnées, mais cela peut être impossible si, par exemple, il n'y a plus d'espace libre sur le disque.

La gestion des volumes logiques (LVM pour *Logical Volume Management*) est une forme de virtualisation du stockage qui offre aux administrateurs système une approche plus souple de la gestion de l'espace disque que le partitionnement traditionnel. L'objectif de LVM est de faciliter la gestion des besoins de stockage de vos utilisateurs finaux. L'unité de base est le volume physique (PV pour *Physical Volume*), qui est un périphérique bloc sur votre système comme une partition de disque ou une grappe RAID.

Les PV sont regroupés en groupes de volumes (VG pour *Volume Groups*) qui font abstraction des périphériques sous-jacents et sont considérés comme un seul périphérique logique, avec la capacité de stockage combinée des PV qui les constituent.

Chaque volume d'un groupe de volumes (VG) est subdivisé en segments de taille fixe appelés *extensions*. Les extensions sur un volume physique (PV) sont appelées extensions physiques (PE pour *Physical Extensions*), tandis que celles sur un volume logique (LV) sont des extensions logiques (LE pour *Logical Extensions*). En règle générale, chaque extension logique est mappée à une extension physique, mais cela peut varier lorsque des fonctionnalités telles que la mise en miroir des disques sont utilisées.

Les groupes de volumes peuvent être subdivisés en volumes logiques (LV), qui fonctionnent de la même manière que les partitions, mais avec plus de souplesse.

La taille d'un volume logique, telle que spécifiée lors de sa création, est en fait définie par la taille des extensions physiques (4 Mo par défaut) multipliée par le nombre d'extensions sur le volume. Partant de là, il est facile de comprendre que pour augmenter la capacité d'un volume logique, par exemple, il suffit à l'administrateur système d'ajouter d'autres extensions à partir du pool disponible dans le groupe de volumes. De même, des extensions peuvent être enlevées pour rétrécir le LV.

Une fois qu'un volume logique est créé, il est considéré par le système d'exploitation comme n'importe quel périphérique bloc. Un périphérique sera créé dans `/dev`, nommé `/dev/NOMVG/NOMLV`, où `NOMVG` est le nom du groupe de volumes, et `NOMLV` est le nom du volume

logique.

Ces périphériques peuvent être formatés avec le système de fichiers souhaité en utilisant des outils standards (comme `mkfs.ext4`, par exemple) et montés en utilisant les méthodes habituelles, soit manuellement avec la commande `mount`, soit automatiquement en les ajoutant au fichier `/etc/fstab`.

Exercices guidés

1. Sur les systèmes Linux, où sont stockés les fichiers du chargeur de démarrage GRUB ?

2. Où doit se terminer la partition de démarrage pour garantir qu'un PC sera toujours capable de charger le noyau ?

3. À quel endroit la partition EFI est-elle généralement montée ?

4. Lors du montage manuel d'un système de fichiers, sous quel répertoire doit-il généralement être monté ?

Exercices d'approfondissement

1. Quelle est la plus petite unité à l'intérieur d'un groupe de volumes VG ?

2. Comment définit-on la taille d'un volume logique LV ?

3. Sur un disque formaté selon le schéma de partitionnement MBR, quel est l'ID de la partition système EFI ?

4. En dehors des partitions d'échange, comment peut-on augmenter rapidement l'espace swap sur un système Linux ?

Résumé

Dans cette leçon, vous avez abordé le partitionnement, les répertoires qui sont généralement conservés dans des partitions séparées et la raison pour laquelle on procède ainsi. Nous avons également abordé une présentation de LVM (*Logical Volume Management*) et montré comment il peut offrir un moyen plus flexible pour allouer vos données et votre espace disque par rapport au partitionnement traditionnel.

Les fichiers, les notions et les outils suivants ont été abordés :

`/`

Le système de fichiers racine Linux.

`/var`

L'emplacement standard pour les "données variables", les données qui peuvent diminuer et augmenter avec le temps.

`/home`

Le répertoire parent standard pour les répertoires personnels des utilisateurs ordinaires d'un système.

`/boot`

L'emplacement standard des fichiers du chargeur de démarrage, du noyau Linux et du disque mémoire initial.

Partition système EFI (ESP)

Utilisé par les systèmes qui utilisent l'UEFI pour le stockage des fichiers de démarrage du système.

Espace swap

Utilisé pour basculer les pages de mémoire du noyau lorsque la RAM est fortement sollicitée.

Points de montage

Emplacements des répertoires où un périphérique (tel qu'un disque dur) sera monté.

Partitions

Subdivisions d'un disque dur.

Réponses aux exercices guidés

1. Sur les systèmes Linux, où sont stockés les fichiers du chargeur de démarrage GRUB ?

   En-dessous de `/boot/grub`.

2. Où doit se terminer la partition de démarrage pour garantir qu'un PC sera toujours capable de charger le noyau ?

   Avant le cylindre 1024.

3. À quel endroit la partition EFI est-elle généralement montée ?

   En-dessous de `/boot/efi`.

4. Lors du montage manuel d'un système de fichiers, sous quel répertoire doit-il généralement être monté ?

   En-dessous de `/mnt`. Toutefois, ce n'est pas obligatoire. Vous pouvez monter une partition sous n'importe quel répertoire.

Réponses aux exercices d'approfondissement

1. Quelle est la plus petite unité à l'intérieur d'un groupe de volumes VG ?

   Les groupes de volumes sont subdivisés en extensions.

2. Comment définit-on la taille d'un volume logique LV ?

   Par la taille des extensions physiques multipliée par le nombre d'extensions sur le volume.

3. Sur un disque formaté selon le schéma de partitionnement MBR, quel est l'ID de la partition système EFI ?

   L'ID est `0xEF`.

4. En dehors des partitions d'échange, comment peut-on augmenter rapidement l'espace swap sur un système Linux ?

   Les fichiers swap peuvent être utilisés.

102.2 Installation d'un gestionnaire d'amorçage

Référence aux objectifs de LPI

LPIC-1 v5, Exam 101, Objective 102.2

Valeur

2

Domaines de connaissance les plus importants

- Démarrage sur des images d'amorçage alternatives et sauvegarde des options de démarrage.
- Installation et configuration d'un chargeur de démarrage tel que GRUB Legacy.
- Modifications élémentaires pour GRUB2.
- Interactions avec le chargeur d'amorçage.

Liste partielle de termes, fichiers et utilitaires utilisés pour cet objectif

- `menu.lst`, `grub.cfg` et `grub.conf`
- `grub-install`
- `grub-mkconfig`
- MBR

102.2 Leçon 1

Certification :	LPIC-1
Version :	5.0
Thème :	102 Installation de Linux et gestion des paquets
Objectif :	102.2 Installation d'un gestionnaire d'amorçage
Leçon :	1 sur 1

Introduction

Lorsqu'un ordinateur est mis sous tension, le premier logiciel à s'exécuter est le chargeur de démarrage. Il s'agit là d'un bout de code dont la seule mission est de charger le noyau d'un système d'exploitation pour lui céder le contrôle. Le noyau va charger les pilotes nécessaires, initialiser le matériel et ensuite charger le reste du système d'exploitation.

GRUB est le chargeur d'amorçage utilisé sur la plupart des distributions Linux. Il est capable de charger le noyau Linux ou d'autres systèmes d'exploitation, tels que Windows, et peut gérer plusieurs images et paramètres du noyau sous forme d'entrées de menu séparées. La sélection du noyau au démarrage se fait via une interface contrôlée par le clavier, et il existe une interface en ligne de commande pour éditer les options et les paramètres de démarrage.

La plupart des distributions Linux installent et configurent GRUB (en fait, GRUB 2) automatiquement, de sorte qu'un utilisateur lambda n'a pas besoin d'y réfléchir. Cependant, en tant qu'administrateur système, il est vital de savoir contrôler le processus d'amorçage afin de pouvoir récupérer le système après un problème d'amorçage suite à un échec de la mise à jour du noyau, par exemple.

Dans cette leçon, vous allez apprendre comment installer, configurer et interagir avec GRUB.

GRUB Legacy vs. GRUB 2

La version originale de GRUB (*Grand Unified Bootloader*) maintenant connue sous le nom de *GRUB Legacy* a été développée en 1995 dans le cadre du projet GNU Hurd et a ensuite été adoptée comme chargeur de démarrage par défaut de nombreuses distributions Linux, en remplacement d'alternatives antérieures telles que LILO.

GRUB 2 est une réécriture complète de GRUB qui se veut plus propre, plus sûre, plus robuste et plus puissante. Parmi les nombreux avantages par rapport à GRUB Legacy figurent un fichier de configuration beaucoup plus flexible (avec beaucoup plus de commandes et d'instructions conditionnelles, comme un langage de script), une conception plus modulaire et une meilleure localisation/internationalisation.

On trouve également la prise en charge de thèmes et de menus de démarrage graphiques avec écrans de démarrage, la possibilité de démarrer les ISO de LiveCD directement depuis le disque dur, une meilleure prise en charge des architectures non-x86, une prise en charge universelle des UUID (ce qui facilite l'identification des disques et des partitions) et bien plus encore.

GRUB Legacy n'est plus développé activement (la dernière version en date était la 0.97, en 2005), et aujourd'hui la plupart des distributions Linux majeures installent GRUB 2 comme chargeur de démarrage par défaut.

Où est le chargeur de démarrage ?

Historiquement, les disques durs des systèmes compatibles IBM PC étaient partitionnés selon le schéma de partitionnement MBR, créé en 1982 pour IBM PC-DOS (MS-DOS) 2.0.

Dans ce schéma, le premier secteur de 512 octets du disque est appelé le *Master Boot Record* et contient une table décrivant les partitions du disque (la table de partitions) ainsi qu'un code d'amorçage appelé "chargeur d'amorçage".

Lorsque l'ordinateur est allumé, ce code de démarrage très réduit (en raison des restrictions de taille) est chargé et exécuté, il passe le contrôle à un chargeur de démarrage secondaire sur le disque, généralement situé dans un espace de 32 ko entre le MBR et la première partition, qui à son tour va charger le ou les systèmes d'exploitation.

Sur un disque partitionné MBR, le code de démarrage de GRUB est installé sur le MBR. Celui-ci se charge et passe le contrôle à une image "core" installée entre le MBR et la première partition. À partir de là, GRUB est capable de charger le reste des ressources nécessaires (définitions de menu,

fichiers de configuration et modules supplémentaires) depuis le disque.

Cependant, le MBR comporte des restrictions quant au nombre de partitions (un maximum de 4 partitions primaires à l'origine, puis un maximum de 3 partitions primaires dont 1 partition étendue subdivisée en un certain nombre de partitions logiques) et une taille maximale de disque de 2 To. Pour pallier ces limitations, un nouveau schéma de partitionnement appelé GPT (*GUID Partition Table*), qui fait partie de la norme UEFI (*Unified Extensible Firmware Interface*), a été créé.

Les disques partitionnés GPT peuvent être utilisés soit avec des ordinateurs dotés du BIOS PC traditionnel, soit avec des ordinateurs équipés du micrologiciel UEFI. Sur les machines avec un BIOS, la deuxième partie de GRUB est stockée dans une partition d'amorçage BIOS spécifique.

Sur les systèmes avec un firmware UEFI, GRUB est chargé par le micrologiciel à partir des fichiers `grubia32.efi` (pour les systèmes 32 bits) ou `grubx64.efi` (pour les systèmes 64 bits) à partir d'une partition appelée ESP (*EFI System Partition*).

La partition `/boot`

Sous Linux, les fichiers nécessaires au processus de démarrage sont généralement stockés sur une partition de démarrage, montée sous le système de fichiers racine et communément appelée `/boot`.

Une partition de démarrage n'est pas nécessaire sur les systèmes actuels dans la mesure où les chargeurs de démarrage comme GRUB permettent généralement de monter le système de fichiers racine et de rechercher les fichiers nécessaires à l'intérieur d'un répertoire `/boot`, mais il s'agit là d'une bonne pratique qui permet de séparer les fichiers nécessaires au processus de démarrage du reste du système de fichiers.

Cette partition est généralement la première sur le disque. Cela tient au fait que le BIOS des PC IBM d'origine s'adressait aux disques en utilisant des cylindres, des têtes et des secteurs (CHS pour *Cylinder/Head/Sector*), avec un maximum de 1024 cylindres, 256 têtes et 63 secteurs, ce qui donne une taille maximale de disque de 528 Mo (504 Mo sous MS-DOS). Ce qui signifie que tout ce qui dépasse cette limite ne serait pas accessible, à moins qu'un autre système d'adressage de disque (comme LBA, *Logical Block Addressing*) ne soit utilisé.

Ainsi, pour une compatibilité maximale, la partition `/boot` est généralement située au début du disque et se termine avant le cylindre 1024 (528 Mo), ce qui garantit que la machine sera toujours capable de charger le noyau. La taille recommandée pour cette partition sur une machine actuelle est de 300 Mo.

D'autres raisons pour une partition `/boot` séparée sont le chiffrement et la compression, étant donné que certaines méthodes peuvent ne pas encore être supportées par GRUB 2, ou si vous avez

besoin de formater la partition racine du système (`/`) en utilisant un système de fichiers non supporté.

Contenu de la partition de démarrage

Le contenu de la partition `/boot` peut varier en fonction de l'architecture système ou du chargeur d'amorçage en vigueur, mais sur un système de type x86, vous trouverez généralement les fichiers ci-dessous. La plupart d'entre eux sont nommés avec un suffixe `-VERSION` qui représente la version du noyau Linux correspondant. Par conséquent, un fichier de configuration pour le noyau Linux en version `4.15.0-65-generic` serait appelé `config-4.15.0-65-generic`.

Fichier `config`

Ce fichier, généralement appelé `config-VERSION` (voir l'exemple ci-dessus), contient les paramètres de configuration du noyau Linux. Ce fichier est généré automatiquement lors de la compilation ou de l'installation d'un nouveau noyau et ne doit pas être modifié directement par l'utilisateur.

`System.map`

Ce fichier est une table de recherche qui fait correspondre les noms de symboles (comme les variables ou les fonctions) à leur position correspondante en mémoire. Il sert à déboguer un type de défaillance du système connue sous le nom de *kernel panic*, étant donné qu'il permet à l'utilisateur de savoir quelle variable ou fonction a été appelée lorsque la défaillance s'est produite. Comme le fichier `config`, le nom est généralement `System.map-VERSION` (par exemple `System.map-4.15.0-65-generic`).

Noyau Linux

C'est le noyau du système d'exploitation à proprement parler. Le nom est généralement `vmlinux-VERSION` (par exemple `vmlinux-4.15.0-65-generic`). Vous trouverez également le nom `vmlinuz` au lieu de `vmlinux`, le `z` final indiquant que le fichier a été compressé.

Disque mémoire initial

Il est généralement appelé `initrd.img-VERSION` et contient un système de fichiers racine minimal chargé dans un disque RAM, qui contient à son tour les outils et les modules du noyau nécessaires pour que le noyau puisse monter le système de fichiers racine proprement dit.

Fichiers liés au chargeur de démarrage

Sur les systèmes avec GRUB installé, ceux-ci sont généralement rangés dans `/boot/grub` et comprennent le fichier de configuration de GRUB (`/boot/grub/grub.cfg` pour GRUB 2 ou `/boot/grub/menu.lst` dans le cas de GRUB Legacy), les modules (dans `/boot/grub/i386-pc`), les fichiers de traduction (dans `/boot/grub/locale`) et les polices (dans

`/boot/grub/fonts`).

GRUB 2

Installation de GRUB 2

GRUB 2 peut être installé à l'aide de la commande `grub-install`. Si vous avez un système non amorçable, démarrez en utilisant un live CD ou un disque de secours, repérez la partition de démarrage de votre système, montez cette partition et exécutez la commande.

NOTE

Les commandes ci-dessous supposent que vous êtes connecté en tant que root. Si ce n'est pas le cas, commencez par invoquer `sudo su -` pour "devenir" root. Lorsque vous avez terminé, tapez `exit` pour vous déconnecter et redevenir un utilisateur normal.

Le premier disque d'un système est généralement le *périphérique de démarrage* et vous devrez éventuellement savoir s'il y a une *partition de démarrage* sur le disque. On peut le faire avec la commande `fdisk`. Pour lister toutes les partitions sur le premier disque de votre machine, utilisez :

```
# fdisk -l /dev/sda
Disk /dev/sda: 111,8 GiB, 120034123776 bytes, 234441648 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x97f8fef5

Device     Boot    Start       End   Sectors   Size Id Type
/dev/sda1  *        2048   2000895   1998848   976M 83 Linux
/dev/sda2        2002942 234440703 232437762 110,9G  5 Extended
/dev/sda5        2002944  18008063  16005120   7,6G 82 Linux swap / Solaris
/dev/sda6       18010112 234440703 216430592 103,2G 83 Linux
```

La partition de démarrage est identifiée par l'astérisque `*` sous la colonne `Boot`. Dans l'exemple ci-dessus, c'est `/dev/sda1`.

Maintenant, créez un répertoire temporaire sous `/mnt` et montez la partition en-dessous :

```
# mkdir /mnt/tmp
```

```
# mount /dev/sda1 /mnt/tmp
```

Ensuite, exécutez `grub-install` en le faisant pointer vers le *périphérique* de démarrage (*et non pas* la partition) ainsi que le répertoire où la partition de démarrage est montée. Si votre système dispose d'une partition de démarrage dédiée, la commande est :

```
# grub-install --boot-directory=/mnt/tmp /dev/sda
```

Si vous effectuez l'installation sur un système qui n'a pas de partition de démarrage, mais seulement un répertoire `/boot` sur le système de fichiers racine, pointez `grub-install` vers celui-ci. Dans ce cas, la commande est :

```
# grub-install --boot-directory=/boot /dev/sda
```

Configuration de GRUB 2

Le fichier de configuration par défaut pour GRUB 2 est `/boot/grub/grub.cfg`. Ce fichier est généré automatiquement et il n'est pas recommandé de le modifier à la main. Pour apporter des modifications à la configuration de GRUB, vous devez éditer le fichier `/etc/default/grub` puis lancer la commande `update-grub` pour générer un fichier valide.

NOTE `update-grub` est généralement un raccourci vers `grub-mkconfig -o /boot/grub/grub.cfg`, et les deux commandes produisent le même résultat.

Il y a une série d'options dans le fichier `/etc/default/grub` qui contrôlent le comportement de GRUB 2, comme le noyau par défaut au démarrage, le délai d'attente, les paramètres supplémentaires de la ligne de commande, etc. Les plus importantes sont :

`GRUB_DEFAULT=`

L'entrée de menu par défaut au démarrage. Il peut s'agir d'une valeur numérique (comme `0`, `1`, etc.), du nom d'une entrée de menu (comme `debian` ou `saved`), qui est utilisée en conjonction avec `GRUB_SAVEDEFAULT=`, voir l'explication ci-dessous. N'oubliez pas que les entrées de menu commencent à zéro, donc la première entrée de menu est `0`, la seconde est `1`, etc.

`GRUB_SAVEDEFAULT=`

Si cette option est définie à `true` et que `GRUB_DEFAULT=` est défini à `saved`, alors l'option de démarrage par défaut sera toujours la dernière sélectionnée dans le menu de démarrage.

`GRUB_TIMEOUT=`

Le délai en secondes avant que l'entrée de menu par défaut ne soit sélectionnée. S'il est défini à `0`, le système démarrera l'entrée par défaut sans afficher le menu. S'il est défini à `-1`, le système attendra que l'utilisateur choisisse une option, peu importe le temps que cela prendra.

`GRUB_CMDLINE_LINUX=`

Cette liste énumère les options en ligne de commande qui seront ajoutées aux entrées du noyau Linux.

`GRUB_CMDLINE_LINUX_DEFAULT=`

Par défaut, deux entrées de menu sont générées pour chaque noyau Linux, une avec les options par défaut et une entrée pour la récupération.

`GRUB_ENABLE_CRYPTODISK=`

Si la valeur est `y`, les commandes comme `grub-mkconfig`, `update-grub` et `grub-install` vont rechercher les disques chiffrés et ajouter les commandes nécessaires pour y accéder au démarrage.

Gérer les entrées de menu

Lorsque `update-grub` est exécuté, GRUB 2 va rechercher les noyaux et les systèmes d'exploitation sur la machine et générer les entrées de menu correspondantes dans le fichier `/boot/grub/grub.cfg`. De nouvelles entrées peuvent être ajoutées manuellement aux fichiers de script dans le répertoire `/etc/grub.`.

Ces fichiers doivent être exécutables, et ils sont traités en ordre numérique par `update-grub`. Ainsi, `05_debian_theme` est traité avant `10_linux` et ainsi de suite. Les entrées de menu personnalisées sont généralement ajoutées au fichier `40_custom`.

La syntaxe de base pour une entrée de menu est présentée ci-dessous :

```
menuentry "Default OS" {
    set root=(hd0,1)
    linux /vmlinuz root=/dev/sda1 ro quiet splash
    initrd /initrd.img
}
```

La première ligne commence toujours par `menuentry` et se termine par `}`. Le texte entre guillemets sera affiché comme titre d'entrée dans le menu de démarrage de GRUB 2.

Le paramètre `set root` définit le disque et la partition où se situe le système de fichiers racine du

système d'exploitation. Notez que dans GRUB 2 les disques sont numérotés à partir de zéro, donc `hd0` est le premier disque (`sda` sous Linux), `hd1` le second, et ainsi de suite. Quant aux partitions, elles sont numérotées à partir de un. Dans l'exemple ci-dessus, le système de fichiers racine est situé sur le premier disque (`hd0`), la première partition (`,1`), ou `sda1`.

Au lieu de spécifier directement le périphérique et la partition, vous pouvez également demander à GRUB 2 de rechercher un système de fichiers avec une étiquette ou un UUID (*Universally Unique Identifier*) spécifique. Pour ce faire, utilisez le paramètre `search --set=root` suivi du paramètre `--label` et de l'étiquette du système de fichiers à rechercher, ou `--fs-uuid` suivi de l'UUID du système de fichiers.

Vous pouvez trouver l'UUID d'un système de fichiers à l'aide de la commande ci-dessous :

```
$ ls -l /dev/disk/by-uuid/
total 0
lrwxrwxrwx 1 root root 10 nov  4 08:40 3e0b34e2-949c-43f2-90b0-25454ac1595d -> ../../sda5
lrwxrwxrwx 1 root root 10 nov  4 08:40 428e35ee-5ad5-4dcb-adca-539aba6c2d84 -> ../../sda6
lrwxrwxrwx 1 root root 10 nov  5 19:10 56C11DCC5D2E1334 -> ../../sdb1
lrwxrwxrwx 1 root root 10 nov  4 08:40 ae71b214-0aec-48e8-80b2-090b6986b625 -> ../../sda1
```

Dans l'exemple ci-dessus, l'UUID pour `/dev/sda1` est `ae71b214-0aec-48e8-80b2-090b6986b625`. Si vous souhaitez le définir comme périphérique racine pour GRUB 2, la commande sera `search --set=root --fs-uuid ae71b214-0aec-48e8-80b2-090b6986b625`.

Quand on utilise la commande `search`, il est courant d'ajouter le paramètre `--no-floppy` pour que GRUB ne perde pas de temps à chercher sur des disquettes floppy.

La ligne `linux` indique où se trouve le noyau du système d'exploitation (dans ce cas, le fichier `vmlinuz` à la racine du système de fichiers). Après cela, vous pouvez passer les paramètres en ligne de commande au noyau.

Dans l'exemple ci-dessus, nous avons spécifié la partition racine (`root=/dev/sda1`) en passant trois paramètres au noyau : la partition racine doit être montée en lecture seule (`ro`), la plupart des messages de journalisation doivent être désactivés (`quiet`) et un écran de démarrage doit être affiché (`splash`).

La ligne `initrd` indique où se trouve le disque mémoire initial. Dans l'exemple ci-dessus, le fichier est `initrd.img`, situé à la racine du système de fichiers.

NOTE La plupart des distributions Linux ne rangent pas vraiment le noyau et l'initrd dans le répertoire racine du système de fichiers racine. Au lieu de cela, il s'agit de liens vers les fichiers à proprement parler à l'intérieur du répertoire ou de la

> partition `/boot`.

La dernière ligne d'une entrée de menu ne doit contenir que le caractère `}`.

Interagir avec GRUB 2

Lorsque vous démarrez un système avec GRUB 2, vous verrez un menu d'options. Utilisez les touches fléchées pour sélectionner une option et `Entrée` pour confirmer et démarrer l'entrée sélectionnée.

TIP Si vous voyez juste un compte à rebours, mais pas de menu, appuyez sur `Maj` pour afficher le menu.

Pour éditer une option, sélectionnez-la avec les touches fléchées et appuyez sur `E`. Cela affichera une fenêtre d'édition avec le contenu `menuentry` associé à cette option, tel que défini dans `/boot/grub/grub.cfg`.

Après avoir édité une option, tapez `Ctrl` + `X` ou `F10` pour démarrer, ou `Échap` pour revenir au menu.

Pour accéder au shell GRUB 2, appuyez sur `C` dans l'écran de menu (ou `Ctrl` + `C`) dans la fenêtre d'édition). Vous verrez une invite de commande comme ceci : `grub >`

Tapez `help` pour voir une liste de toutes les commandes disponibles, ou appuyez sur `Échap` pour quitter le shell et revenir à l'écran de menu.

NOTE Rappelez-vous que ce menu n'apparaîtra pas si `GRUB_TIMEOUT` est paramétré à `0` dans `/etc/default/grub`.

Démarrer depuis le shell GRUB 2

Vous pouvez utiliser le shell GRUB 2 pour démarrer le système au cas où une mauvaise configuration dans une entrée de menu le ferait échouer.

La première chose à faire est de trouver l'emplacement de la partition de démarrage. Vous pouvez le faire avec la commande `ls`, qui vous affichera une liste des partitions et des disques que GRUB 2 a trouvés.

```
grub> ls
(proc) (hd0) (hd0,msdos1)
```

Dans l'exemple ci-dessus, les choses sont simples. Il n'y a qu'un seul disque `(hd0)` avec une seule partition : `(hd0,msdos1)`.

Les disques et partitions répertoriés varieront selon votre système. Dans notre exemple, la première partition de `hd0` est appelée `msdos1` parce que le disque a été partitionné en utilisant le schéma de partitionnement MBR. S'il était partitionné en GPT, le nom serait `gpt1`.

Pour démarrer Linux, nous avons besoin d'un noyau et d'un disque mémoire initial (initrd). Examinons le contenu de `(hd0,msdos1)` :

```
grub> ls (hd0,msdos1)/
lost+found/ swapfile etc/ media/ bin/ boot/ dev/ home/ lib/ lib64/ mnt/ opt/ proc/ root/
run/ sbin/ srv/ sys/ tmp/ usr/ var/ initrd.img initrd.img.old vmlinuz cdrom/
```

Vous pouvez ajouter le paramètre `-l` à `ls` pour obtenir un affichage détaillé, comparable à ce que vous obtiendriez dans un terminal Linux. Utilisez `Tab` pour compléter automatiquement les noms des disques, des partitions et des fichiers.

Notez que nous avons un noyau (`vmlinuz`) et des images initrd (`initrd.img`) directement dans le répertoire racine. Dans le cas contraire, nous pouvons vérifier le contenu de `/boot` avec `list (hd0,msdos1)/boot/`.

À présent, définissez la partition de démarrage :

```
grub> set root=(hd0,msdos1)
```

Chargez le noyau Linux avec la commande `linux`, suivie du chemin vers le noyau et de l'option `root=` pour indiquer au noyau où se trouve le système de fichiers racine du système d'exploitation.

```
grub> linux /vmlinuz root=/dev/sda1
```

Chargez le disque mémoire initial avec `initrd`, suivi du chemin complet vers le fichier `initrd.img` :

```
grub> initrd /initrd.img
```

Maintenant, démarrez le système avec `boot`.

Démarrer depuis le shell de secours

En cas de défaillance du démarrage, GRUB 2 peut charger un shell de secours, une version

simplifiée du shell que nous avons mentionné ci-dessus.

La procédure pour démarrer un système depuis ce shell est presque la même que celle illustrée précédemment. Cependant, vous devrez charger quelques modules de GRUB 2 pour que tout fonctionne.

Une fois que vous avez identifié la partition de démarrage (avec `ls`, comme indiqué précédemment), utilisez la commande `set prefix=`, suivie du chemin complet vers le répertoire contenant les fichiers de GRUB 2. Habituellement `/boot/grub`. Dans notre exemple :

```
grub rescue> set prefix=(hd0,msdos1)/boot/grub
```

A présent, chargez les modules `normal` et `linux` à l'aide de la commande `insmod` :

```
grub rescue> insmod normal
grub rescue> insmod linux
```

Ensuite, définissez la partition de démarrage avec `set root=` comme indiqué précédemment, chargez le noyau Linux (avec `linux`), le disque mémoire initial (`initrd`) et essayez de démarrer avec `boot`.

GRUB Legacy

Installer GRUB Legacy sur un système en état de fonctionnement

Pour installer GRUB Legacy sur un disque depuis un système en marche, nous utiliserons la commande `grub-install`. La commande de base est `grub-install PERIPHERIQUE` où `PERIPHERIQUE` est le disque sur lequel vous souhaitez installer GRUB Legacy. Un exemple serait `/dev/sda`.

```
# grub-install /dev/sda
```

Notez que vous devez spécifier le *périphérique* sur lequel GRUB Legacy sera installé, comme `/dev/sda/`, *et non pas la partition* comme dans `/dev/sda1`.

Par défaut, GRUB va copier les fichiers nécessaires dans le répertoire `/boot` sur le périphérique spécifié. Si vous souhaitez les copier vers un autre répertoire, utilisez le paramètre `--boot-directory=`, suivi du chemin complet vers l'endroit vers lequel les fichiers doivent être copiés.

Installer GRUB Legacy depuis un shell GRUB

Si vous ne pouvez pas démarrer le système pour une raison quelconque et que vous devez réinstaller GRUB Legacy, vous pouvez le faire à partir du shell GRUB sur un disque de démarrage GRUB Legacy.

Depuis le shell GRUB (tapez `c` dans le menu de démarrage pour accéder à l'invite `grub>`), la première étape consiste à définir le périphérique de démarrage, qui contient le répertoire `/boot`. Par exemple, si ce répertoire se trouve dans la première partition du premier disque, la commande sera :

```
grub> root (hd0,0)
```

Si vous ne savez pas quel périphérique contient le répertoire `/boot`, vous pouvez demander à GRUB de le rechercher avec la commande `find`, comme ci-dessous :

```
grub> find /boot/grub/stage1
 (hd0,0)
```

Ensuite, définissez la partition de démarrage comme indiqué ci-dessus et utilisez la commande `setup` pour installer GRUB Legacy dans le MBR et copier les fichiers nécessaires sur le disque :

```
grub> setup (hd0)
```

Une fois terminé, redémarrez le système et il devrait démarrer normalement.

Configurer les entrées de menu et les paramètres de GRUB Legacy

Les entrées de menu et les paramètres de GRUB Legacy sont stockés dans le fichier `/boot/grub/menu.lst`. Il s'agit d'un fichier texte simple avec une liste de commandes et de paramètres, qui peut être édité directement avec votre éditeur de texte préféré.

Les lignes commençant par `#` sont considérées comme des commentaires, et les lignes vides sont ignorées.

Une entrée de menu comporte au moins trois commandes. La première, `title`, définit le titre du système d'exploitation dans l'écran de menu. La seconde, `root`, indique à GRUB Legacy le périphérique ou la partition à partir de laquelle il doit démarrer.

La troisième entrée, `kernel`, spécifie le chemin complet vers l'image du noyau qui doit être

chargée lorsque l'entrée correspondante est sélectionnée.

Voici un exemple simple :

```
# Cette ligne est un commentaire
title Ma distribution Linux
root (hd0,0)
kernel /vmlinuz root=/dev/hda1
```

Contrairement à GRUB 2, dans GRUB Legacy, les disques *et* les partitions sont numérotés à partir de zéro. Ainsi, la commande `root (hd0,0)` va définir la partition de démarrage comme la première partition (`0`) du premier disque (`hd0`).

Vous pouvez omettre l'instruction `root` si vous spécifiez le périphérique de démarrage en tête du chemin à la commande `kernel`. La syntaxe est la même, donc :

```
kernel (hd0,0)/vmlinuz root=dev/hda1
```

équivaut à :

```
root (hd0,0)
kernel /vmlinuz root=/dev/hda1
```

Les deux vont charger le fichier `vmlinuz` à partir du répertoire racine (`/`) de la première partition du premier disque (`hd0,0`).

Le paramètre `root=/dev/hda1` après la commande `kernel` indique au noyau Linux quelle partition doit être utilisée comme système de fichiers racine. Il s'agit d'un paramètre du noyau Linux, et non d'une commande GRUB Legacy.

NOTE Pour en savoir plus sur les paramètres du noyau, voir https://www.kernel.org/doc/html/v4.14/admin-guide/kernel-parameters.html.

Vous devrez éventuellement spécifier l'emplacement du disque mémoire initial pour le système d'exploitation à l'aide du paramètre `initrd`. Le chemin complet vers le fichier peut être spécifié comme pour le paramètre `kernel`, et vous pouvez également spécifier un périphérique ou une partition avant le chemin, par exemple :

```
# Cette ligne est un commentaire
title Ma distribution Linux
```

```
root (hd0,0)
kernel /vmlinuz root=/dev/hda1
initrd /initrd.img
```

GRUB Legacy a une conception modulaire, où des modules (généralement stockés sous forme de fichiers `.mod` dans `/boot/grub/i386-pc`) peuvent être chargés pour ajouter des fonctionnalités supplémentaires, comme le support de matériel, de systèmes de fichiers ou de nouveaux algorithmes de compression inhabituels.

Les modules sont chargés en utilisant la commande `module`, suivie du chemin complet vers le fichier `.mod` correspondant. Gardez à l'esprit que, comme pour les noyaux et les images initrd, ce chemin est relatif au périphérique spécifié dans la directive `root`.

L'exemple ci-dessous va charger le module `915resolution`, requis pour régler correctement la résolution du framebuffer sur les systèmes équipés de chipsets vidéo Intel de la série 800 ou 900.

```
module /boot/grub/i386-pc/915resolution.mod
```

Multi-amorçage avec d'autres systèmes d'exploitation

GRUB Legacy peut être utilisé pour charger des systèmes d'exploitation non pris en charge, comme Windows, en utilisant un processus appelé *chainloading* (multi-amorçage). GRUB Legacy est chargé en premier, et lorsque l'option correspondante est sélectionnée, le chargeur de démarrage du système en question est chargé.

Une entrée typique pour un double boot Windows ressemblerait à ceci :

```
# Charger Windows
title Windows XP
root (hd0,1)
makeactive
chainload +1
boot
```

Passons en revue chaque paramètre. Comme précédemment, `root (hd0,1)` spécifie le périphérique et la partition où se trouve le chargeur de démarrage du système d'exploitation que nous souhaitons charger. Dans cet exemple, la *seconde* partition du premier disque.

`makeactive`

définit un fanion indiquant qu'il s'agit d'une partition active. Cela ne fonctionne que sur les

partitions primaires DOS.

`chainload +1`

indique à GRUB de charger le premier secteur de la partition de démarrage. C'est là que se trouvent généralement les chargeurs de démarrage.

`boot`

va exécuter le chargeur de démarrage et charger le système d'exploitation correspondant.

Exercices guidés

1. Quel est l'emplacement par défaut du fichier de configuration de GRUB 2 ?

2. Quelles sont les étapes nécessaires pour modifier les réglages de GRUB 2 ?

3. Dans quel fichier doit-on ajouter les entrées personnalisées du menu GRUB 2 ?

4. Où sont stockées les entrées du menu de GRUB Legacy ?

5. À partir d'un menu GRUB 2 ou GRUB Legacy, comment pouvez-vous accéder au shell GRUB ?

Exercices d'approfondissement

1. Supposons qu'un utilisateur configure GRUB Legacy pour démarrer à partir de la deuxième partition du premier disque. Il crée l'entrée de menu personnalisée suivante :

   ```
   title Ma Distrib Linux
   root (hd0,2)
   kernel /vmlinuz root=/dev/hda1
   initrd /initrd.img
   ```

   Pourtant, le système ne démarre pas. Qu'est-ce qui ne va pas ?

2. Supposons que vous ayez un disque identifié en tant que `/dev/sda` avec plusieurs partitions. Quelle commande peut être utilisée pour déterminer la partition de démarrage d'un système ?

3. Quelle commande peut être utilisée pour déterminer l'UUID d'une partition ?

4. Considérez l'entrée suivante pour GRUB 2 :

   ```
   menuentry "OS principal" {
       set root=(hd0,1)
       linux /vmlinuz root=/dev/sda1 ro quiet splash
       initrd /initrd.img
   }
   ```

   Modifiez-la pour que le système démarre à partir d'un disque avec l'UUID `5dda0af3-c995-481a-a6f3-46dcd3b6998d`.

5. Comment pouvez-vous paramétrer GRUB 2 pour qu'il attende 10 secondes avant de démarrer l'entrée de menu par défaut ?

6. Depuis un shell GRUB Legacy, quelles sont les commandes pour installer GRUB sur la première partition du second disque ?

Résumé

Dans cette leçon, nous avons appris :

- Qu'est-ce qu'un chargeur de démarrage.
- Les différences entre GRUB Legacy et GRUB 2.
- Qu'est-ce qu'une partition de démarrage, et qu'est-ce qu'elle contient.
- Comment installer GRUB Legacy et GRUB 2.
- Comment configurer GRUB Legacy et GRUB 2.
- Comment ajouter des entrées de menu personnalisées à GRUB Legacy et GRUB 2.
- Comment interagir avec l'écran de menu et la console de GRUB Legacy et GRUB 2.
- Comment démarrer un système à partir du shell GRUB Legacy ou GRUB 2 ou du shell de secours.

Les commandes suivantes ont été abordées dans cette leçon :

- `grub-install`
- `update-grub`
- `grub-mkconfig`

Réponses aux exercices guidés

1. Quel est l'emplacement par défaut du fichier de configuration de GRUB 2 ?

   `/boot/grub/grub.cfg`

2. Quelles sont les étapes nécessaires pour modifier les réglages de GRUB 2 ?

   Apportez vos modifications au fichier `/etc/default/grub`, puis actualisez la configuration avec `update-grub`.

3. Dans quel fichier doit-on ajouter les entrées personnalisées du menu GRUB 2 ?

   `/etc/grub.d/40_custom`

4. Où sont stockées les entrées du menu de GRUB Legacy ?

   `/boot/grub/menu.lst`

5. À partir d'un menu GRUB 2 ou GRUB Legacy, comment pouvez-vous accéder au shell GRUB ?

   Appuyez sur `c` dans l'écran de menu.

Réponses aux exercices d'approfondissement

1. Supposons qu'un utilisateur configure GRUB Legacy pour démarrer à partir de la deuxième partition du premier disque. Il crée l'entrée de menu personnalisée suivante :

   ```
   title Ma Distrib Linux
   root (hd0,2)
   kernel /vmlinuz root=/dev/hda1
   initrd /initrd.img
   ```

   Pourtant, le système ne démarre pas. Qu'est-ce qui ne va pas ?

   La partition de démarrage est mal renseignée. Rappelez-vous que, contrairement à GRUB 2, GRUB Legacy compte les partitions à partir de *zéro*. Ainsi, la bonne directive pour la deuxième partition du premier disque devrait être `root (hd0,1)`.

2. Supposons que vous ayez un disque identifié en tant que `/dev/sda` avec plusieurs partitions. Quelle commande peut être utilisée pour déterminer la partition de démarrage d'un système ?

   Invoquez `fdisk -l /dev/sda`. La partition de démarrage sera marquée d'un astérisque (`*`) dans le listing.

3. Quelle commande peut être utilisée pour déterminer l'UUID d'une partition ?

   Utilisez `ls -la /dev/disk/by-uuid/` et repérez l'UUID qui pointe vers la partition.

4. Considérez l'entrée suivante pour GRUB 2 :

   ```
   menuentry "OS principal" {
       set root=(hd0,1)
       linux /vmlinuz root=/dev/sda1 ro quiet splash
       initrd /initrd.img
   }
   ```

   Modifiez-la pour que le système démarre à partir d'un disque avec l'UUID `5dda0af3-c995-481a-a6f3-46dcd3b6998d`.

   Vous devrez modifier l'instruction `set root`. Au lieu de spécifier un disque et une partition, dites à GRUB de rechercher la partition avec l'UUID de votre choix.

   ```
   menuentry "OS principal" {
   ```

```
    search --set=root --fs-uuid 5dda0af3-c995-481a-a6f3-46dcd3b6998d
    linux /vmlinuz root=/dev/sda1 ro quiet splash
    initrd /initrd.img
}
```

5. Comment pouvez-vous paramétrer GRUB 2 pour qu'il attende 10 secondes avant de démarrer l'entrée de menu par défaut ?

   Ajoutez le paramètre `GRUB_TIMEOUT=10` à `/etc/default/grub`.

6. Depuis un shell GRUB Legacy, quelles sont les commandes pour installer GRUB sur la première partition du second disque ?

```
grub> root (hd1,0)
grub> setup (hd1)
```

102.3 Gestion des bibliothèques partagées

Référence aux objectifs de LPI

LPIC-1 v5, Exam 101, Objective 102.3

Valeur

1

Domaines de connaissance les plus importants

- Identification des bibliothèques partagées.
- Identification des emplacements typiques des bibliothèques systèmes.
- Chargement des bibliothèques partagées.

Liste partielle de termes, fichiers et utilitaires utilisés pour cet objectif

- `ldd`
- `ldconfig`
- `/etc/ld.so.conf`
- `LD_LIBRARY_PATH`

102.3 Leçon 1

Certification:	LPIC-1
Version :	5.0
Thème :	102 Installer Linux et gérer les paquets
Objectif :	102.3 Gérer les bibliothèques partagées
Leçon :	1 sur 1

Introduction

Dans cette leçon, nous allons parler des *bibliothèques partagées*, également connues sous le nom d'objets partagés (*shared objects*) : des bouts de code compilés et réutilisables comme des fonctions ou des classes, et qui sont utilisés de manière récurrente par différents programmes.

Pour commencer, nous allons expliquer ce que sont les bibliothèques partagées, comment les identifier et où les trouver. Ensuite, nous allons voir comment configurer leurs emplacements de stockage. Enfin, nous allons montrer comment rechercher les bibliothèques partagées dont dépend un programme particulier.

Le concept des bibliothèques partagées

Tout comme leurs équivalents matériels, les bibliothèques logicielles sont des collections de code destinées à être utilisées par une multitude de programmes différents ; tout comme les bibliothèques physiques conservent des livres et d'autres ressources qui peuvent être utilisés par une foule de gens différents.

Pour construire un fichier exécutable à partir du code source d'un programme, deux étapes

importantes sont nécessaires. Pour commencer, le *compilateur* transforme le code source en code machine qui est stocké dans ce qu'on appelle des *fichiers objets*. Ensuite, l'éditeur de liens (*linker*) combine les fichiers objets et les *lie* aux bibliothèques afin de générer le fichier exécutable final. Cette édition des liens peut se faire de manière *statique* ou *dynamique*. Selon la méthode choisie, nous parlerons de bibliothèques statiques ou, dans le cas d'une édition de liens dynamique, de bibliothèques partagées. Expliquons leurs différences.

Bibliothèques statiques

Une bibliothèque statique est intégrée au programme au moment de la création du lien. Une copie du code de la bibliothèque est embarquée dans le programme pour en faire partie. Ainsi, le programme ne dépend pas de la bibliothèque au moment de l'exécution car il contient déjà le code de la bibliothèque. L'absence de dépendances peut être considérée comme un avantage, étant donné que vous n'avez pas à vous soucier de la disponibilité des bibliothèques utilisées. L'inconvénient, c'est que les programmes liés statiquement sont plus lourds.

Bibliothèques partagées (ou dynamiques)

Dans le cas des bibliothèques partagées, l'éditeur de liens veille simplement à ce que le programme référence correctement les bibliothèques. En revanche, l'éditeur de liens ne copie aucun code de la bibliothèque dans le fichier du programme. Au moment de l'exécution, la bibliothèque partagée doit cependant être disponible pour satisfaire les dépendances du programme. C'est donc une approche économique de la gestion des ressources du système qui permet de réduire la taille des fichiers des programmes en ne chargeant qu'une seule copie de la bibliothèque en mémoire, même si elle est utilisée par plusieurs programmes.

Conventions de nommage des fichiers objets partagés

Le nom d'une bibliothèque partagée (*soname*) respecte un schéma composé de trois éléments :

- Nom de la bibliothèque (normalement préfixé par `lib`)
- `so` (qui signifie "shared object")
- Numéro de version de la bibliothèque

Voici un exemple : `libpthread.so.0`

En comparaison, les noms des bibliothèques statiques se terminent en `.a`, par exemple `libpthread.a`.

NOTE

Comme les fichiers contenant les bibliothèques partagées doivent être disponibles lorsque le programme est exécuté, la plupart des systèmes Linux contiennent des bibliothèques partagées. Puisque les bibliothèques statiques ne sont requises dans

> un fichier dédié que lorsqu'un programme est lié, elles peuvent ne pas être présentes sur un système d'utilisateur final.

La `glibc` (*GNU C library*) est un bon exemple de bibliothèque partagée. Sur un système Debian GNU/Linux 9.9, son fichier est nommé `libc.so.6`. Ces noms de fichiers plutôt génériques sont normalement des liens symboliques qui pointent vers le fichier réel contenant une bibliothèque, dont le nom contient le numéro de version exact. Dans le cas de la `glibc`, ce lien symbolique ressemble à ceci :

```
$ ls -l /lib/x86_64-linux-gnu/libc.so.6
lrwxrwxrwx 1 root root 12 feb  6 22:17 /lib/x86_64-linux-gnu/libc.so.6 -> libc-2.24.so
```

Cette façon de référencer les fichiers de bibliothèques partagées nommés selon une version spécifique par des noms de fichiers plus généraux constitue une pratique courante.

D'autres exemples de bibliothèques partagées comprennent `libreadline` (qui permet aux utilisateurs de modifier les lignes de commande au fur et à mesure qu'elles sont tapées et inclut le support des modes d'édition Emacs et vi), `libcrypt` (qui contient des fonctions liées au chiffrement, au hachage et à l'encodage), ou `libcurl` (qui est une bibliothèque de transfert de fichiers multiprotocole).

Voici les emplacements habituels des bibliothèques partagées dans un système Linux :

- `/lib`
- `/lib32`
- `/lib64`
- `/usr/lib`
- `/usr/local/lib`

NOTE Le concept des bibliothèques partagées n'est pas spécifique à Linux. Sous Windows, par exemple, elles sont appelées DLL, ce qui signifie *Dynamic Link Library* (bibliothèque de liens dynamiques).

Configuration des chemins de bibliothèques partagées

Les références contenues dans les programmes liés dynamiquement sont résolues par le chargeur de liens dynamiques (`ld.so` ou `ld-linux.so`) lorsque le programme est exécuté. Le chargeur de liens dynamiques recherche les bibliothèques dans une série de répertoires. Ces répertoires sont spécifiés par le *chemin des bibliothèques*. Le chemin des bibliothèques est configuré dans le

répertoire `/etc`, à savoir dans le fichier `/etc/ld.so.conf` et, plus couramment de nos jours, dans les fichiers qui se trouvent dans le répertoire `/etc/ld.so.conf.d`. Normalement, le premier ne comprend qu'une seule ligne `include` pour les fichiers `*.conf` dans le second :

```
$ cat /etc/ld.so.conf
include /etc/ld.so.conf.d/*.conf
```

Le répertoire `/etc/ld.so.conf.d` contient des fichiers `*.conf` :

```
$ ls /etc/ld.so.conf.d/
libc.conf  x86_64-linux-gnu.conf
```

Ces fichiers `*.conf` doivent inclure les chemins absolus vers les répertoires des bibliothèques partagées :

```
$ cat /etc/ld.so.conf.d/x86_64-linux-gnu.conf
# Multiarch support
/lib/x86_64-linux-gnu
/usr/lib/x86_64-linux-gnu
```

La commande `ldconfig` se charge de lire ces fichiers de configuration, de créer l'ensemble des liens symboliques ci-dessus qui aident à localiser les différentes bibliothèques et enfin de mettre à jour le fichier cache `/etc/ld.so.cache`. Ainsi, `ldconfig` doit être exécuté chaque fois que des fichiers de configuration sont ajoutés ou mis à jour.

Voici quelques options utiles pour `ldconfig` :

`-v`, `--verbose`

Afficher les numéros de version des bibliothèques, le nom de chaque répertoire et les liens qui sont créés :

```
$ sudo ldconfig -v
/usr/local/lib:
/lib/x86_64-linux-gnu:
    libnss_myhostname.so.2 -> libnss_myhostname.so.2
    libfuse.so.2 -> libfuse.so.2.9.7
    libidn.so.11 -> libidn.so.11.6.16
    libnss_mdns4.so.2 -> libnss_mdns4.so.2
    libparted.so.2 -> libparted.so.2.0.1
```

```
    (...)
```

Nous voyons donc, par exemple, comment `libfuse.so.2` est lié au fichier d'objets partagés effectif `libfuse.so.2.9.7`.

`-p`, `--print-cache`

Afficher les listes de répertoires et de bibliothèques candidates stockées dans le cache actuel :

```
$ sudo ldconfig -p
1094 libs found in the cache `/etc/ld.so.cache'
    libzvbi.so.0 (libc6,x86-64) => /usr/lib/x86_64-linux-gnu/libzvbi.so.0
    libzvbi-chains.so.0 (libc6,x86-64) => /usr/lib/x86_64-linux-gnu/libzvbi-chains.so.0
    libzmq.so.5 (libc6,x86-64) => /usr/lib/x86_64-linux-gnu/libzmq.so.5
    libzeitgeist-2.0.so.0 (libc6,x86-64) => /usr/lib/x86_64-linux-gnu/libzeitgeist-
2.0.so.0
    (...)
```

Notez comment le cache utilise le *soname* pleinement qualifié des liens :

```
$ sudo ldconfig -p |grep libfuse
      libfuse.so.2 (libc6,x86-64) => /lib/x86_64-linux-gnu/libfuse.so.2
```

Si nous faisons un listing détaillé de `/lib/x86_64-linux-gnu/libfuse.so.2`, nous voyons la référence au fichier d'objet partagé réel `libfuse.so.2.9.7` stocké dans le même répertoire :

```
$ ls -l /lib/x86_64-linux-gnu/libfuse.so.2
lrwxrwxrwx 1 root root 16 Aug 21  2018 /lib/x86_64-linux-gnu/libfuse.so.2 ->
libfuse.so.2.9.7
```

NOTE

Puisqu'il faut un accès en écriture au cache `/etc/ld.so.cache` (appartenant à root), vous devez soit être root, soit utiliser `sudo` pour invoquer `ldconfig`. Pour plus d'informations sur la commande `ldconfig`, reportez-vous à sa page de manuel.

En complément des fichiers de configuration décrits ci-dessus, la variable d'environnement `LD_LIBRARY_PATH` peut être utilisée pour ajouter temporairement de nouveaux chemins pour les bibliothèques partagées. Elle est constituée d'un ensemble de répertoires séparés par deux points (`:`) où les bibliothèques sont recherchées. Par exemple, pour ajouter `/usr/local/mylib` au chemin des bibliothèques dans la session shell courante, vous pouvez taper :

```
$ LD_LIBRARY_PATH=/usr/local/mylib
```

Vous pouvez maintenant vérifier sa valeur :

```
$ echo $LD_LIBRARY_PATH
/usr/local/mylib
```

Pour ajouter `/usr/local/mylib` au chemin des bibliothèques dans la session shell actuelle en l'exportant vers tous les processus enfants créés à partir de ce shell, vous devez taper :

```
$ export LD_LIBRARY_PATH=/usr/local/mylib
```

Pour supprimer la variable d'environnement `LD_LIBRARY_PATH`, il suffit de taper :

```
$ unset LD_LIBRARY_PATH
```

Pour rendre les changements permanents, vous pouvez écrire la ligne

```
export LD_LIBRARY_PATH=/usr/local/mylib
```

dans un des scripts d'initialisation de Bash tels que `/etc/bash.bashrc` ou `~/.bashrc`.

NOTE `LD_LIBRARY_PATH` est aux bibliothèques partagées ce que `PATH` est aux exécutables. Pour plus d'informations sur les variables d'environnement et la configuration du shell, reportez-vous aux leçons correspondantes.

Chercher les dépendances d'un exécutable donné

Pour rechercher les bibliothèques partagées requises par un programme spécifique, utilisez la commande `ldd` suivie du chemin absolu vers le programme. Le résultat indique le chemin du fichier de la bibliothèque partagée ainsi que l'adresse mémoire hexadécimale à laquelle il est chargé :

```
$ ldd /usr/bin/git
    linux-vdso.so.1 =>  (0x00007ffcbb310000)
    libpcre.so.3 => /lib/x86_64-linux-gnu/libpcre.so.3 (0x00007f18241eb000)
    libz.so.1 => /lib/x86_64-linux-gnu/libz.so.1 (0x00007f1823fd1000)
    libresolv.so.2 => /lib/x86_64-linux-gnu/libresolv.so.2 (0x00007f1823db6000)
```

```
    libpthread.so.0 => /lib/x86_64-linux-gnu/libpthread.so.0 (0x00007f1823b99000)
    librt.so.1 => /lib/x86_64-linux-gnu/librt.so.1 (0x00007f1823991000)
    libc.so.6 => /lib/x86_64-linux-gnu/libc.so.6 (0x00007f18235c7000)
    /lib64/ld-linux-x86-64.so.2 (0x00007f182445b000)
```

De même, nous utilisons `ldd` pour rechercher les dépendances d'un objet partagé :

```
$ ldd /lib/x86_64-linux-gnu/libc.so.6
    /lib64/ld-linux-x86-64.so.2 (0x00007fbfed578000)
    linux-vdso.so.1 (0x00007fffb7bf5000)
```

Avec l'option `-u` (ou `--unused`), `ldd` affiche les dépendances directes inutilisées (si elles existent) :

```
$ ldd -u /usr/bin/git
Unused direct dependencies:
    /lib/x86_64-linux-gnu/libz.so.1
    /lib/x86_64-linux-gnu/libpthread.so.0
    /lib/x86_64-linux-gnu/librt.so.1
```

La présence de dépendances inutilisées est liée aux options utilisées par l'éditeur de liens lors de la construction du binaire. Même si le programme n'a pas besoin d'une bibliothèque inutilisée, il a quand même été lié et étiqueté comme `NEEDED` (requis) dans les informations sur le fichier objet. Vous pouvez explorer ce sujet en utilisant des commandes comme `readelf` ou `objdump`, que vous utiliserez bientôt dans l'exercice d'approfondissement.

Exercices guidés

1. Séparez les noms des bibliothèques partagées suivantes en leurs composantes :

Nom complet du fichier	**Nom de la bibliothèque**	**suffixe *so***	**Numéro de version**
`linux-vdso.so.1`			
`libprocps.so.6`			
`libdl.so.2`			
`libc.so.6`			
`libsystemd.so.0`			
`ld-linux-x86-64.so.2`			

2. Vous avez développé un logiciel et vous souhaitez ajouter un nouveau répertoire de bibliothèques partagées à votre système (`/opt/lib/mylib`). Vous écrivez son chemin absolu dans un fichier appelé `mylib.conf`.
   - Dans quel répertoire devez-vous ranger ce fichier ?
   - Quelle commande devez-vous exécuter pour que les changements soient pleinement pris en compte ?

3. Quelle commande utiliseriez-vous pour recenser les bibliothèques partagées requises par `kill` ?

Exercices d'approfondissement

1. `objdump` est un outil en ligne de commande qui affiche les informations relatives aux fichiers objets. Vérifiez s'il est installé sur votre système avec `which objdump`. Si ce n'est pas le cas, veuillez l'installer.
   - Utilisez `objdump` avec l'option `p` (ou `--private-headers`) et `grep` pour afficher les dépendances de `glibc` :
   - Utilisez `objdump` avec l'option `p` (ou `--private-headers`) et `grep` pour afficher le *soname* de `glibc` :
   - Utilisez `objdump` avec l'option `p` (ou `--private-headers`) et `grep` pour afficher les dépendances de Bash :

Résumé

Dans cette leçon, nous avons appris :

- Qu'est-ce qu'une bibliothèque partagée (ou dynamique).
- Les différences entre les bibliothèques partagées et les bibliothèques statiques.
- Les noms des bibliothèques partagées (*sonames*).
- Les emplacements préférés pour les bibliothèques partagées dans un système Linux comme `/lib` ou `/usr/lib`.
- La fonction de l'éditeur de liens dynamiques `ld.so` (ou `ld-linux.so`).
- Comment configurer les chemins des bibliothèques partagées par le biais de fichiers dans `/etc/` comme `ld.so.conf` ou ceux dans le répertoire `ld.so.conf.d`.
- Comment configurer les chemins des bibliothèques partagées au moyen de la variable d'environnement `LD_LIBRARY_PATH`.
- Comment rechercher les dépendances des exécutables et des bibliothèques partagées.

Les commandes suivantes ont été abordées dans cette leçon :

`ls`
: Afficher le contenu d'un répertoire.

`cat`
: Concaténer des fichiers et afficher sur la sortie standard.

`sudo`
: Faire exécuter une commande par le super-utilisateur avec les privilèges de l'administrateur.

`ldconfig`
: Configurer les dépendances d'exécution de l'éditeur de liens.

`echo`
: Afficher la valeur d'une variable d'environnement.

`export`
: Exporter la valeur d'une variable d'environnement vers les shells enfants.

`unset`
: Supprimer une variable d'environnement.

`ldd`

Afficher les dépendances en objets partagés d'un programme.

`readelf`

Afficher les informations sur les fichiers ELF (ELF signifie *executable and linkable format*).

`objdump`

Afficher les informations des fichiers objets.

Réponses aux exercices guidés

1. Séparez les noms des bibliothèques partagées suivantes en leurs composantes :

Nom complet du fichier	**Nom de la bibliothèque**	**suffixe *so***	**Numéro de version**
`linux-vdso.so.1`	`linux-vdso`	`so`	`1`
`libprocps.so.6`	`libprocps`	`so`	`6`
`libdl.so.2`	`libdl`	`so`	`2`
`libc.so.6`	`libc`	`so`	`6`
`libsystemd.so.0`	`libsystemd`	`so`	`0`
`ld-linux-x86-64.so.2`	`ld-linux-x86-64`	`so`	`2`

2. Vous avez développé un logiciel et vous souhaitez ajouter un nouveau répertoire de bibliothèques partagées à votre système (`/opt/lib/mylib`). Vous écrivez son chemin absolu dans un fichier appelé `mylib.conf`.
   - Dans quel répertoire devez-vous ranger ce fichier ?

     `/etc/ld.so.conf.d`

   - Quelle commande devez-vous exécuter pour que les changements soient pleinement pris en compte ?

     `ldconfig`

3. Quelle commande utiliseriez-vous pour recenser les bibliothèques partagées requises par `kill` ?

   `ldd /bin/kill`

Réponses aux exercices d'approfondissement

1. `objdump` est un outil en ligne de commande qui affiche les informations relatives aux fichiers objets. Vérifiez s'il est installé sur votre système avec `which objdump`. Si ce n'est pas le cas, veuillez l'installer.
   - Utilisez `objdump` avec l'option `p` (ou `--private-headers`) et `grep` pour afficher les dépendances de `glibc` :

     ```
     objdump -p /lib/x86_64-linux-gnu/libc.so.6 | grep NEEDED
     ```

   - Utilisez `objdump` avec l'option `p` (ou `--private-headers`) et `grep` pour afficher le *soname* de `glibc` :

     ```
     objdump -p /lib/x86_64-linux-gnu/libc.so.6 | grep SONAME
     ```

   - Utilisez `objdump` avec l'option `p` (ou `--private-headers`) et `grep` pour afficher les dépendances de Bash :

     ```
     objdump -p /bin/bash | grep NEEDED
     ```

102.4 Utilisation du gestionnaire de paquetage Debian

Référence aux objectifs de LPI

LPIC-1 v5, Exam 101, Objective 102.4

Valeur

3

Domaines de connaissance les plus importants

- Installation, mise à jour et désinstallation des paquetages binaires Debian.
- Recherche des paquetages contenant des fichiers ou des bibliothèques spécifiques installés ou non.
- Obtention d'informations sur un paquetage Debian comme la version, le contenu, les dépendances, l'intégrité du paquetage, et l'état d'installation (que le paquetage soit installé ou non).
- Connaissance de base de apt.

Liste partielle de termes, fichiers et utilitaires utilisés pour cet objectif

- `/etc/apt/sources.list`
- `dpkg`
- `dpkg-reconfigure`
- `apt-get`
- `apt-cache`

102.4 Leçon 1

Certification :	LPIC-1
Version :	5.0
Thème :	102 Installer Linux et gérer les paquets
Objectif :	102.4 Gérer les paquets sous Debian
Leçon :	1 sur 1

Introduction

Il y a longtemps, lorsque Linux en était encore à ses débuts, le moyen le plus courant de distribuer un logiciel était sous forme d'un fichier compressé (généralement une archive `.tar.gz`) avec le code source, qu'il fallait décompresser et compiler soi-même.

Cependant, à mesure que la quantité et la complexité des logiciels augmentaient, la nécessité de trouver un moyen de distribuer des logiciels pré-compilés s'imposait. En effet, tout le monde n'avait pas les ressources, en termes de temps et de puissance de calcul, pour compiler de gros projets comme le noyau Linux ou un serveur X.

Petit à petit, les efforts pour normaliser un mode de distribution de ces "paquets" logiciels se sont multipliés, et les premiers gestionnaires de paquets ont vu le jour. Ces outils ont considérablement facilité l'installation, la configuration ou la suppression de logiciels sur un système.

L'un d'entre eux était le format de paquet Debian (`.deb`) et son outil de paquets (`dpkg`). Aujourd'hui, ils sont largement utilisés non seulement sur Debian lui-même, mais aussi sur ses dérivés, comme Ubuntu et ses variantes.

Un autre outil de gestion des paquets populaire sur les systèmes basés sur Debian est le *Advanced*

Package Tool (`apt`), qui peut simplifier de nombreux aspects de l'installation, de la maintenance et de la suppression des paquets, ce qui rend la tâche beaucoup plus facile.

Dans cette leçon, nous allons apprendre comment utiliser aussi bien `dpkg` que `apt` pour obtenir, installer, maintenir et supprimer des logiciels sur un système Linux basé sur Debian.

L'outil Debian Package (dpkg)

L'outil *Debian Package* (`dpkg`) est l'utilitaire de base pour installer, configurer, maintenir et supprimer des paquets logiciels sur les systèmes basés sur Debian. L'opération la plus simple consiste à installer un paquet `.deb`, ce qui peut être fait avec :

```
# dpkg -i PACKAGENAME
```

Où `NOMDUPAQUET` est le nom du fichier `.deb` que vous voulez installer.

Les mises à jour des paquets sont traitées de la même manière. Avant d'installer un paquet, `dpkg` va vérifier si une version précédente existe déjà sur le système. Si c'est le cas, le paquet sera mis à niveau vers la nouvelle version. Autrement, une nouvelle copie sera installée.

Gérer les dépendances

La plupart du temps, un paquet pourra dépendre d'autres paquets pour fonctionner comme prévu. Par exemple, un éditeur d'images pourra avoir besoin de bibliothèques pour ouvrir des fichiers JPEG, ou un autre programme pourra avoir besoin d'un *widget toolkit* comme Qt ou GTK pour son interface utilisateur.

`dpkg` va vérifier si ces dépendances sont installées sur votre système, et il ne pourra pas installer le paquet si elles ne sont pas présentes. Dans ce cas, `dpkg` affichera la liste des paquets manquants. Cependant, il ne peut pas *résoudre* les dépendances par lui-même.

Dans l'exemple ci-dessous, l'utilisateur essaie d'installer le paquet de l'éditeur vidéo OpenShot, mais certaines dépendances sont absentes :

```
# dpkg -i openshot-qt_2.4.3+dfsg1-1_all.deb
(Reading database ... 269630 files and directories currently installed.)
Preparing to unpack openshot-qt_2.4.3+dfsg1-1_all.deb ...
Unpacking openshot-qt (2.4.3+dfsg1-1) over (2.4.3+dfsg1-1) ...
dpkg: dependency problems prevent configuration of openshot-qt:
 openshot-qt depends on fonts-cantarell; however:
  Package fonts-cantarell is not installed.
```

```
 openshot-qt depends on python3-openshot; however:
  Package python3-openshot is not installed.
 openshot-qt depends on python3-pyqt5; however:
  Package python3-pyqt5 is not installed.
 openshot-qt depends on python3-pyqt5.qtsvg; however:
  Package python3-pyqt5.qtsvg is not installed.
 openshot-qt depends on python3-pyqt5.qtwebkit; however:
  Package python3-pyqt5.qtwebkit is not installed.
 openshot-qt depends on python3-zmq; however:
  Package python3-zmq is not installed.

dpkg: error processing package openshot-qt (--install):
 dependency problems - leaving unconfigured
Processing triggers for mime-support (3.60ubuntu1) ...
Processing triggers for gnome-menus (3.32.0-1ubuntu1) ...
Processing triggers for desktop-file-utils (0.23-4ubuntu1) ...
Processing triggers for hicolor-icon-theme (0.17-2) ...
Processing triggers for man-db (2.8.5-2) ...
Errors were encountered while processing:
 openshot-qt
```

Comme indiqué ci-dessus, OpenShot dépend des paquets `fonts-cantarell`, `python3-openshot`, `python3-pyqt5`, `python3-pyqt5.qtsvg`, `python3-pyqt5.qtwebkit` et `python3-zmq`. Tous ces composants doivent être installés avant que l'installation d'OpenShot puisse réussir.

Supprimer des paquets

Pour supprimer un paquet, passez le paramètre `-r` à `dpkg`, suivi du nom du paquet. Par exemple, la commande suivante va supprimer le paquet `unrar` du système :

```
# dpkg -r unrar
(Reading database ... 269630 files and directories currently installed.)
Removing unrar (1:5.6.6-2) ...
Processing triggers for man-db (2.8.5-2) ...
```

L'opération de suppression effectue également un contrôle des dépendances, et un paquet ne peut être supprimé que si tous les autres paquets qui en dépendent sont également supprimés. Si vous essayez de le faire, vous obtiendrez un message d'erreur comme celui ci-dessous :

```
# dpkg -r p7zip
dpkg: dependency problems prevent removal of p7zip:
```

```
 winetricks depends on p7zip; however:
  Package p7zip is to be removed.
 p7zip-full depends on p7zip (= 16.02+dfsg-6).

dpkg: error processing package p7zip (--remove):
 dependency problems - not removing
Errors were encountered while processing:
 p7zip
```

Vous pouvez passer plusieurs noms de paquets à `dpkg -r`, de sorte qu'ils soient tous supprimés en même temps.

Lorsqu'un paquet est supprimé, les fichiers de configuration correspondants restent en place sur le système. Si vous voulez supprimer *tout* ce qui est associé au paquet, utilisez l'option `-P` (*purge*) au lieu de `-r`.

NOTE

Vous pouvez forcer `dpkg` à installer ou supprimer un paquet, même si les dépendances ne sont pas respectées, en ajoutant le paramètre `--force` comme dans `dpkg -i --force NOMDUPAQUET`. Cependant, une telle démarche laissera très probablement le paquet installé, voire votre système, dans un état dégradé. N'utilisez *pas* `--force` à moins d'être absolument sûr de ce que vous faites.

Obtenir des informations sur les paquets

Pour obtenir des informations sur un paquet `.deb`, telles que sa version, son architecture, son mainteneur, ses dépendances et autres, utilisez la commande `dpkg` avec le paramètre `-I`, suivi du nom de fichier du paquet que vous voulez inspecter :

```
# dpkg -I google-chrome-stable_current_amd64.deb
 new Debian package, version 2.0.
 size 59477810 bytes: control archive=10394 bytes.
    1222 bytes,    13 lines      control
   16906 bytes,   457 lines   *  postinst             #!/bin/sh
   12983 bytes,   344 lines   *  postrm               #!/bin/sh
    1385 bytes,    42 lines   *  prerm                #!/bin/sh
 Package: google-chrome-stable
 Version: 76.0.3809.100-1
 Architecture: amd64
 Maintainer: Chrome Linux Team <chromium-dev@chromium.org>
 Installed-Size: 205436
 Pre-Depends: dpkg (>= 1.14.0)
 Depends: ca-certificates, fonts-liberation, libappindicator3-1, libasound2 (>= 1.0.16),
```

```
libatk-bridge2.0-0 (>= 2.5.3), libatk1.0-0 (>= 2.2.0), libatspi2.0-0 (>= 2.9.90), libc6 (>=
2.16), libcairo2 (>= 1.6.0), libcups2 (>= 1.4.0), libdbus-1-3 (>= 1.5.12), libexpat1 (>=
2.0.1), libgcc1 (>= 1:3.0), libgdk-pixbuf2.0-0 (>= 2.22.0), libglib2.0-0 (>= 2.31.8),
libgtk-3-0 (>= 3.9.10), libnspr4 (>= 2:4.9-2~), libnss3 (>= 2:3.22), libpango-1.0-0 (>=
1.14.0), libpangocairo-1.0-0 (>= 1.14.0), libuuid1 (>= 2.16), libx11-6 (>= 2:1.4.99.1),
libx11-xcb1, libxcb1 (>= 1.6), libxcomposite1 (>= 1:0.3-1), libxcursor1 (>> 1.1.2),
libxdamage1 (>= 1:1.1), libxext6, libxfixes3, libxi6 (>= 2:1.2.99.4), libxrandr2 (>=
2:1.2.99.3), libxrender1, libxss1, libxtst6, lsb-release, wget, xdg-utils (>= 1.0.2)
 Recommends: libu2f-udev
 Provides: www-browser
 Section: web
 Priority: optional
 Description: The web browser from Google
  Google Chrome is a browser that combines a minimal design with sophisticated technology to
make the web faster, safer, and easier.
```

Afficher les paquets installés et leur contenu

Pour obtenir une liste de tous les paquets installés sur votre système, utilisez l'option `--get-selections`, comme dans `dpkg --get-selections`. Vous pouvez également obtenir une liste de tous les fichiers installés par un paquet spécifique en passant le paramètre `-L NOMDUPAQUET` à `dpkg`, comme ci-dessous :

```
# dpkg -L unrar
/.
/usr
/usr/bin
/usr/bin/unrar-nonfree
/usr/share
/usr/share/doc
/usr/share/doc/unrar
/usr/share/doc/unrar/changelog.Debian.gz
/usr/share/doc/unrar/copyright
/usr/share/man
/usr/share/man/man1
/usr/share/man/man1/unrar-nonfree.1.gz
```

Savoir à quel paquet appartient un fichier

Il est parfois nécessaire de savoir à quel paquet appartient un fichier donné dans votre système. Pour ce faire, vous pouvez utiliser l'outil `dpkg-query` suivi du paramètre `-S` et du chemin d'accès au fichier en question :

```
# dpkg-query -S /usr/bin/unrar-nonfree
unrar: /usr/bin/unrar-nonfree
```

Reconfigurer des paquets installés

Lorsqu'un paquet est installé, il y a une étape de configuration appelée *post-install* où un script est exécuté pour configurer tout ce qui est nécessaire au bon fonctionnement du logiciel, comme les permissions, le placement des fichiers de configuration, etc. Des questions peuvent également être posées à l'utilisateur afin de définir ses préférences relatives au fonctionnement du logiciel.

Parfois, en raison d'un fichier de configuration endommagé ou malformé, vous pouvez souhaiter restaurer les paramètres d'un paquet à son état initial. Ou alors, vous souhaitez peut-être modifier les réponses que vous avez fournies aux questions de configuration initiale. Pour ce faire, invoquez la commande `dpkg-reconfigure` suivie du nom du paquet.

Cette commande va sauvegarder les anciens fichiers de configuration, décompresser les nouveaux vers les bons répertoires et exécuter le script *post-install* fourni par le paquet, comme si le paquet avait été installé pour la première fois. Essayez de reconfigurer le paquet `tzdata` comme suit :

```
# dpkg-reconfigure tzdata
```

Advanced Package Tool (apt)

APT (*Advanced Package Tool*) est un système de gestion des paquets qui comprend un ensemble d'outils et simplifie considérablement l'installation, la mise à jour, la suppression et la gestion des paquets. APT offre des fonctionnalités telles que la recherche avancée et la résolution automatique des dépendances.

APT n'est pas un "remplacement" de `dpkg`. Vous pouvez l'imaginer comme un "frontal" qui facilite les opérations et vient combler les lacunes des fonctionnalités de `dpkg`, comme la résolution des dépendances.

APT travaille de concert avec les dépôts de logiciels qui contiennent les paquets disponibles à l'installation. Ces dépôts peuvent être un serveur local ou distant, ou (moins courant) un disque CD-ROM.

Les distributions Linux comme Debian et Ubuntu maintiennent leurs propres dépôts, et d'autres dépôts peuvent être mis à disposition par des développeurs ou des groupes d'utilisateurs dans le but de fournir des logiciels qui ne sont pas disponibles dans les dépôts officiels des distributions.

Il existe de nombreux outils qui interagissent avec APT, les principaux étant :

`apt-get`
: utilisé pour télécharger, installer, mettre à jour ou supprimer des paquets du système.

`apt-cache`
: utilisé pour effectuer des opérations dans l'index des paquets, notamment la recherche.

`apt-file`
: utilisé pour la recherche de fichiers à l'intérieur des paquets.

Il existe également un outil plus convivial appelé simplement `apt`, qui combine les options les plus utilisées de `apt-get` et `apt-cache` en une seule commande. La plupart des commandes pour `apt` sont les mêmes que celles pour `apt-get` et donc souvent interchangeables. Cependant, comme `apt` peut ne pas être installé sur le système, il est recommandé d'apprendre à utiliser `apt-get` et `apt-cache`.

NOTE `apt` et `apt-get` peuvent nécessiter une connexion réseau à partir du moment où les paquets et les index de paquets doivent être récupérés sur un serveur distant.

Mettre à jour l'index des paquets

Avant d'installer ou de mettre à jour un logiciel avec APT, il est recommandé d'actualiser l'index des paquets afin de récupérer les informations sur les nouveaux paquets et les paquets mis à jour. Cela s'effectue avec la commande `apt-get` suivie du paramètre `update` :

```
# apt-get update
Ign:1 http://dl.google.com/linux/chrome/deb stable InRelease
Hit:2 https://repo.skype.com/deb stable InRelease
Hit:3 http://us.archive.ubuntu.com/ubuntu disco InRelease
Hit:4 http://repository.spotify.com stable InRelease
Hit:5 http://dl.google.com/linux/chrome/deb stable Release
Hit:6 http://apt.pop-os.org/proprietary disco InRelease
Hit:7 http://ppa.launchpad.net/system76/pop/ubuntu disco InRelease
Hit:8 http://us.archive.ubuntu.com/ubuntu disco-security InRelease
Hit:9 http://us.archive.ubuntu.com/ubuntu disco-updates InRelease
Hit:10 http://us.archive.ubuntu.com/ubuntu disco-backports InRelease
Reading package lists... Done
```

TIP Au lieu de `apt-get update`, vous pouvez également utiliser `apt update`.

Installer et supprimer des paquets

Maintenant que l'index des paquets est à jour, vous pouvez installer un paquet. Cela se fait avec `apt-get install` suivi du nom du paquet que vous souhaitez installer :

```
# apt-get install xournal
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  xournal
0 upgraded, 1 newly installed, 0 to remove and 75 not upgraded.
Need to get 285 kB of archives.
After this operation, 1041 kB of additional disk space will be used.
```

De même, pour supprimer un paquet, utilisez `apt-get remove` suivi du nom du paquet :

```
# apt-get remove xournal
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages will be REMOVED:
  xournal
0 upgraded, 0 newly installed, 1 to remove and 75 not upgraded.
After this operation, 1041 kB disk space will be freed.
Do you want to continue? [Y/n]
```

Sachez que lors de l'installation ou de la suppression de paquets, APT va effectuer une résolution automatique des dépendances. Cela signifie que tout paquet supplémentaire requis par le paquet que vous installez *sera également installé*, et que les paquets qui dépendent du paquet que vous supprimez *seront également supprimés*. APT va toujours afficher ce qui va être installé ou supprimé avant de vous demander si vous voulez continuer :

```
# apt-get remove p7zip
Reading package lists... Done
Building dependency tree
The following packages will be REMOVED:
  android-libbacktrace android-libunwind android-libutils
  android-libziparchive android-sdk-platform-tools fastboot p7zip p7zip-full
0 upgraded, 0 newly installed, 8 to remove and 75 not upgraded.
After this operation, 6545 kB disk space will be freed.
```

```
Do you want to continue? [Y/n]
```

Notez que lorsqu'un paquet est retiré, les fichiers de configuration correspondants sont laissés en place sur le système. Pour supprimer le paquet *et* tout fichier de configuration, utilisez le paramètre `purge` au lieu de `remove` ou le paramètre `remove` avec l'option `--purge` :

```
# apt-get purge p7zip
```

ou bien

```
# apt-get remove --purge p7zip
```

TIP Vous pouvez également utiliser `apt install` et `apt remove`.

Réparer les dépendances cassées

Il est possible d'avoir des "dépendances cassées" sur un système. Cela signifie qu'un ou plusieurs paquets installés dépendent d'autres paquets qui n'ont pas été installés ou qui ne sont plus présents. Cela peut se produire suite à une erreur APT ou à cause d'un paquet installé manuellement.

Pour résoudre ce problème, utilisez la commande `apt-get install -f`. Cette opération vise à réparer (*fix*) les paquets cassés en installant les dépendances manquantes, afin de s'assurer que tous les paquets soient à nouveau cohérents.

TIP Vous pouvez également utiliser `apt install -f`.

Mettre à jour des paquets

APT peut être utilisé pour mettre à jour automatiquement tous les paquets installés vers les dernières versions disponibles dans les dépôts. Cela se fait à l'aide de la commande `apt-get upgrade`. Avant de l'invoquer, pensez à mettre à jour l'index des paquets avec `apt-get update`.

```
# apt-get update
Hit:1 http://us.archive.ubuntu.com/ubuntu disco InRelease
Hit:2 http://us.archive.ubuntu.com/ubuntu disco-security InRelease
Hit:3 http://us.archive.ubuntu.com/ubuntu disco-updates InRelease
Hit:4 http://us.archive.ubuntu.com/ubuntu disco-backports InRelease
Reading package lists... Done
```

```
# apt-get upgrade
Reading package lists... Done
Building dependency tree
Reading state information... Done
Calculating upgrade... Done
The following packages have been kept back:
  gnome-control-center
The following packages will be upgraded:
  cups cups-bsd cups-client cups-common cups-core-drivers cups-daemon
  cups-ipp-utils cups-ppdc cups-server-common firefox-locale-ar (...)

74 upgraded, 0 newly installed, 0 to remove and 1 not upgraded.
Need to get 243 MB of archives.
After this operation, 30.7 kB of additional disk space will be used.
Do you want to continue? [Y/n]
```

La liste récapitulative en bas de l'affichage indique le nombre de paquets qui seront mis à niveau, le nombre de paquets qui seront installés, retirés ou conservés, la taille totale du téléchargement et l'espace disque supplémentaire nécessaire pour mener à bien l'opération. Pour effectuer la mise à niveau, répondez par `Y` et attendez que `apt-get` complète la tâche.

Pour mettre à jour un seul paquet, il suffit d'invoquer `apt-get upgrade` suivi du nom du paquet.

TIP Vous pouvez également utiliser `apt upgrade` et `apt update`.

Le cache local

Lorsque vous installez ou mettez à jour un paquet, le fichier `.deb` correspondant est téléchargé dans un répertoire de cache local avant que le paquet ne soit installé. Par défaut, ce répertoire est `/var/cache/apt/archives`. Les fichiers partiellement téléchargés sont copiés dans `/var/cache/apt/archives/partial/`.

Au fur et à mesure que vous installez et mettez à jour des paquets, le répertoire de cache peut devenir assez volumineux. Pour récupérer de l'espace, vous pouvez vider le cache en utilisant la commande `apt-get clean`. Cette opération supprime le contenu des répertoires `/var/cache/apt/archives` et `/var/cache/apt/archives/partial/`.

TIP Vous pouvez également utiliser `apt clean`.

Rechercher des paquets

L'outil `apt-cache` peut être utilisé pour effectuer des opérations sur l'index des paquets, telles

que la recherche d'un paquet spécifique ou la liste des paquets qui contiennent un fichier particulier.

Pour effectuer une recherche, invoquez `apt-cache search` suivi de la chaîne de caractères à rechercher. Le résultat sera une liste de chaque paquet qui contient le motif recherché, soit dans son nom de paquet, soit dans sa description ou dans les fichiers fournis.

```
# apt-cache search p7zip
liblzma-dev - XZ-format compression library - development files
liblzma5 - XZ-format compression library
forensics-extra - Forensics Environment - extra console components (metapackage)
p7zip - 7zr file archiver with high compression ratio
p7zip-full - 7z and 7za file archivers with high compression ratio
p7zip-rar - non-free rar module for p7zip
```

Dans l'exemple ci-dessus, l'entrée `liblzma5 - XZ-format compression library` ne semble pas correspondre au motif recherché. Cependant, si nous affichons les informations complètes avec la description pour le paquet en invoquant le paramètre `show`, nous retrouvons la chaîne de caractères recherchée :

```
# apt-cache show liblzma5
Package: liblzma5
Architecture: amd64
Version: 5.2.4-1
Multi-Arch: same
Priority: required
Section: libs
Source: xz-utils
Origin: Ubuntu
Maintainer: Ubuntu Developers <ubuntu-devel-discuss@lists.ubuntu.com>
Original-Maintainer: Jonathan Nieder <jrnieder@gmail.com>
Bugs: https://bugs.launchpad.net/ubuntu/+filebug
Installed-Size: 259
Depends: libc6 (>= 2.17)
Breaks: liblzma2 (<< 5.1.1alpha+20110809-3~)
Filename: pool/main/x/xz-utils/liblzma5_5.2.4-1_amd64.deb
Size: 92352
MD5sum: 223533a347dc76a8cc9445cfc6146ec3
SHA1: 8ed14092fb1caecfebc556fda0745e1e74ba5a67
SHA256: 01020b5a0515dbc9a7c00b464a65450f788b0258c3fbb733ecad0438f5124800
Homepage: https://tukaani.org/xz/
Description-en: XZ-format compression library
```

```
XZ is the successor to the Lempel-Ziv/Markov-chain Algorithm
compression format, which provides memory-hungry but powerful
compression (often better than bzip2) and fast, easy decompression.
.
The native format of liblzma is XZ; it also supports raw (headerless)
streams and the older LZMA format used by lzma. (For 7-Zip's related
format, use the p7zip package instead.)
```

Vous pouvez utiliser des *expressions régulières* pour le motif de la recherche, ce qui permet des recherches très complexes (et précises). En revanche, ce sujet n'entre pas dans le cadre de cette leçon.

TIP Vous pouvez également utiliser `apt search` au lieu de `apt-cache search` et `apt show` au lieu de `apt-cache show`.

La liste des sources

APT utilise une liste de sources pour savoir où récupérer les paquets. Cette liste est conservée dans le fichier `sources.list`, situé dans le répertoire `/etc/apt`. Ce fichier peut être édité directement avec un éditeur de texte comme `vi`, `pico` ou `nano`, ou avec des outils graphiques comme `aptitude` ou `synaptic`.

Une entrée typique dans `sources.list` ressemble à ceci :

```
deb http://us.archive.ubuntu.com/ubuntu/ disco main restricted universe multiverse
```

La syntaxe comprend le type d'archive, l'URL, la distribution et un où plusieurs composants, avec :

Type d'archive
: Un dépôt peut contenir des paquets contenant des logiciels prêts à l'emploi (paquets binaires de type `deb`) ou le code source de ces logiciels (paquets sources de type `deb-src`). L'exemple ci-dessus fournit des paquets binaires.

URL
: L'URL du dépôt.

Distribution
: Le nom (ou nom de code) de la distribution pour laquelle les paquets sont fournis. Un dépôt peut héberger des paquets pour plusieurs distributions. Dans l'exemple ci-dessus, `disco` est le nom de code pour Ubuntu 19.04 *Disco Dingo*.

Composants

Chaque composant représente un ensemble de paquets. Ces composants peuvent varier selon les différentes distributions de Linux. Par exemple, sur Ubuntu et ses déclinaisons, on aura :

`main`

contient des paquets *open source* officiellement pris en charge.

`restricted`

contient des logiciels propriétaires officiellement pris en charge, comme les pilotes de cartes graphiques.

`universe`

contient des logiciels *open source* maintenus par la communauté.

`multiverse`

contient des logiciels propriétaires non pris en charge ou autrement protégés par un brevet.

Sur Debian, les principaux composants sont :

`main`

contient les paquets conformes aux principes du logiciel libre selon Debian (DFSG pour *Debian Free Software Guidelines*), qui ne dépendent pas de logiciels extérieurs à ce domaine pour fonctionner. Les paquets inclus ici sont considérés comme faisant partie de la distribution Debian.

`contrib`

contient les paquets conformes aux DFSG, mais qui dépendent d'autres paquets qui ne sont pas dans `main`.

`non-free`

contient les paquets qui ne sont pas conformes aux DFSG.

`security`

contient les mises à jour de sécurité.

`backports`

contient des versions plus récentes de paquets qui sont dans `main`. Le cycle de développement des versions stables de Debian est assez long (environ deux ans), et cela permet aux utilisateurs d'obtenir les paquets les plus récents sans avoir à modifier le dépôt principal `main`.

NOTE Vous pouvez en savoir plus sur les *Debian Free Software Guidelines* à l'adresse

> suivante : https://www.debian.org/social_contract#guidelines

Pour ajouter un nouveau dépôt pour obtenir des paquets, vous pouvez simplement ajouter la ligne correspondante (généralement fournie par le mainteneur du dépôt) à la fin de `sources.list`, enregistrer le fichier et recharger l'index du paquet avec `apt-get update`. Une fois que c'est fait, les paquets du nouveau dépôt seront disponibles à l'installation avec `apt-get install`.

Gardez à l'esprit que les lignes commençant par le caractère `#` sont considérées comme des commentaires et donc ignorées.

Le répertoire `/etc/apt/sources.list.d`

Le répertoire `/etc/apt/sources.list.d` vous permet d'ajouter des fichiers avec des dépôts supplémentaires exploitables par APT, et sans modifier le fichier principal `/etc/apt/sources.list`. Il s'agit là de simples fichiers texte avec la même syntaxe que celle décrite ci-dessus et l'extension de fichier `.list`.

Voici le contenu d'un fichier nommé `/etc/apt/sources.list.d/buster-backports.list` :

```
deb http://deb.debian.org/debian buster-backports main contrib non-free
deb-src http://deb.debian.org/debian buster-backports main contrib non-free
```

Afficher le contenu d'un paquet et rechercher des fichiers

Un outil nommé `apt-file` peut être utilisé pour effectuer d'autres opérations dans l'index des paquets, comme afficher le contenu d'un paquet ou repérer un paquet qui contient un fichier donné. Cet outil peut ne pas être installé par défaut sur votre système. Dans ce cas, vous pouvez généralement l'installer en utilisant `apt-get` :

```
# apt-get install apt-file
```

Après l'installation, vous devrez mettre à jour le cache de paquets utilisé pour `apt-file` :

```
# apt-file update
```

Cela ne prend généralement que quelques secondes. Une fois que c'est fait, `apt-file` est prêt à l'emploi.

Pour afficher le contenu d'un paquet, utilisez le paramètre `list` suivi du nom du paquet :

```
# apt-file list unrar
unrar: /usr/bin/unrar-nonfree
unrar: /usr/share/doc/unrar/changelog.Debian.gz
unrar: /usr/share/doc/unrar/copyright
unrar: /usr/share/man/man1/unrar-nonfree.1.gz
```

TIP Vous pouvez également utiliser `apt list` au lieu de `apt-file list`.

Vous pouvez rechercher un fichier dans tous les paquets en utilisant le paramètre `search` suivi du nom du fichier. Par exemple, si vous souhaitez savoir quel paquet fournit un fichier appelé `libSDL2.so`, vous pouvez utiliser :

```
# apt-file search libSDL2.so
libsdl2-dev: /usr/lib/x86_64-linux-gnu/libSDL2.so
```

La réponse est le paquet `libsdl2-dev`, qui fournit le fichier `/usr/lib/x86_64-linux-gnu/libSDL2.so`.

La différence entre `apt-file search` et `dpkg-query` est que `apt-file search` prend également en compte les paquets non installés dans l'opération de recherche, alors que `dpkg-query` ne peut afficher que les fichiers qui appartiennent à un paquet installé.

Exercices guidés

1. Quelle est la commande pour installer un paquet nommé `paquet.deb` en utilisant `dpkg` ?

2. En utilisant `dpkg-query`, trouvez le paquet qui contient un fichier nommé `7zr.1.gz`.

3. Pouvez-vous supprimer un paquet nommé `unzip` du système en utilisant `dpkg -r unzip` si le paquet `file-roller` en dépend ? Si ce n'est pas le cas, quelle serait la bonne façon de procéder ?

4. En utilisant `apt-file`, comment pouvez-vous savoir quel paquet contient le fichier `unrar` ?

5. En utilisant `apt-cache`, quelle est la commande pour afficher les informations relatives au paquet `gimp` ?

Exercices d'approfondissement

1. Considérez un dépôt avec des paquets source Debian pour la distribution `xenial`, hébergé à `http://us.archive.ubuntu.com/ubuntu/` et avec des paquets pour le composant `universe`. Quelle serait la ligne correspondante à ajouter à `/etc/apt/sources.list` ?

2. Lors de la compilation d'un programme, vous vous retrouvez confronté à un message d'erreur qui vous signale que le fichier d'en-tête `zip-io.h` n'est pas présent sur votre système. Comment pouvez-vous savoir quel paquet fournit ce fichier ?

3. Comment pouvez-vous passer outre un avertissement de dépendance et supprimer un paquet en utilisant `dpkg`, même s'il y a des paquets qui en dépendent sur le système ?

4. Comment pouvez-vous obtenir plus d'informations sur un paquet appelé `midori` en utilisant `apt` ?

5. Avant d'installer ou de mettre à jour des paquets avec `apt`, quelle commande doit être utilisée pour s'assurer que l'index des paquets est à jour ?

Résumé

Dans cette leçon, vous avez appris à :

- Comment utiliser `dpkg` pour installer et supprimer des paquets.
- Comment afficher la liste des paquets installés ainsi que leur contenu.
- Comment reconfigurer un paquet installé.
- La commande `apt` et comment installer, mettre à jour et supprimer des paquets en l'utilisant.
- Comment utiliser `apt-cache` pour rechercher des paquets.
- Comment fonctionne le fichier `/etc/apt/sources.list`.
- Comment utiliser `apt-file` pour afficher le contenu d'un paquet, ou comment savoir quel paquet contient un fichier donné.

Les commandes suivantes ont été abordées :

`dpkg -i`
: Installe un seul paquet, ou une liste de paquets séparés par des espaces.

`dpkg -r`
: Supprime un paquet, ou une liste de paquets séparés par des espaces.

`dpkg -I`
: Inspecte un paquet, en fournissant des détails sur le logiciel qu'il installe et les dépendances nécessaires.

`dpkg --get-selections`
: Liste tous les paquets que `dpkg` a installés sur le système.

`dpkg -L`
: Affiche une liste de tous les fichiers installés par un paquet donné.

`dpkg-query`
: Avec un nom de fichier fourni en argument, cette commande affiche le paquet qui a installé le fichier.

`dpkg-reconfigure`
: Cette commande va relancer le script *post-install* d'un paquet afin qu'un administrateur puisse faire des ajustements de configuration à l'installation du paquet.

`apt-get update`

Cette commande va mettre à jour l'index local des paquets pour qu'il corresponde à ce qui est disponible dans les dépôts configurés dans le répertoire `/etc/apt/`.

`apt-get install`

Cette commande va télécharger un paquet depuis un dépôt distant et l'installer avec ses dépendances. Elle peut également être utilisée pour installer un paquet Debian qui a déjà été téléchargé.

`apt-get remove`

Cette commande permet de désinstaller le(s) paquet(s) spécifié(s) du système.

`apt-cache show`

Tout comme la commande `dpkg -I`, cette commande peut être utilisée pour afficher des détails sur un paquet donné.

`apt-cache search`

Cette commande permet de rechercher un paquet donné dans votre base de données APT locale en cache.

`apt-file update`

Cette commande va mettre à jour le cache de paquets afin que la commande `apt-file` puisse interroger son contenu.

`apt-file search`

Cette commande permet de rechercher le paquet qui contient un fichier. Une liste de tous les paquets contenant le motif recherché est renvoyée.

`apt-file list`

Cette commande est utilisée pour afficher le contenu d'un paquet, tout comme la commande `dpkg -L`.

Réponses aux exercices guidés

1. Quelle est la commande pour installer un paquet nommé `paquet.deb` en utilisant `dpkg` ?

   Passez le paramètre `-i` à `dpkg` :

   ```
   # dpkg -i paquet.deb
   ```

2. En utilisant `dpkg-query`, trouvez le paquet qui contient un fichier nommé `7zr.1.gz`.

   Ajoutez le paramètre `-S` à `dpkg-query`:

   ```
   # dpkg-query -S 7zr.1.gz
   ```

3. Pouvez-vous supprimer un paquet nommé `unzip` du système en utilisant `dpkg -r unzip` si le paquet `file-roller` en dépend ? Si ce n'est pas le cas, quelle serait la bonne façon de procéder ?

   Non. `dpkg` ne résoudra pas les dépendances et ne vous permettra pas de supprimer un paquet si un autre paquet installé en dépend. Dans cet exemple, vous pourriez d'abord supprimer `file-roller` (en supposant que rien n'en dépend) et ensuite supprimer `unzip`, ou supprimer les deux en même temps avec :

   ```
   # dpkg -r unzip file-roller
   ```

4. En utilisant `apt-file`, comment pouvez-vous savoir quel paquet contient le fichier `unrar` ?

   Utilisez le paramètre `search` suivi du chemin (ou du nom de fichier) :

   ```
   # apt-file search /usr/bin/unrar
   ```

5. En utilisant `apt-cache`, quelle est la commande pour afficher les informations relatives au paquet `gimp` ?

   Utilisez le paramètre `show` suivi du nom du paquet :

   ```
   # apt-cache show gimp
   ```

Réponses aux exercices d'approfondissement

1. Considérez un dépôt avec des paquets source Debian pour la distribution `xenial`, hébergé à `http://us.archive.ubuntu.com/ubuntu/` et avec des paquets pour le composant `universe`. Quelle serait la ligne correspondante à ajouter à `/etc/apt/sources.list` ?

   Les paquets sources sont du type `deb-src`, donc la ligne devrait être :

   ```
   deb-src http://us.archive.ubuntu.com/ubuntu/ xenial universe
   ```

   Cette ligne pourrait également être ajoutée dans un fichier `.list` dans `/etc/apt/sources.list.d/`. Vous êtes libre de choisir le nom, mais il est censé être descriptif, quelque chose comme `xenial_sources.list`.

2. Lors de la compilation d'un programme, vous vous retrouvez confronté à un message d'erreur qui vous signale que le fichier d'en-tête `zip-io.h` n'est pas présent sur votre système. Comment pouvez-vous savoir quel paquet fournit ce fichier ?

   Utilisez `apt-file search` pour savoir quel paquet contient un fichier qui n'est pas présent sur le système :

   ```
   # apt-file search zzip-io.h
   ```

3. Comment pouvez-vous passer outre un avertissement de dépendance et supprimer un paquet en utilisant `dpkg`, même s'il y a des paquets qui en dépendent sur le système ?

   Le paramètre `--force` peut être utilisé, mais cela ne devrait *jamais* être fait à moins que vous ne sachiez exactement ce que vous faites, car il y a un grand risque que votre système se retrouve dans un état inconsistant ou "cassé".

4. Comment pouvez-vous obtenir plus d'informations sur un paquet appelé `midori` en utilisant `apt` ?

   Utilisez `apt-cache show` suivi du nom du paquet :

   ```
   # apt-cache show midori
   ```

5. Avant d'installer ou de mettre à jour des paquets avec `apt`, quelle commande doit être utilisée pour s'assurer que l'index des paquets est à jour ?

Il faut utiliser `apt-get update`. Cette opération va télécharger les derniers index de paquets depuis les dépôts renseignés dans le fichier `/etc/apt/sources.list` ou dans le répertoire `/etc/apt/sources.list.d/`.

102.5 Utilisation des gestionnaires de paquetage RPM et YUM

Référence aux objectifs de LPI

LPIC-1 v5, Exam 101, Objective 102.5

Valeur

3

Liste partielle de termes, fichiers et utilitaires utilisés pour cet objectif

- Installation, réinstallation, mise à jour et suppression des paquetages avec RPM, YUM et Zypper.
- Obtention d'informations sur un paquetage RPM comme la version, le contenu, les dépendances, l'intégrité du paquetage, la signature et l'état d'installation.
- Détermination des fichiers relatifs à un paquetage donné, et recherche du paquetage auquel appartient un fichier donné.
- Connaissance de base de dnf.

Partial list of the used files, terms and utilities

- `rpm`
- `rpm2cpio`
- `/etc/yum.conf`
- `/etc/yum.repos.d/`
- `yum`
- `zypper`

102.5 Leçon 1

Certification :	LPIC-1
Version :	5.0
Thème :	102 Installer Linux et gérer les paquets
Objectif :	102.5 Utiliser les gestionnaires de paquets RPM et Yum
Leçon :	1 sur 1

Introduction

Il y a longtemps, lorsque Linux en était encore à ses débuts, le moyen le plus courant de distribuer un logiciel était sous forme d'un fichier compressé (généralement une archive `.tar.gz`) avec le code source, qu'il fallait décompresser et compiler soi-même.

Cependant, à mesure que la quantité et la complexité des logiciels augmentaient, la nécessité de trouver un moyen de distribuer des logiciels pré-compilés s'imposait. En effet, tout le monde n'avait pas les ressources, en termes de temps et de puissance de calcul, pour compiler de gros projets comme le noyau Linux ou un serveur X.

Petit à petit, les efforts pour normaliser un mode de distribution de ces "paquets" logiciels se sont multipliés, et les premiers gestionnaires de paquets ont vu le jour. Ces outils ont considérablement facilité l'installation, la configuration ou la suppression de logiciels sur un système.

Parmi ceux-ci figurait le *RPM Package Manager* et son outil correspondant (`rpm`), développé par Red Hat. De nos jours, ils sont largement utilisés non seulement sur Red Hat Enterprise Linux (RHEL) lui-même, mais aussi sur ses descendants, comme Fedora, CentOS et Oracle Linux, d'autres

distributions comme openSUSE et même d'autres systèmes d'exploitation, comme AIX d'IBM.

D'autres outils de gestion des paquets populaires sur les distributions compatibles Red Hat sont `yum` (YellowDog Updater Modified), `dnf` (Dandified YUM) et `zypper`, qui peuvent simplifier de nombreux aspects de l'installation, de la maintenance et de la suppression des paquets, ce qui rend la tâche beaucoup plus facile.

Dans cette leçon, nous allons apprendre à utiliser `rpm`, `yum`, `dnf` et `zypper` pour obtenir, installer, maintenir et supprimer des logiciels sur un système Linux.

NOTE

Malgré l'utilisation du même format de paquets, il y a des différences internes entre les distributions, de sorte qu'un paquet conçu spécifiquement pour openSUSE ne fonctionne pas forcément sur un système RHEL, et vice-versa. Lorsque vous recherchez des paquets, vérifiez toujours leur compatibilité et essayez d'en trouver un qui soit adapté à votre distribution spécifique dans la mesure du possible.

Le gestionnaire de paquets RPM (rpm)

Le gestionnaire de paquets RPM (`rpm`) est l'outil de référence pour la gestion des paquets logiciels sur les systèmes basés sur (ou dérivés de) Red Hat.

Installer, mettre à jour et supprimer des paquets

L'opération la plus basique consiste à installer un paquet, ce qui peut être effectué avec :

```
# rpm -i PAQUET
```

Où `PAQUET` est le nom du paquet `.rpm` que vous souhaitez installer.

S'il existe une version précédente d'un paquet sur le système, vous pouvez passer à une version plus récente en utilisant l'option `-U` :

```
# rpm -U PAQUET
```

S'il n'y a pas de version précédente de `PAQUET` installée, alors une nouvelle copie sera installée. Pour éviter cela et *seulement* mettre à jour un paquet *installé*, utilisez l'option `-F`.

Dans les deux cas, vous pouvez ajouter l'option `-v` pour obtenir un affichage détaillé (plus d'informations sont fournies pendant l'installation) et `-h` pour obtenir des signes dièse (#) qui s'affichent en guise de barre de progression. Plusieurs options peuvent être combinées, de sorte

que `rpm -i -v -h` est identique à `rpm -ivh`.

Pour supprimer un paquet installé, passez l'option `-e` (comme "erase" ou *effacer*) à `rpm`, suivie du nom du paquet que vous souhaitez supprimer :

```
# rpm -e wget
```

Si un paquet installé dépend du paquet en cours de suppression, vous obtiendrez un message d'erreur :

```
# rpm -e unzip
error: Failed dependencies:
    /usr/bin/unzip is needed by (installed) file-roller-3.28.1-2.el7.x86_64
```

Pour venir à bout de l'opération, vous devrez d'abord supprimer les paquets qui dépendent de celui que vous souhaitez supprimer (dans l'exemple ci-dessus, `file-roller`). Vous pouvez fournir plusieurs noms de paquets en argument à `rpm -e` pour supprimer plusieurs paquets à la fois.

Gérer les dépendances

La plupart du temps, un paquet dépendra d'autres paquets pour fonctionner comme prévu. À titre d'exemple, un éditeur d'images pourra recourir à certaines bibliothèques pour gérer les fichiers JPG, ou un logiciel pourra avoir besoin d'un kit de développement de widgets comme Qt ou GTK pour son interface utilisateur.

`rpm` va vérifier si ces dépendances sont installées sur votre système, et va échouer à installer le paquet dans le cas contraire. Dans ce cas, `rpm` affichera la liste des composants manquants. Cependant, il n'est pas capable de résoudre les dépendances par lui-même.

Dans l'exemple ci-dessous, l'utilisateur a essayé d'installer un paquet pour l'éditeur d'images GIMP, mais certaines dépendances étaient manquantes :

```
# rpm -i gimp-2.8.22-1.el7.x86_64.rpm
error: Failed dependencies:
    babl(x86-64) >= 0.1.10 is needed by gimp-2:2.8.22-1.el7.x86_64
    gegl(x86-64) >= 0.2.0 is needed by gimp-2:2.8.22-1.el7.x86_64
    gimp-libs(x86-64) = 2:2.8.22-1.el7 is needed by gimp-2:2.8.22-1.el7.x86_64
    libbabl-0.1.so.0()(64bit) is needed by gimp-2:2.8.22-1.el7.x86_64
    libgegl-0.2.so.0()(64bit) is needed by gimp-2:2.8.22-1.el7.x86_64
```

```
    libgimp-2.0.so.0()(64bit) is needed by gimp-2:2.8.22-1.el7.x86_64
    libgimpbase-2.0.so.0()(64bit) is needed by gimp-2:2.8.22-1.el7.x86_64
    libgimpcolor-2.0.so.0()(64bit) is needed by gimp-2:2.8.22-1.el7.x86_64
    libgimpconfig-2.0.so.0()(64bit) is needed by gimp-2:2.8.22-1.el7.x86_64
    libgimpmath-2.0.so.0()(64bit) is needed by gimp-2:2.8.22-1.el7.x86_64
    libgimpmodule-2.0.so.0()(64bit) is needed by gimp-2:2.8.22-1.el7.x86_64
    libgimpthumb-2.0.so.0()(64bit) is needed by gimp-2:2.8.22-1.el7.x86_64
    libgimpui-2.0.so.0()(64bit) is needed by gimp-2:2.8.22-1.el7.x86_64
    libgimpwidgets-2.0.so.0()(64bit) is needed by gimp-2:2.8.22-1.el7.x86_64
    libmng.so.1()(64bit) is needed by gimp-2:2.8.22-1.el7.x86_64
    libwmf-0.2.so.7()(64bit) is needed by gimp-2:2.8.22-1.el7.x86_64
    libwmflite-0.2.so.7()(64bit) is needed by gimp-2:2.8.22-1.el7.x86_64
```

C'est à l'utilisateur de trouver les paquets `.rpm` avec les dépendances correspondantes et de les installer. Les gestionnaires de paquets comme `yum`, `zypper` et `dnf` disposent d'outils qui permettent de savoir quel paquet fournit un fichier spécifique. Nous y reviendrons un peu plus loin dans cette leçon.

Afficher la liste des paquets installés

Pour obtenir une liste de tous les paquets installés sur votre système, utilisez la commande `rpm -qa` (comme dans "query all").

```
# rpm -qa
selinux-policy-3.13.1-229.el7.noarch
pciutils-libs-3.5.1-3.el7.x86_64
redhat-menus-12.0.2-8.el7.noarch
grubby-8.28-25.el7.x86_64
hunspell-en-0.20121024-6.el7.noarch
dejavu-fonts-common-2.33-6.el7.noarch
xorg-x11-drv-dummy-0.3.7-1.el7.1.x86_64
libevdev-1.5.6-1.el7.x86_64
[...]
```

Obtenir des informations sur les paquets

Pour obtenir des informations sur un paquet *installé*, comme son numéro de version, son architecture, sa date d'installation, le nom du mainteneur du paquet, une description sommaire, etc., utilisez `rpm` avec l'option `-qi` (comme "query info"), suivie du nom du paquet.

```
# rpm -qi unzip
```

```
Name        : unzip
Version     : 6.0
Release     : 19.el7
Architecture: x86_64
Install Date: Sun 25 Aug 2019 05:14:39 PM EDT
Group       : Applications/Archiving
Size        : 373986
License     : BSD
Signature   : RSA/SHA256, Wed 25 Apr 2018 07:50:02 AM EDT, Key ID 24c6a8a7f4a80eb5
Source RPM  : unzip-6.0-19.el7.src.rpm
Build Date  : Wed 11 Apr 2018 01:24:53 AM EDT
Build Host  : x86-01.bsys.centos.org
Relocations : (not relocatable)
Packager    : CentOS BuildSystem <http://bugs.centos.org>
Vendor      : CentOS
URL         : http://www.info-zip.org/UnZip.html
Summary     : A utility for unpacking zip files
Description :
The unzip utility is used to list, test, or extract files from a zip
archive. Zip archives are commonly found on MS-DOS systems. The zip
utility, included in the zip package, creates zip archives. Zip and
unzip are both compatible with archives created by PKWARE(R)'s PKZIP
for MS-DOS, but the programs' options and default behaviors do differ
in some respects.

Install the unzip package if you need to list, test or extract files from
a zip archive.
```

Pour obtenir une liste des fichiers contenus dans un paquet *installé*, utilisez l'option `-ql` (comme "query list") suivie du nom du paquet :

```
# rpm -ql unzip
/usr/bin/funzip
/usr/bin/unzip
/usr/bin/unzipsfx
/usr/bin/zipgrep
/usr/bin/zipinfo
/usr/share/doc/unzip-6.0
/usr/share/doc/unzip-6.0/BUGS
/usr/share/doc/unzip-6.0/LICENSE
/usr/share/doc/unzip-6.0/README
/usr/share/man/man1/funzip.1.gz
/usr/share/man/man1/unzip.1.gz
```

```
/usr/share/man/man1/unzipsfx.1.gz
/usr/share/man/man1/zipgrep.1.gz
/usr/share/man/man1/zipinfo.1.gz
```

Si vous souhaitez obtenir des informations ou une liste de fichiers d'un paquet qui n'a *pas* encore été installé, ajoutez simplement l'option `-p` aux commandes ci-dessus, suivie du nom du fichier RPM (`FICHIER`). Donc `rpm -qi PAQUET` devient `rpm -qip FICHIER`, et `rpm -ql PAQUET` devient `rpm -qlp FICHIER`, comme indiqué ci-dessous.

```
# rpm -qip atom.x86_64.rpm
Name        : atom
Version     : 1.40.0
Release     : 0.1
Architecture: x86_64
Install Date: (not installed)
Group       : Unspecified
Size        : 570783704
License     : MIT
Signature   : (none)
Source RPM  : atom-1.40.0-0.1.src.rpm
Build Date  : sex 09 ago 2019 12:36:31 -03
Build Host  : b01bbeaf3a88
Relocations : /usr
URL         : https://atom.io/
Summary     : A hackable text editor for the 21st Century.
Description :
A hackable text editor for the 21st Century.
```

```
# rpm -qlp atom.x86_64.rpm
/usr/bin/apm
/usr/bin/atom
/usr/share/applications/atom.desktop
/usr/share/atom
/usr/share/atom/LICENSE
/usr/share/atom/LICENSES.chromium.html
/usr/share/atom/atom
/usr/share/atom/atom.png
/usr/share/atom/blink_image_resources_200_percent.pak
/usr/share/atom/content_resources_200_percent.pak
/usr/share/atom/content_shell.pak
```

```
(suite du listing)
```

Savoir à quel paquet appartient un fichier

Pour savoir quel paquet installé possède un fichier, utilisez l'option `-qf` (pensez à "query file") suivie du chemin complet du fichier :

```
# rpm -qf /usr/bin/unzip
unzip-6.0-19.el7.x86_64
```

Dans l'exemple ci-dessus, le fichier `/usr/bin/unzip` appartient au paquet `unzip-6.0-19.el7.x86_64`.

YellowDog Updater Modified (YUM)

`yum` a été développé à l'origine sous le nom de *Yellow Dog Updater* (YUP), un outil pour gérer les paquets de la distribution Yellow Dog Linux. Au fil du temps, il a évolué pour gérer les paquets sur d'autres systèmes basés sur RPM, tels que Fedora, CentOS, Red Hat Enterprise Linux et Oracle Linux.

D'un point de vue fonctionnel, il est similaire à l'outil `apt` sur les systèmes basés sur Debian, en étant capable de rechercher, installer, mettre à jour et supprimer des paquets tout en gérant automatiquement les dépendances. `yum` peut être utilisé pour installer un seul paquet ou pour mettre à jour d'une traite un système entier.

Rechercher des paquets

Avant d'installer un paquet, vous devez connaître son nom. Pour ce faire, vous pouvez effectuer une recherche avec `yum search MOTIF`, où `MOTIF` est le nom du paquet recherché. Le résultat est une liste de paquets dont le nom ou le résumé contient le motif de recherche spécifié. Par exemple, si vous avez besoin d'un outil pour gérer les fichiers compressés 7Zip (avec l'extension `.7z`), vous pouvez utiliser :

```
# yum search 7zip
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
 * base: mirror.ufscar.br
 * epel: mirror.globo.com
 * extras: mirror.ufscar.br
 * updates: mirror.ufscar.br
```

```
========================== N/S matchyutr54ed: 7zip ==========================
p7zip-plugins.x86_64 : Additional plugins for p7zip
p7zip.x86_64 : Very high compression ratio file archiver
p7zip-doc.noarch : Manual documentation and contrib directory
p7zip-gui.x86_64 : 7zG - 7-Zip GUI version

  Name and summary matches only, use "search all" for everything.
```

Installer, mettre à jour et supprimer des paquets

Pour installer un paquet à l'aide de `yum`, utilisez la commande `yum install PAQUET`, où `PAQUET` est le nom du paquet. `yum` va récupérer le paquet et les dépendances correspondantes depuis un dépôt en ligne et installer le tout sur votre système.

```
# yum install p7zip
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
 * base: mirror.ufscar.br
 * epel: mirror.globo.com
 * extras: mirror.ufscar.br
 * updates: mirror.ufscar.br
Resolving Dependencies
--> Running transaction check
---> Package p7zip.x86_64 0:16.02-10.el7 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package        Arch            Version               Repository     Size
================================================================================
Installing:
 p7zip          x86_64          16.02-10.el7          epel          604 k

Transaction Summary
================================================================================
Install  1 Package

Total download size: 604 k
Installed size: 1.7 M
Is this ok [y/d/N]:
```

Pour mettre à jour un paquet installé, utilisez `yum update PAQUET`, où `PAQUET` est le nom du paquet que vous souhaitez mettre à jour. Par exemple :

```
# yum update wget
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
 * base: mirror.ufscar.br
 * epel: mirror.globo.com
 * extras: mirror.ufscar.br
 * updates: mirror.ufscar.br
Resolving Dependencies
--> Running transaction check
---> Package wget.x86_64 0:1.14-18.el7 will be updated
---> Package wget.x86_64 0:1.14-18.el7_6.1 will be an update
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package     Arch          Version                   Repository      Size
================================================================================
Updating:
 wget        x86_64        1.14-18.el7_6.1           updates        547 k

Transaction Summary
================================================================================
Upgrade  1 Package

Total download size: 547 k
Is this ok [y/d/N]:
```

Si vous ne fournissez aucun nom de paquet en argument, vous pouvez mettre à jour tous les paquets du système pour lesquels une mise à jour est disponible.

Pour vérifier si une mise à jour est disponible pour un paquet spécifique, utilisez `yum check-update PAQUET`. De manière similaire, si vous omettez le nom du paquet, `yum` vérifiera les mises à jour pour chacun des paquets installés sur le système.

Pour supprimer un paquet installé, utilisez `yum remove PAQUET`, où `PAQUET` est le nom du paquet que vous souhaitez supprimer.

Savoir quel paquet fournit un fichier donné

Dans un exemple précédent, nous avons montré une tentative d'installation de l'éditeur d'images `gimp`, qui a échoué pour cause de dépendances non satisfaites. Certes, `rpm` montre quels fichiers sont manquants, mais il ne liste pas le nom des paquets qui les fournissent.

Par exemple, l'une des dépendances manquantes était `libgimpui-2.0.so.0`. Pour voir quel paquet la fournit, vous pouvez utiliser `yum whatprovides` suivi du nom du fichier recherché :

```
# yum whatprovides libgimpui-2.0.so.0
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
 * base: mirror.ufscar.br
 * epel: mirror.globo.com
 * extras: mirror.ufscar.br
 * updates: mirror.ufscar.br
2:gimp-libs-2.8.22-1.el7.i686 : GIMP libraries
Repo        : base
Matched from:
Provides    : libgimpui-2.0.so.0
```

La réponse est `gimp-libs-2.8.22-1.el7.i686`. Partant de là, vous pouvez installer le paquet avec la commande `yum install gimp-libs`.

Cela fonctionne également pour les fichiers déjà présents sur votre système. Par exemple, si vous souhaitez savoir d'où vient le fichier `/etc/hosts`, vous pouvez utiliser :

```
# yum whatprovides /etc/hosts
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
 * base: mirror.ufscar.br
 * epel: mirror.globo.com
 * extras: mirror.ufscar.br
 * updates: mirror.ufscar.br
setup-2.8.71-10.el7.noarch : A set of system configuration and setup files
Repo        : base
Matched from:
Filename    : /etc/hosts
```

La réponse est `setup-2.8.71-10.el7.noarch`.

En savoir plus sur un paquet

Pour obtenir des informations sur un paquet, comme sa version, son architecture, sa description, sa taille et plus encore, utilisez `yum info PAQUET` où `PAQUET` est le nom du paquet pour lequel vous voulez en savoir plus :

```
# yum info firefox
Last metadata expiration check: 0:24:16 ago on Sat 21 Sep 2019 02:39:43 PM -03.
Installed Packages
Name         : firefox
Version      : 69.0.1
Release      : 3.fc30
Architecture : x86_64
Size         : 268 M
Source       : firefox-69.0.1-3.fc30.src.rpm
Repository   : @System
From repo    : updates
Summary      : Mozilla Firefox Web browser
URL          : https://www.mozilla.org/firefox/
License      : MPLv1.1 or GPLv2+ or LGPLv2+
Description  : Mozilla Firefox is an open-source web browser, designed
             : for standards compliance, performance and portability.
```

Gérer les dépôts de logiciels

Pour `yum`, les dépôts de paquets (“repos”) figurent dans le répertoire `/etc/yum.repos.d/`. Chaque dépôt est représenté par un fichier `.repo`, comme `CentOS-Base.repo`.

Des dépôts supplémentaires peuvent être ajoutés par l’utilisateur en ajoutant un fichier `.repo` dans le répertoire mentionné ci-dessus, ou à la fin de `/etc/yum.conf`. Ceci étant dit, la procédure recommandée pour ajouter ou gérer les dépôts de paquets passe par l’outil `yum-config-manager`.

Pour ajouter un dépôt, utilisez le paramètre `--add-repo` suivi de l’URL d’un fichier `.repo`.

```
# yum-config-manager --add-repo https://rpms.remirepo.net/enterprise/remi.repo
Loaded plugins: fastestmirror, langpacks
adding repo from: https://rpms.remirepo.net/enterprise/remi.repo
grabbing file https://rpms.remirepo.net/enterprise/remi.repo to /etc/yum.repos.d/remi.repo
repo saved to /etc/yum.repos.d/remi.repo
```

Pour obtenir une liste de tous les dépôts disponibles, utilisez `yum repolist all`. Vous obtiendrez

un résultat semblable à celui-ci :

```
# yum repolist all
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
 * base: mirror.ufscar.br
 * epel: mirror.globo.com
 * extras: mirror.ufscar.br
 * updates: mirror.ufscar.br
repo id                      repo name                    status
updates/7/x86_64             CentOS-7 - Updates           enabled:  2,500
updates-source/7             CentOS-7 - Updates Sources   disabled
```

Les dépôts désactivés (`disabled`) seront ignorés lors de l'installation ou de la mise à jour des logiciels. Pour activer ou désactiver un dépôt, invoquez l'outil `yum-config-manager` en renseignant l'identifiant du dépôt.

Dans l'exemple ci-dessus, l'identifiant du dépôt est indiqué dans la première colonne (`repo id`) de chaque ligne. Utilisez uniquement la partie avant le premier `/`, de sorte que l'identifiant pour le dépôt `CentOS-7 - Updates` est `updates`, et non pas `updates/7/x86_64`.

```
# yum-config-manager --disable updates
```

La commande ci-dessus va désactiver le dépôt `updates`. Pour le réactiver, utilisez :

```
# yum-config-manager --enable updates
```

NOTE

Yum conserve les paquets téléchargés et les métadonnées associées dans un répertoire de cache (généralement `/var/cache/yum`). Au fur et à mesure que le système est mis à jour et que de nouveaux paquets sont installés, ce cache peut devenir assez volumineux. Pour nettoyer le cache et récupérer de l'espace disque, vous pouvez utiliser la commande `yum clean` suivie de ce qu'il faut nettoyer. Les paramètres les plus pertinents sont `packages` (`yum clean packages`) pour supprimer les paquets téléchargés et `metadata` (`yum clean metadata`) pour supprimer les métadonnées associées. Consultez la page de manuel de `yum` (tapez `man yum`) pour plus d'informations.

DNF

`dnf` est l'outil de gestion de paquets utilisé sous Fedora, c'est un fork de `yum`. De ce fait, les commandes et les paramètres se ressemblent beaucoup. Cette section va vous donner un aperçu rapide de `dnf`.

Recherche de paquets

`dnf search MOTIF`, où `MOTIF` correspond à ce que vous recherchez. Par exemple, `dnf search unzip` affichera tous les paquets qui contiennent le mot `unzip` dans le nom ou la description.

En savoir plus sur un paquet

```
dnf info PAQUET
```

Installer des paquets

`dnf install PAQUET`, où `PAQUET` est le nom du paquet que vous souhaitez installer. Vous pouvez trouver le nom exact en effectuant une recherche.

Supprimer des paquets

```
dnf remove PAQUET
```

Mettre à jour des paquets

`dnf upgrade PAQUET` pour mettre à jour un seul paquet. Invoquez la commande sans paramètre pour mettre à jour l'ensemble des paquets du système.

Savoir à quel paquet appartient un fichier

```
dnf provides FICHIER
```

Obtenir une liste de tous les paquets installés sur le système

```
dnf list --installed
```

Afficher le contenu d'un paquet

```
dnf repoquery -l PAQUET
```

NOTE

`dnf` dispose d'un système d'aide intégré, qui affiche des informations détaillées (comme les paramètres supplémentaires) pour chaque commande. Pour l'utiliser, tapez `dnf help` suivi de la commande, comme `dnf help install`.

Gérer les dépôts de logiciels

Tout comme `yum` et `zypper`, `dnf` fonctionne avec des dépôts de logiciels (*repos*). Chaque distribution dispose d'une liste de dépôts par défaut, et les administrateurs peuvent ajouter ou

supprimer des dépôts en cas de besoin.

Pour obtenir une liste de tous les dépôts disponibles, utilisez `dnf repolist`. Pour répertorier uniquement les dépôts activés, ajoutez l'option `--enabled`, et pour afficher uniquement les dépôts désactivés, utilisez l'option `--disabled`.

```
# dnf repolist
Last metadata expiration check: 0:20:09 ago on Sat 21 Sep 2019 02:39:43 PM -03.
repo id                    repo name                                    status
*fedora                    Fedora 30 - x86_64                           56,582
*fedora-modular            Fedora Modular 30 - x86_64                      135
*updates                   Fedora 30 - x86_64 - Updates                 12,774
*updates-modular           Fedora Modular 30 - x86_64 - Updates             145
```

Pour ajouter un dépôt, invoquez `dnf config-manager --add_repo URL`, où `URL` est l'URL complète du dépôt. Pour activer un dépôt, utilisez `dnf config-manager --set-enabled ID_DEPOT`.

De même, pour désactiver un dépôt, utilisez `dnf config-manager --set-disabled ID_DEPOT`. Dans les deux cas, `ID_DEPOT` est l'ID unique du dépôt, que vous pouvez obtenir en invoquant `dnf repolist`. Les dépôts ajoutés sont activés par défaut.

Les dépôts sont stockés dans des fichiers `.repo` dans le répertoire `/etc/yum.repos.d/` et utilisent exactement la même syntaxe que `yum`.

Zypper

`zypper` est l'outil de gestion de paquets utilisé sous SUSE Linux et OpenSUSE. Ses fonctionnalités sont similaires à celles de `apt` et `yum`, puisqu'il permet d'installer, de mettre à jour et de supprimer des paquets sur un système, avec une résolution automatique des dépendances.

Mettre à jour l'index des paquets

Tout comme d'autres outils de gestion de paquets, `zypper` fonctionne avec des dépôts contenant des paquets et des métadonnées. Ces métadonnées doivent être rafraîchies de temps en temps, afin que le gestionnaire soit au courant des derniers paquets disponibles. Pour effectuer un rafraîchissement, tapez simplement :

```
# zypper refresh
Repository 'Non-OSS Repository' is up to date.
Repository 'Main Repository' is up to date.
```

```
Repository 'Main Update Repository' is up to date.
Repository 'Update Repository (Non-Oss)' is up to date.
All repositories have been refreshed.
```

`zypper` dispose d'une fonctionnalité de rafraîchissement automatique qui peut être activée individuellement pour chaque dépôt. Autrement dit, certains dépôts peuvent être rafraîchis automatiquement avant une requête ou l'installation d'un paquet, alors que d'autres devront être rafraîchis manuellement. Vous apprendrez bientôt comment vous servir de cette fonctionnalité.

Rechercher des paquets

Pour rechercher un paquet, utilisez la commande `search` (ou `se`) suivie du nom du paquet :

```
# zypper se gnumeric
Loading repository data...
Reading installed packages...

S | Name           | Summary                           | Type
--+----------------+-----------------------------------+--------
  | gnumeric       | Spreadsheet Application           | package
  | gnumeric-devel | Spreadsheet Application           | package
  | gnumeric-doc   | Documentation files for Gnumeric  | package
  | gnumeric-lang  | Translations for package gnumeric | package
```

La commande `search` peut également être utilisée pour obtenir une liste de tous les paquets installés sur le système. Pour ce faire, utilisez l'option `-i` sans nom de paquet, comme dans `zypper se -i`.

Pour voir si un paquet spécifique est installé, ajoutez le nom du paquet à la commande ci-dessus. Par exemple, la requête suivante va rechercher parmi les paquets installés tous ceux qui contiennent "firefox" dans leur nom :

```
# zypper se -i firefox
Loading repository data...
Reading installed packages...

S | Name                                | Summary                 | Type
--+-------------------------------------+-------------------------+--------
i | MozillaFirefox                      | Mozilla Firefox Web B-> | package
i | MozillaFirefox-branding-openSUSE    | openSUSE branding of -> | package
i | MozillaFirefox-translations-common | Common translations f-> | package
```

Pour rechercher uniquement parmi les paquets *non installés*, ajoutez l'option `-u` à l'opérateur `se`.

Installer, mettre à jour et supprimer des paquets

Pour installer un paquet logiciel, utilisez la commande `install` (ou `in`) suivie du nom du paquet. Comme ceci :

```
# zypper in unrar
zypper in unrar
Loading repository data...
Reading installed packages...
Resolving package dependencies...

The following NEW package is going to be installed:
  unrar

1 new package to install.
Overall download size: 141.2 KiB. Already cached: 0 B. After the operation, additional 301.6
KiB will be used.
Continue? [y/n/v/...? shows all options] (y): y
Retrieving package unrar-5.7.5-lp151.1.1.x86_64
                                    (1/1), 141.2 KiB (301.6 KiB unpacked)
Retrieving: unrar-5.7.5-lp151.1.1.x86_64.rpm .....................[done]
Checking for file conflicts: .....................................[done]
(1/1) Installing: unrar-5.7.5-lp151.1.1.x86_64 ...................[done]
```

`zypper` peut également être utilisé pour installer un paquet RPM depuis le disque local tout en essayant de satisfaire ses dépendances avec les paquets en provenance des dépôts. Pour ce faire, il suffit de fournir le chemin complet du paquet au lieu d'un nom de paquet, comme `zypper in /home/john/nouveaupaquet.rpm`.

Pour mettre à jour les paquets installés sur le système, utilisez `zypper update`. Comme pour le processus d'installation, cette opération va afficher une liste de paquets à installer/mettre à jour avant de vous demander si vous voulez continuer.

Si vous souhaitez seulement afficher la liste des mises à jour disponibles sans rien installer, vous pouvez utiliser `zypper list-updates`.

Pour supprimer un paquet, utilisez la commande `remove` (ou `rm`) suivie du nom du paquet :

```
# zypper rm unrar
Loading repository data...
```

```
Reading installed packages...
Resolving package dependencies...

The following package is going to be REMOVED:
  unrar

1 package to remove.
After the operation, 301.6 KiB will be freed.
Continue? [y/n/v/...? shows all options] (y): y
(1/1) Removing unrar-5.7.5-lp151.1.1.x86_64 .....................[done]
```

Gardez à l'esprit que la suppression d'un paquet entraîne la suppression de tous les autres paquets qui en dépendent. Par exemple :

```
# zypper rm libgimp-2_0-0
Loading repository data...
Warning: No repositories defined. Operating only with the installed resolvables. Nothing can
be installed.
Reading installed packages...
Resolving package dependencies...

The following 6 packages are going to be REMOVED:
  gimp gimp-help gimp-lang gimp-plugins-python libgimp-2_0-0
  libgimpui-2_0-0

6 packages to remove.
After the operation, 98.0 MiB will be freed.
Continue? [y/n/v/...? shows all options] (y):
```

Savoir quel paquet fournit un fichier donné

Pour savoir quel paquet contient un fichier donné, utilisez la commande `search` suivie de l'option `--provides` et du nom du fichier (ou de son chemin complet). Par exemple, si vous voulez savoir quel paquet contient le fichier `libgimpmodule-2.0.so.0` dans `/usr/lib64/` vous invoquerez :

```
# zypper se --provides /usr/lib64/libgimpmodule-2.0.so.0
Loading repository data...
Reading installed packages...

S | Name          | Summary                                          | Type
--+---------------+--------------------------------------------------+--------
```

```
i | libgimp-2_0-0 | The GNU Image Manipulation Program - Libra-> | package
```

Obtenir des informations sur les paquets

Pour voir les métadonnées associées à un paquet, utilisez la commande `info` suivie du nom du paquet. Cela vous indiquera le dépôt d'origine, le nom du paquet, la version, l'architecture, le fabricant, la taille installée, s'il est installé ou non, le statut (s'il est à jour), le paquet source ainsi qu'une description.

```
# zypper info gimp
Loading repository data...
Reading installed packages...

Information for package gimp:
 ----------------------------
Repository     : Main Repository
Name           : gimp
Version        : 2.8.22-lp151.4.6
Arch           : x86_64
Vendor         : openSUSE
Installed Size : 29.1 MiB
Installed      : Yes (automatically)
Status         : up-to-date
Source package : gimp-2.8.22-lp151.4.6.src
Summary        : The GNU Image Manipulation Program
Description    :
    The GIMP is an image composition and editing program, which can be
    used for creating logos and other graphics for Web pages. The GIMP
    offers many tools and filters, and provides a large image
    manipulation toolbox, including channel operations and layers,
    effects, subpixel imaging and antialiasing, and conversions, together
    with multilevel undo. The GIMP offers a scripting facility, but many
    of the included scripts rely on fonts that we cannot distribute.
```

Gérer les dépôts de logiciels

`zypper` peut également être utilisé pour gérer les dépôts de logiciels. Pour voir une liste de tous les dépôts enregistrés sur votre système, utilisez `zypper repos` :

```
# zypper repos
Repository priorities are without effect. All enabled repositories share the same priority.
```

```
#  | Alias                     | Name                                | Enabled | GPG Check |
Refresh
---+---------------------------+-------------------------------------+---------+-----------+
--------
 1 | openSUSE-Leap-15.1-1      | openSUSE-Leap-15.1-1                | No      | ----      |
----
 2 | repo-debug                | Debug Repository                    | No      | ----      |
----
 3 | repo-debug-non-oss        | Debug Repository (Non-OSS)          | No      | ----      |
----
 4 | repo-debug-update         | Update Repository (Debug)           | No      | ----      |
----
 5 | repo-debug-update-non-oss | Update Repository (Debug, Non-OSS) | No      | ----      |
----
 6 | repo-non-oss              | Non-OSS Repository                  | Yes     | (r ) Yes  |
Yes
 7 | repo-oss                  | Main Repository                     | Yes     | (r ) Yes  |
Yes
 8 | repo-source               | Source Repository                   | No      | ----      |
----
 9 | repo-source-non-oss       | Source Repository (Non-OSS)         | No      | ----      |
----
10 | repo-update               | Main Update Repository              | Yes     | (r ) Yes  |
Yes
11 | repo-update-non-oss       | Update Repository (Non-Oss)         | Yes     | (r ) Yes  |
Yes
```

Dans la colonne `Enabled`, notez que certains dépôts sont activés, alors que d'autres ne le sont pas. Vous pouvez changer cela avec la commande `modifyrepo` suivie de l'option `-e` (*enable*) ou `-d` (*disable*) et l'alias du dépôt en question (qui figure dans la deuxième colonne dans l'affichage ci-dessus).

```
# zypper modifyrepo -d repo-non-oss
Repository 'repo-non-oss' has been successfully disabled.

# zypper modifyrepo -e repo-non-oss
Repository 'repo-non-oss' has been successfully enabled.
```

Nous avons vu précédemment que `zypper` a une fonctionnalité de rafraîchissement automatique qui peut être activée individuellement pour chaque dépôt. Lorsqu'il est activé, ce paramètre va faire en sorte que `zypper` déclenche une opération de rafraîchissement (la même que l'exécution

de `zypper refresh`) avant d'interagir avec le dépôt spécifié. Ceci peut être contrôlé avec les options `-f` et `-F` de la commande `modifyrepo` :

```
# zypper modifyrepo -F repo-non-oss
Autorefresh has been disabled for repository 'repo-non-oss'.

# zypper modifyrepo -f repo-non-oss
Autorefresh has been enabled for repository 'repo-non-oss'.
```

Ajouter et supprimer des dépôts

Pour ajouter un nouveau dépôt logiciel pour `zypper`, utilisez la commande `addrepo` suivie de l'URL et du nom du dépôt, comme ci-dessous :

```
# zypper addrepo http://packman.inode.at/suse/openSUSE_Leap_15.1/ packman
Adding repository 'packman' .....................................[done]
Repository 'packman' successfully added

URI         : http://packman.inode.at/suse/openSUSE_Leap_15.1/
Enabled     : Yes
GPG Check   : Yes
Autorefresh : No
Priority    : 99 (default priority)

Repository priorities are without effect. All enabled repositories share the same priority.
```

Lorsque vous ajoutez un dépôt, vous pouvez activer le rafraîchissement automatique avec l'option `-f`. Les dépôts ajoutés sont activés par défaut, mais vous pouvez ajouter et désactiver un dépôt d'une traite en utilisant l'option `-d`.

Pour supprimer un dépôt, utilisez la commande `removerepo`, suivie du nom du dépôt (Alias). Pour supprimer le dépôt ajouté dans l'exemple ci-dessus, la commande serait :

```
# zypper removerepo packman
Removing repository 'packman' ...................................[done]
Repository 'packman' has been removed.
```

Exercices guidés

1. En utilisant `rpm` sur un système Red Hat Enterprise Linux, comment installeriez-vous le paquet `file-roller-3.28.1-2.el7.x86_64.rpm` en affichant une barre de progression pendant l'installation ?

2. En utilisant `rpm`, trouvez le paquet qui contient le fichier `/etc/redhat-release`.

3. Comment utiliseriez-vous `yum` pour vérifier les mises à jour concernant tous les paquets du système ?

4. En utilisant `zypper`, comment feriez-vous pour désactiver un dépôt appelé repo-extras ?

5. Si vous avez un fichier `.repo` qui décrit un nouveau dépôt, où devez-vous le ranger pour qu'il soit reconnu par DNF ?

Exercices d'approfondissement

1. Comment utiliseriez-vous `zypper` pour savoir quel paquet détient le fichier `/usr/sbin/swapon` ?

2. Comment peut-on obtenir une liste de tous les paquets installés sur le système en utilisant `dnf` ?

3. En utilisant `dnf`, quelle est la commande pour ajouter un dépôt disponible à l'adresse `https://www.example.url/home:reponame.repo` au système ?

4. Comment pouvez-vous utiliser `zypper` pour vérifier si le paquet `unzip` est installé ?

5. En utilisant `yum`, trouvez le paquet qui fournit le fichier `/bin/wget`.

Résumé

Dans cette leçon, vous avez appris à :

- Comment utiliser `rpm` pour installer, mettre à jour et supprimer des paquets.
- Comment utiliser `yum`, `zypper` et `dnf`.
- Comment obtenir des informations sur un paquet.
- Comment lister le contenu d'un paquet.
- Comment savoir de quel paquet provient un fichier.
- Comment lister, ajouter, supprimer, activer ou désactiver des dépôts de logiciels.

Les commandes suivantes ont été abordées :

- `rpm`
- `yum`
- `dnf`
- `zypper`

Réponses aux exercices guidés

1. En utilisant `rpm` sur un système Red Hat Enterprise Linux, comment installeriez-vous le paquet `file-roller-3.28.1-2.el7.x86_64.rpm` en affichant une barre de progression pendant l'installation ?

   Utilisez l'option `-i` pour installer un paquet et l'option `-h` pour activer les signes dièse # (*hash marks*) qui montrent la progression de l'installation. La réponse est donc : `rpm -ih file-roller-3.28.1-2.el7.x86_64.rpm`.

2. En utilisant `rpm`, trouvez le paquet qui contient le fichier `/etc/redhat-release`.

   Vous demandez des informations sur un fichier, vous devez donc utiliser l'option `-qf` : `rpm -qf /etc/redhat-release`.

3. Comment utiliseriez-vous `yum` pour vérifier les mises à jour concernant tous les paquets du système ?

   Utilisez la commande `check-update` *sans* nom de paquet : `yum check-update`.

4. En utilisant `zypper`, comment feriez-vous pour désactiver un dépôt appelé repo-extras ?

   Utilisez la commande `modifyrepo` pour modifier les paramètres d'un dépôt, et l'option `-d` pour le désactiver : `zypper modifyrepo -d repo-extras`.

5. Si vous avez un fichier `.repo` qui décrit un nouveau dépôt, où devez-vous le ranger pour qu'il soit reconnu par DNF ?

   Les fichiers `.repo` pour DNF doivent être rangés au même endroit que ceux utilisés par YUM, dans `/etc/yum.repos.d/`.

Réponses aux exercices d'approfondissement

1. Comment utiliseriez-vous `zypper` pour savoir quel paquet détient le fichier `/usr/sbin/swapon` ?

   Utilisez la commande `se` (`search`) et l'option `--provides` : `zypper se --provides /usr/sbin/swapon`.

2. Comment peut-on obtenir une liste de tous les paquets installés sur le système en utilisant `dnf` ?

   Utilisez la commande `list` suivie de l'option `--installed` : `dnf list --installed`.

3. En utilisant `dnf`, quelle est la commande pour ajouter un dépôt disponible à l'adresse `https://www.example.url/home:reponame.repo` au système ?

   Manipuler les dépôts reviens à gérer la configuration, il faut donc utiliser la commande `config-manager` et l'option `--add-repo` : `dnf config-manager --add-repo https://www.example.url/home:reponame.repo`.

4. Comment pouvez-vous utiliser `zypper` pour vérifier si le paquet `unzip` est installé ?

   Vous devez effectuer une recherche (`se`) sur les paquets installés (`-i`) : `zypper se -i unzip`.

5. En utilisant `yum`, trouvez le paquet qui fournit le fichier `/bin/wget`.

   Pour savoir quel paquet fournit un fichier, utilisez `whatprovides` et le nom du fichier : `yum whatprovides /bin/wget`.

102.6 Linux en tant que système virtuel hébergé

Référence aux objectifs de LPI

LPIC-1, Exam 101, Objective 102.6

Valeur

1

Domaines de connaissance les plus importants

- Compréhension des concepts généraux concernant la virtualisation et les conteneurs
- Compréhension des éléments communs de virtualisation dans le Cloud IaaS (Infratruction as a Service), comme les instances de machines, les blocs de stockage et le réseau
- Compréhension des propriétés de configuration uniques d'un système Linux à changer en cas de clone ou d'utilisation d'un modèle
- Compréhension de la manière dont les images système sont utilisées pour déployer les machines virtuelles, instances de machines dans le cloud ou les conteneurs
- Compréhension des extensions Linux permettant d'intégrer Linux à un outil de virtualisation
- Connaissance de base de cloud-init

Liste partielle de termes, fichiers et utilitaires utilisés pour cet objectif

- Machine virtuelles
- Conteneur Linux
- Conteneur d'application
- Pilotes invité (guest drivers)
- Clés SSH machine

- D-Bus machine id

102.6 Leçon 1

Certification :	LPIC-1
Version :	5.0
Thème :	102 Installation de Linux et gestion des paquets
Objectif :	102.6 Linux en tant que système virtualisé
Leçon :	1 sur 1

Introduction

Un des principaux atouts de Linux est sa polyvalence. Un aspect de cette polyvalence est la possibilité d'utiliser Linux pour héberger d'autres systèmes d'exploitation ou des applications individuelles dans un environnement complètement isolé et sécurisé. Cette leçon se concentre sur les concepts de virtualisation et de technologies de conteneurs ainsi que sur certains détails techniques à prendre en compte lors du déploiement d'une machine virtuelle sur une plate-forme de *cloud computing*.

Aperçu de la virtualisation

La virtualisation est une technologie qui permet à une plate-forme logicielle appelée *hyperviseur* d'exécuter des processus qui renferment un système informatique intégralement émulé. L'hyperviseur est chargé de gérer les ressources physiques du matériel qui pourront être utilisées par les différentes machines virtuelles. Ces machines virtuelles sont appelées des systèmes *virtualisés* (ou *invités*) de l'hyperviseur. Dans une machine virtuelle, bon nombre des caractéristiques d'un ordinateur physique sont émulées sous forme logicielle, comme le BIOS du système et les contrôleurs de disques durs. Une machine virtuelle utilise souvent des images disque qui sont stockées sous forme de fichiers individuels, elle aura accès à la RAM et au

processeur de la machine hôte par le biais du logiciel hyperviseur. L'hyperviseur répartit l'accès aux ressources matérielles du système hôte entre les invités, ce qui permet à plusieurs systèmes d'exploitation de fonctionner sur un seul système hôte.

Voici les hyperviseurs les plus courants sous Linux :

Xen

Xen est un hyperviseur *open source* de type 1, ce qui signifie qu'il ne dépend pas d'un système d'exploitation sous-jacent pour fonctionner. Un hyperviseur de ce type est connu sous le nom d'hyperviseur *bare metal* (natif), étant donné que l'ordinateur peut démarrer directement l'hyperviseur.

KVM

KVM (*Kernel-based Virtual Machine*) est un module du noyau Linux pour la virtualisation. KVM est à la fois un hyperviseur de type 1 et 2 ; même s'il a besoin d'un système d'exploitation Linux de base pour fonctionner, il est capable d'assumer parfaitement son rôle d'hyperviseur en s'intégrant à une installation Linux en cours d'exécution. Les machines virtuelles déployées avec KVM utilisent le démon `libvirt` et les logiciels associés pour être créées et gérées.

VirtualBox

Une application de bureau populaire qui permet de créer et de gérer facilement des machines virtuelles. Oracle VM VirtualBox est multiplateforme et fonctionne sous Linux, macOS et Microsoft Windows. Comme VirtualBox a besoin d'un système d'exploitation sous-jacent pour fonctionner, il s'agit d'un hyperviseur de type 2.

Certains hyperviseurs permettent la réaffectation dynamique d'une machine virtuelle. Le fait de transférer une machine virtuelle d'un hyperviseur vers un autre est appelé une *migration*, les techniques utilisées diffèrent selon les différents types d'hyperviseur. Certaines migrations peuvent être effectuées uniquement lorsque le système invité est complètement à l'arrêt, tandis que d'autres peuvent se faire pendant que l'invité est en cours d'exécution (ce que l'on appelle une *migration à chaud*). Ces techniques peuvent s'avérer utiles pendant les fenêtres de maintenance des hyperviseurs ou pour la résilience du système, lorsqu'un hyperviseur tombe en panne et que le système virtualisé peut être déplacé vers un hyperviseur en état de marche.

Types de machines virtuelles

On distingue trois types de machines virtuelles, les systèmes invités *pleinement virtualisés*, les systèmes *paravirtualisés* et les systèmes *hybrides*.

VM pleinement virtualisée

Toutes les instructions qu'un système d'exploitation invité est censé exécuter doivent pouvoir

tourner dans une installation entièrement virtualisée du système d'exploitation. La raison en est qu'aucun pilote logiciel supplémentaire n'est installé dans l'invité pour traduire les instructions en matériel simulé ou réel. Un invité entièrement virtualisé est un système dans lequel l'invité (ou la *HardwareVM*) ne sait pas qu'il s'agit d'une instance de machine virtuelle en cours d'exécution. Pour que ce type de virtualisation puisse fonctionner sur du matériel x86, les extensions CPU Intel VT-x ou AMD-V doivent être activées sur le système sur lequel l'hyperviseur est installé. Cette opération peut être effectuée à partir du menu de configuration du BIOS ou de l'UEFI.

VM paravirtualisée

Une machine virtuelle paravirtualisée (ou PVM) est un système invité dont l'OS est conscient qu'il s'agit d'une instance de machine virtuelle en cours d'exécution. Ces types de systèmes invités utilisent un noyau modifié et des pilotes spécifiques (appelés *pilotes invités*) qui aident le système d'exploitation invité à utiliser les ressources logicielles et matérielles de l'hyperviseur. Les performances d'un invité paravirtualisé sont souvent meilleures que celles de l'invité entièrement virtualisé grâce aux avantages que lui procurent ces pilotes logiciels.

VM hybride

La paravirtualisation et la virtualisation complète peuvent être combinées pour permettre aux systèmes d'exploitation non modifiés de bénéficier de performances d'E/S quasi natives en utilisant des pilotes paravirtualisés sur des systèmes d'exploitation entièrement virtualisés. Les pilotes paravirtualisés contiennent des pilotes de périphériques de stockage et de réseau avec des performances améliorées en matière d'E/S de disque et de réseau.

Les plateformes de virtualisation fournissent souvent des pilotes invités sous forme de paquets pour les systèmes d'exploitation virtualisés. KVM utilise les pilotes du projet *Virtio* tandis qu'Oracle VM VirtualBox utilise les *Guest Extensions* (Additions Invité) disponibles à partir d'un fichier image ISO CD-ROM téléchargeable.

Exemple de machine virtuelle `libvirt`

Nous allons voir de plus près un exemple de machine virtuelle qui est gérée par `libvirt` et qui utilise l'hyperviseur KVM. Une machine virtuelle est souvent composée d'un groupe de fichiers, principalement un fichier XML qui *définit* la machine virtuelle (comme sa configuration matérielle, sa connectivité réseau, ses capacités d'affichage, etc.) et un fichier image disque associé qui contient l'installation du système d'exploitation et de ses logiciels.

Commençons par jeter un œil sur un exemple de fichier de configuration XML pour une machine virtuelle et son environnement réseau :

```
$ ls /etc/libvirt/qemu
```

```
total 24
drwxr-xr-x 3 root root 4096 Oct 29 17:48 networks
-rw------- 1 root root 5667 Jun 29 17:17 rhel8.0.xml
```

NOTE

La partie `qemu` du chemin vers le répertoire fait référence au logiciel sous-jacent sur lequel reposent les machines virtuelles basées sur KVM. Le projet QEMU fournit un logiciel permettant à l'hyperviseur d'émuler les périphériques matériels que la machine virtuelle utilisera, tels que les contrôleurs de disque, l'accès au CPU de l'hôte, l'émulation de la carte réseau, etc.

Notez qu'il y a un répertoire nommé `networks`. Ce répertoire contient des fichiers de définition (également au format XML) qui créent des configurations réseau que les machines virtuelles pourront utiliser. Cet hyperviseur n'utilise qu'un seul réseau, il n'y a donc qu'un seul fichier de définition qui contient une configuration pour un segment de réseau virtuel que ces systèmes utiliseront.

```
$ ls -l /etc/libvirt/qemu/networks/
total 8
drwxr-xr-x 2 root root 4096 Jun 29 17:15 autostart
-rw------- 1 root root  576 Jun 28 16:39 default.xml
$ sudo cat /etc/libvirt/qemu/networks/default.xml
<!--
WARNING: THIS IS AN AUTO-GENERATED FILE. CHANGES TO IT ARE LIKELY TO BE
OVERWRITTEN AND LOST. Changes to this xml configuration should be made using:
  virsh net-edit default
or other application using the libvirt API.
-->

<network>
  <name>default</name>
  <uuid>55ab064f-62f8-49d3-8d25-8ef36a524344</uuid>
  <forward mode='nat'/>
  <bridge name='virbr0' stp='on' delay='0'/>
  <mac address='52:54:00:b8:e0:15'/>
  <ip address='192.168.122.1' netmask='255.255.255.0'>
    <dhcp>
      <range start='192.168.122.2' end='192.168.122.254'/>
    </dhcp>
  </ip>
</network>
```

Cette définition comprend un réseau privé de classe C et un périphérique matériel émulé faisant

office de routeur pour ce réseau. Il y a également une plage d'adresses IP que l'hyperviseur utilisera avec une implémentation de serveur DHCP et qui pourront être attribuées aux machines virtuelles utilisant ce réseau. Cette configuration réseau utilise également la translation d'adresse réseau (NAT) pour transférer les paquets vers d'autres réseaux, comme le réseau local de l'hyperviseur.

Nous allons maintenant porter notre attention sur un fichier de définition de machine virtuelle Red Hat Enterprise Linux 8. (Les sections à retenir sont en caractères gras) :

```
$ sudo cat /etc/libvirt/qemu/rhel8.0.xml
<!--
WARNING: THIS IS AN AUTO-GENERATED FILE. CHANGES TO IT ARE LIKELY TO BE
OVERWRITTEN AND LOST. Changes to this xml configuration should be made using:
  virsh edit rhel8.0
or other application using the libvirt API.
-->

<domain type='kvm'>
  <name>rhel8.0</name>
  <uuid>fadd8c5d-c5e1-410e-b425-30da7598d0f6</uuid>
  <metadata>
    <libosinfo:libosinfo xmlns:libosinfo="http://libosinfo.org/xmlns/libvirt/domain/1.0">
      <libosinfo:os id="http://redhat.com/rhel/8.0"/>
    </libosinfo:libosinfo>
  </metadata>
  <memory unit='KiB'>4194304</memory>
  <currentMemory unit='KiB'>4194304</currentMemory>
  <vcpu placement='static'>2</vcpu>
  <os>
    <type arch='x86_64' machine='pc-q35-3.1'>hvm</type>
    <boot dev='hd'/>
  </os>
  <features>
    <acpi/>
    <apic/>
    <vmport state='off'/>
  </features>
  <cpu mode='host-model' check='partial'>
    <model fallback='allow'/>
  </cpu>
  <clock offset='utc'>
    <timer name='rtc' tickpolicy='catchup'/>
    <timer name='pit' tickpolicy='delay'/>
```

```
    <timer name='hpet' present='no'/>
  </clock>
  <on_poweroff>destroy</on_poweroff>
  <on_reboot>restart</on_reboot>
  <on_crash>destroy</on_crash>
  <pm>
    <suspend-to-mem enabled='no'/>
    <suspend-to-disk enabled='no'/>
  </pm>
  <devices>
    <emulator>/usr/bin/qemu-system-x86_64</emulator>
    <disk type='file' device='disk'>
      <driver name='qemu' type='qcow2'/>
      <source file='/var/lib/libvirt/images/rhel8'/>
      <target dev='vda' bus='virtio'/>
      <address type='pci' domain='0x0000' bus='0x04' slot='0x00' function='0x0'/>
    </disk>
    <controller type='usb' index='0' model='qemu-xhci' ports='15'>
      <address type='pci' domain='0x0000' bus='0x02' slot='0x00' function='0x0'/>
    </controller>
    <controller type='sata' index='0'>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x1f' function='0x2'/>
    </controller>
    <controller type='pci' index='0' model='pcie-root'/>
    <controller type='pci' index='1' model='pcie-root-port'>
      <model name='pcie-root-port'/>
      <target chassis='1' port='0x10'/>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x02' function='0x0'
multifunction='on'/>
    </controller>
    <controller type='pci' index='2' model='pcie-root-port'>
      <model name='pcie-root-port'/>
      <target chassis='2' port='0x11'/>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x02' function='0x1'/>
    </controller>
    <controller type='pci' index='3' model='pcie-root-port'>
      <model name='pcie-root-port'/>
      <target chassis='3' port='0x12'/>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x02' function='0x2'/>
    </controller>
    <controller type='pci' index='4' model='pcie-root-port'>
      <model name='pcie-root-port'/>
      <target chassis='4' port='0x13'/>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x02' function='0x3'/>
```

```
    </controller>
    <controller type='pci' index='5' model='pcie-root-port'>
      <model name='pcie-root-port'/>
      <target chassis='5' port='0x14'/>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x02' function='0x4'/>
    </controller>
    <controller type='pci' index='6' model='pcie-root-port'>
      <model name='pcie-root-port'/>
      <target chassis='6' port='0x15'/>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x02' function='0x5'/>
    </controller>
    <controller type='pci' index='7' model='pcie-root-port'>
      <model name='pcie-root-port'/>
      <target chassis='7' port='0x16'/>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x02' function='0x6'/>
    </controller>
    <controller type='virtio-serial' index='0'>
      <address type='pci' domain='0x0000' bus='0x03' slot='0x00' function='0x0'/>
    </controller>
    <interface type='network'>
      <mac address='52:54:00:50:a7:18'/>
      <source network='default'/>
      <model type='virtio'/>
      <address type='pci' domain='0x0000' bus='0x01' slot='0x00' function='0x0'/>
    </interface>
    <serial type='pty'>
      <target type='isa-serial' port='0'>
        <model name='isa-serial'/>
      </target>
    </serial>
    <console type='pty'>
      <target type='serial' port='0'/>
    </console>
    <channel type='unix'>
      <target type='virtio' name='org.qemu.guest_agent.0'/>
      <address type='virtio-serial' controller='0' bus='0' port='1'/>
    </channel>
    <channel type='spicevmc'>
      <target type='virtio' name='com.redhat.spice.0'/>
      <address type='virtio-serial' controller='0' bus='0' port='2'/>
    </channel>
    <input type='tablet' bus='usb'>
      <address type='usb' bus='0' port='1'/>
    </input>
```

```
    <input type='mouse' bus='ps2'/>
    <input type='keyboard' bus='ps2'/>
    <graphics type='spice' autoport='yes'>
      <listen type='address'/>
      <image compression='off'/>
    </graphics>
    <sound model='ich9'>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x1b' function='0x0'/>
    </sound>
    <video>
      <model type='virtio' heads='1' primary='yes'/>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x01' function='0x0'/>
    </video>
    <redirdev bus='usb' type='spicevmc'>
      <address type='usb' bus='0' port='2'/>
    </redirdev>
    <redirdev bus='usb' type='spicevmc'>
      <address type='usb' bus='0' port='3'/>
    </redirdev>
    <memballoon model='virtio'>
      <address type='pci' domain='0x0000' bus='0x05' slot='0x00' function='0x0'/>
    </memballoon>
    <rng model='virtio'>
      <backend model='random'>/dev/urandom</backend>
      <address type='pci' domain='0x0000' bus='0x06' slot='0x00' function='0x0'/>
    </rng>
  </devices>
</domain>
```

Ce fichier définit un certain nombre de paramètres matériels qui sont utilisés par ce système virtualisé, comme la quantité de RAM qui lui sera attribuée, le nombre de cœurs de CPU de l'hyperviseur auxquels l'invité aura accès, le fichier image disque qui est associé à cet invité (sous la rubrique `disk`), ses capacités d'affichage (via le protocole SPICE) et l'accès de l'invité aux périphériques USB ainsi qu'aux entrées émulées du clavier et de la souris.

Exemple de stockage sur disque d'une machine virtuelle

L'image disque de cette machine virtuelle se situe dans `/var/lib/libvirt/images/rhel8`. Voici l'image disque en question sur cet hyperviseur :

```
$ sudo ls -lh /var/lib/libvirt/images/rhel8
-rw------- 1 root root 5.5G Oct 25 15:57 /var/lib/libvirt/images/rhel8
```

La taille actuelle de cette image disque ne prend que 5,5 Go d'espace sur l'hyperviseur. Cependant, le système d'exploitation dans l'invité voit un disque de 23,3 Go, comme le met en évidence la sortie de la commande suivante dans la machine virtuelle en cours d'exécution :

```
$ lsblk
NAME          MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
vda           252:0    0 23.3G  0 disk
├─vda1        252:1    0    1G  0 part /boot
└─vda2        252:2    0 22.3G  0 part
  ├─rhel-root 253:0    0   20G  0 lvm  /
  └─rhel-swap 253:1    0  2.3G  0 lvm  [SWAP]
```

Cela est dû au type de provisionnement de disque utilisé pour ce système virtualisé. Il existe plusieurs types d'images disque qu'une machine virtuelle peut utiliser, mais les deux principaux types sont les suivants :

COW

Le *copy-on-write* (ou copie sur écriture, également appelé *thin provisioning* ou allocation dynamique) est une méthode selon laquelle un fichier disque est créé avec une limite de taille supérieure prédéfinie. La taille de l'image disque n'augmente que lorsque de nouvelles données sont écrites sur le disque. Comme dans l'exemple précédent, le système d'exploitation virtualisé voit la limite prédéfinie du disque de 23,3 Go, mais n'a écrit que 5,5 Go de données sur le fichier du disque. Le format d'image disque utilisé pour la machine virtuelle dans l'exemple est `qcow2`, qui est un format de fichier QEMU COW.

RAW

Un type de disque *raw* ou brut est un fichier dont tout l'espace est pré-alloué. Par exemple, un fichier image disque brut de 10 Go consomme 10 Go d'espace disque réel sur l'hyperviseur. Ce type de disque présente un avantage en termes de performances, étant donné que tout l'espace disque nécessaire existe déjà, de sorte que l'hyperviseur sous-jacent peut simplement écrire des données sur le disque sans avoir à surveiller l'image disque pour s'assurer qu'elle n'a pas encore atteint sa limite et sans avoir à étendre la taille du fichier au fur et à mesure que de nouvelles données y sont écrites.

Il existe d'autres plateformes de gestion de la virtualisation, telles que *Red Hat Enterprise Virtualization* et *oVirt*, qui peuvent utiliser des disques physiques pour servir d'emplacements de stockage pour le système d'exploitation d'une machine virtuelle. Ces systèmes peuvent utiliser des périphériques de stockage SAN (*Storage Area Network*) ou NAS (*Network Attached Storage*) pour y écrire leurs données, l'hyperviseur garde la trace des emplacements de stockage appartenant à chaque machine virtuelle. Ces systèmes de stockage peuvent utiliser des technologies telles que la

gestion des volumes logiques (LVM) pour augmenter ou réduire la taille du stockage sur disque d'une machine virtuelle selon les besoins et pour faciliter la création et la gestion des instantanés de stockage.

Travailler avec des modèles de machines virtuelles

Étant donné que les machines virtuelles ne sont généralement que des fichiers exécutés sur un hyperviseur, il est facile de créer des *modèles* (ou *templates*) qui peuvent être personnalisés pour des scénarios de déploiement particuliers. Souvent, une machine virtuelle aura une installation de base du système d'exploitation et certains paramètres de configuration d'authentification préconfigurés pour faciliter les déploiements ultérieurs du système. Cela permet de réduire le temps nécessaire à la construction d'un nouveau système en réduisant la quantité de tâches répétitives, telles que l'installation des paquets de base et la configuration des paramètres régionaux.

Ce modèle de machine virtuelle pourra ensuite être copié vers un nouveau système invité. Dans ce cas, le nouveau système virtualisé sera renommé, une nouvelle adresse MAC sera générée pour son interface réseau et d'autres modifications pourront être apportées en fonction de l'utilisation prévue.

L'identifiant unique D-Bus machine-id

La plupart des systèmes Linux utilisent un numéro d'identification de la machine généré au moment de l'installation, appelé *D-Bus machine ID*. Cependant, si une machine virtuelle est *clonée* en vue d'être utilisée comme modèle pour d'autres installations de machines virtuelles, un nouvel identifiant unique D-Bus devra être créé pour garantir que les ressources système de l'hyperviseur sont dirigées vers le système invité approprié.

La commande suivante peut être utilisée pour valider qu'un identifiant unique D-Bus existe pour le système en cours d'exécution :

```
$ dbus-uuidgen --ensure
```

Si aucun message d'erreur ne s'affiche, cela signifie qu'un ID existe pour le système. Pour afficher l'ID D-Bus en vigueur, exécutez la commande suivante :

```
$ dbus-uuidgen --get
17f2e0698e844e31b12ccd3f9aa4d94a
```

La chaîne de caractères qui s'affiche est le numéro d'identification actuel. Deux systèmes Linux

tournant sur un même hyperviseur ne doivent pas avoir le même identifiant de machine.

L'ID D-Bus de la machine se trouve dans `/var/lib/dbus/machine-id` avec un lien symbolique vers `/etc/machine-id`. Il est déconseillé de modifier ce numéro d'identification sur un système en cours, étant donné que cela entraînerait une instabilité du système et une série de pannes. Au cas où deux machines virtuelles auraient le même ID de machine, suivez la procédure ci-dessous pour en générer un nouveau :

```
$ sudo rm -f /etc/machine-id
$ sudo dbus-uuidgen --ensure=/etc/machine-id
```

Au cas où `/var/lib/dbus/machine-id` ne serait pas un lien symbolique vers `/etc/machine-id`, il faudra supprimer `/var/lib/dbus/machine-id`.

Déployer des machines virtuelles dans le cloud

Il existe une multitude de fournisseurs IaaS (*Infrastructure as a Service*) qui font tourner des systèmes d'hyperviseurs et qui peuvent déployer des images de systèmes virtualisés pour une organisation. Pratiquement tous ces fournisseurs disposent d'outils qui permettent à un administrateur de construire, déployer et configurer des machines virtuelles personnalisées basées sur une panoplie de distributions Linux. Nombre de ces entreprises disposent également de systèmes qui permettent le déploiement et la migration de machines virtuelles construites au sein de l'organisation d'un client.

Le déploiement d'un système Linux dans un environnement IaaS doit prendre en compte certains éléments clés dont l'administrateur devra tenir compte :

Instances

Beaucoup de fournisseurs de cloud facturent des tarifs d'utilisation basés sur des "instances de calcul", c'est-à-dire la quantité de temps CPU que votre infrastructure cloud utilisera. Une planification minutieuse du temps de traitement dont les applications auront réellement besoin permettra de maîtriser les coûts d'une solution cloud.

Les instances désignent tout aussi bien le nombre de machines virtuelles provisionnées dans un environnement cloud. Là encore, le nombre d'instances exécutées en même temps influera sur le temps CPU global facturé à l'organisation.

Stockage en mode bloc

Les fournisseurs de cloud proposent également différents niveaux de stockage en bloc qu'une organisation pourra utiliser. Certaines offres sont simplement destinées à servir de stockage

réseau basé sur le web pour les fichiers, alors que d'autres offres concernent le stockage externe pour une machine virtuelle provisionnée dans le cloud à utiliser pour héberger des fichiers.

Le coût de ces offres varie en fonction de la quantité de stockage utilisée et de la vitesse du stockage dans les centres de données du fournisseur. Un accès plus rapide au stockage coûte généralement plus cher, et inversement, les données "au repos" (comme c'est le cas pour le stockage d'archives) sont souvent très peu coûteuses.

Mise en réseau

L'un des principaux aspects à prendre en compte lorsqu'on travaille avec un fournisseur de solutions cloud est la manière dont le réseau virtuel sera configuré. Beaucoup de fournisseurs IaaS disposent d'une interface web qui permet de concevoir et de mettre en œuvre toutes sortes de configurations de routage, de sous-réseaux et de pare-feu. Certains vont même fournir des solutions DNS afin que des noms de domaine pleinement qualifiés (FQDN) publiquement routables puissent être attribués à vos systèmes avec une ouverture frontale sur Internet. Il existe même des solutions "hybrides" qui permettent de connecter une infrastructure réseau existante sur site à une infrastructure cloud par le biais d'un VPN (*réseau privé virtuel*), reliant ainsi les deux infrastructures.

Accès sécurisé aux VM dans le cloud

La méthode la plus courante pour accéder à une VM distante sur une plate-forme *cloud* repose sur l'utilisation du logiciel OpenSSH. Un système Linux qui tourne dans le *cloud* dispose du serveur OpenSSH, tandis qu'un administrateur utilise un client OpenSSH avec des clés pré-partagées pour l'accès à distance.

Un administrateur exécutera la commande suivante :

```
$ ssh-keygen
```

et suivra les instructions pour créer une paire de clés SSH publique et privée. La clé privée reste sur le système local de l'administrateur (stockée dans `~/.ssh/`) et la clé publique est copiée sur le système *cloud* distant, exactement la même méthode que celle utilisée pour travailler avec des machines en réseau sur un LAN d'entreprise.

L'administrateur exécutera alors la commande suivante :

```
$ ssh-copy-id -i <public_key> user@cloud_server
```

Cette opération va copier la clé publique SSH de la paire de clés qui vient d'être générée vers le serveur *cloud* distant. La clé publique sera enregistrée dans le fichier `~/.ssh/authorized_keys` du serveur *cloud*, avec les permissions appropriées sur le fichier.

NOTE

S'il n'y a qu'un seul fichier de clé publique dans le répertoire `~/.ssh/`, alors l'option `-i` peut être omise, car la commande `ssh-copy-id` prendra par défaut le fichier de clé publique du répertoire (typiquement le fichier se terminant par l'extension `.pub`).

Certains fournisseurs de *cloud* vont générer automatiquement une paire de clés lorsqu'un nouveau système Linux est approvisionné. L'administrateur devra ensuite télécharger la clé privée du nouveau système depuis le fournisseur de *cloud* et la stocker sur son système local. Notez que les permissions pour les clés SSH doivent être `0600` pour une clé privée, et `0644` pour une clé publique.

Préconfiguration des systèmes cloud

Parmi les outils pratiques qui simplifient les déploiements de machines virtuelles dans le cloud, on trouve l'utilitaire `cloud-init`. Cette commande, ainsi que les fichiers de configuration correspondants et l'image de machine virtuelle prédéfinie, constitue une méthode agnostique en termes de fournisseur pour déployer une VM Linux sur une multitude de plate-formes IaaS. À l'aide de fichiers texte YAML (*YAML Ain't Markup Language*), un administrateur peut préconfigurer les paramètres réseau, la sélection des paquets logiciels, la configuration des clés SSH, la création des comptes utilisateurs, les paramètres régionaux ainsi qu'une myriade d'autres options qui permettent de déployer rapidement de nouveaux systèmes.

Lors du démarrage initial d'un nouveau système, `cloud-init` va lire les paramètres depuis les fichiers de configuration YAML pour les appliquer ensuite. Ce processus ne s'applique que lors de la configuration initiale d'un système et facilite le déploiement d'une série de nouveaux systèmes sur la plate-forme d'un fournisseur de *cloud computing*.

La syntaxe du fichier YAML utilisé avec `cloud-init` est appelée *cloud-config*. Voici un exemple de fichier `cloud-config` :

```
#cloud-config
timezone: Africa/Dar_es_Salaam
hostname: test-system

# Update the system when it first boots up
apt_update: true
apt_upgrade: true
```

```
# Install the Nginx web server
packages:
 - nginx
```

Notez que sur la première ligne, il n'y a pas d'espace entre le signe dièse (#) et le terme `cloud-config`.

NOTE `cloud-init` n'est pas réservé aux machines virtuelles. La suite d'outils `cloud-init` peut également être utilisée pour préconfigurer les conteneurs (par exemple les conteneurs Linux LXD) avant leur déploiement.

Les conteneurs

La technologie des conteneurs ressemble dans une certaine mesure à une machine virtuelle, dans le sens où l'on obtient un environnement isolé pour déployer facilement une application. Tandis qu'avec une machine virtuelle, un système entier est émulé, un conteneur utilise juste assez de ressources logicielles pour exécuter une application. Cette façon de procéder permet de réduire considérablement le gaspillage des ressources.

Les conteneurs offrent une plus grande flexibilité que les machines virtuelles. Un conteneur d'application peut être migré d'un hôte vers un autre, tout comme une machine virtuelle peut être migrée d'un hyperviseur vers un autre. Cependant, une machine virtuelle nécessite parfois d'être arrêtée avant la migration, alors qu'avec un conteneur, l'application reste en cours d'exécution pendant la migration. Les conteneurs facilitent également le déploiement de nouvelles versions d'applications en parallèle avec une version existante. Au fur et à mesure que les utilisateurs ferment leurs sessions avec des conteneurs en cours d'exécution, ces conteneurs peuvent être automatiquement retirés du système par le logiciel d'orchestration de conteneurs et remplacés par la nouvelle version, ce qui réduit les temps d'arrêt.

NOTE Il existe de nombreuses technologies de conteneurs disponibles pour Linux, telles que *Docker*, *Kubernetes*, *LXD/LXC*, *systemd-nspawn*, *OpenShift*, etc. La mise en œuvre détaillée d'un logiciel de conteneur dépasse le cadre de l'examen LPIC-1.

Les conteneurs utilisent le mécanisme des *control groups* (communément appelés *cgroups*) au sein du noyau Linux. Les *cgroups* permettent de partitionner les ressources système telles que la mémoire, le temps processeur ainsi que la bande passante disque et réseau pour une application individuelle. Un administrateur peut utiliser directement les *cgroups* pour fixer des limites de ressources système pour une application ou un groupe d'applications pouvant exister dans un seul *cgroup*. C'est essentiellement ce que font les logiciels de conteneurs pour l'administrateur, en plus de fournir des outils qui facilitent la gestion et le déploiement des *cgroups*.

NOTE

Actuellement, la connaissance des cgroups n'est pas nécessaire pour passer l'examen LPIC-1. Le concept des *cgroups* est mentionné ici afin que le candidat ait au moins quelques connaissances de base sur la manière dont une application est isolée dans un souci d'utilisation des ressources du système.

Exercices guidés

1. Quelles extensions de CPU sont nécessaires sur une plate-forme matérielle de type x86 qui fera fonctionner des systèmes invités entièrement virtualisés ?

2. Une installation de serveur de production nécessitant les performances les plus rapides utilisera probablement quel type de virtualisation ?

3. Deux machines virtuelles clonées à partir du même *template* et qui utilisent D-Bus se comportent de manière erratique. Elles ont toutes deux des noms d'hôtes et des paramètres réseau distincts. Quelle commande permet de déterminer si chacune des machines virtuelles dispose de son propre identifiant de machine D-Bus distinct ?

Exercices d'approfondissement

1. Exécutez la commande suivante pour voir si votre système a déjà des extensions de CPU activées pour exécuter une machine virtuelle (vos résultats peuvent varier en fonction de votre CPU) :

   ```
   grep --color -E "vmx|svm" /proc/cpuinfo
   ```

   Selon le résultat, vous pouvez avoir `vmx` en surbrillance (pour les CPU Intel VT-x) ou `svm` en surbrillance (pour les CPU AMD SVM). Si vous n'obtenez aucun résultat, consultez les instructions de votre BIOS ou de votre firmware UEFI pour savoir comment activer la virtualisation pour votre processeur.

2. Si votre processeur gère la virtualisation, consultez la documentation de votre distribution pour faire tourner un hyperviseur KVM.
   - Installez les paquets nécessaires pour faire tourner un hyperviseur KVM.
   - Si vous utilisez un environnement de bureau graphique, il est recommandé d'installer également l'application `virt-manager`, un frontal graphique qui peut être utilisé sur une installation KVM. Cela facilitera l'installation et la gestion des machines virtuelles.
   - Téléchargez une image ISO d'une distribution Linux de votre choix et, en suivant la documentation de cette distribution, créez une nouvelle machine virtuelle à l'aide de l'image ISO.

Résumé

Dans cette leçon, nous avons abordé les concepts de base des machines virtuelles et des conteneurs et comment ces technologies peuvent être utilisées avec Linux.

Nous avons décrit succinctement les commandes suivantes :

`dbus-uuidgen`

Utilisé pour vérifier et afficher l'identifiant D-Bus d'un système.

`ssh-keygen`

Utilisé pour générer une paire de clés SSH publique et privée pour accéder à des systèmes distants basés sur le *cloud*.

`ssh-copy-id`

Utilisé pour copier la clé publique SSH d'un système vers un système distant afin de faciliter l'authentification à distance.

`cloud-init`

Utilisé pour faciliter la configuration et le déploiement de machines virtuelles et de conteneurs dans un environnement *cloud*.

Réponses aux exercices guidés

1. Quelles extensions de CPU sont nécessaires sur une plate-forme matérielle de type x86 qui fera fonctionner des systèmes invités entièrement virtualisés ?

   VT-x pour les CPU Intel ou AMD-V pour les CPU AMD.

2. Une installation de serveur de production nécessitant les performances les plus rapides utilisera probablement quel type de virtualisation ?

   Un système d'exploitation qui utilise la paravirtualisation, comme Xen, pourra mieux utiliser les ressources matérielles à sa disposition en tant que système d'exploitation invité grâce à l'utilisation de pilotes logiciels conçus pour fonctionner avec l'hyperviseur.

3. Deux machines virtuelles clonées à partir du même *template* et qui utilisent D-Bus se comportent de manière erratique. Elles ont toutes deux des noms d'hôtes et des paramètres réseau distincts. Quelle commande permet de déterminer si chacune des machines virtuelles dispose de son propre identifiant de machine D-Bus distinct ?

   `dbus-uuidgen --get`

Réponses aux exercices d'approfondissement

1. Exécutez la commande suivante pour voir si votre système a déjà des extensions de CPU activées pour exécuter une machine virtuelle (vos résultats peuvent varier en fonction de votre CPU) :

   `grep --color -E "vmx|svm" /proc/cpuinfo`

   Selon le résultat, vous pouvez avoir `vmx` en surbrillance (pour les CPU Intel VT-x) ou `svm` en surbrillance (pour les CPU AMD SVM). Si vous n'obtenez aucun résultat, consultez les instructions de votre BIOS ou de votre firmware UEFI pour savoir comment activer la virtualisation pour votre processeur.

   Les résultats varieront en fonction du CPU que vous possédez. Voici un exemple de résultats obtenus sur un ordinateur équipé d'un CPU Intel avec des extensions de virtualisation activées dans le firmware UEFI :

   ```
   $ grep --color -E "vmx|svm" /proc/cpuinfo
   flags    : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36
   clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe syscall nx pdpe1gb rdtscp lm constant_tsc
   art arch_perfmon pebs bts rep_good nopl xtopology nonstop_tsc cpuid aperfmperf pni
   pclmulqdq dtes64 monitor ds_cpl vmx smx est tm2 ssse3 sdbg fma cx16 xtpr pdcm pcid sse4_1
   sse4_2 x2apic movbe popcnt tsc_deadline_timer aes xsave avx f16c rdrand lahf_lm abm
   3dnowprefetch cpuid_fault epb invpcid_single pti ssbd ibrs ibpb stibp tpr_shadow vnmi
   flexpriority ept vpid ept_ad fsgsbase tsc_adjust bmi1 hle avx2 smep bmi2 erms invpcid rtm
   mpx rdseed adx smap clflushopt intel_pt xsaveopt xsavec xgetbv1 xsaves dtherm ida arat
   pln pts hwp hwp_notify hwp_act_window hwp_epp md_clear flush_l1d
   ```

2. Si votre processeur gère la virtualisation, consultez la documentation de votre distribution pour faire tourner un hyperviseur KVM.
   - Installez les paquets nécessaires pour faire tourner un hyperviseur KVM.

     Cela variera en fonction de votre distribution, mais voici quelques pistes de réflexion :

     Ubuntu — https://help.ubuntu.com/lts/serverguide/libvirt.html

     Fedora — https://docs.fedoraproject.org/en-US/quick-docs/getting-started-with-virtualization/

     Arch Linux — https://wiki.archlinux.org/index.php/KVM

   - Si vous utilisez un environnement de bureau graphique, il est recommandé d'installer également l'application `virt-manager`, un frontal graphique qui peut être utilisé sur une

installation KVM. Cela facilitera l'installation et la gestion des machines virtuelles.

Là encore, cela varie selon la distribution. Un exemple avec Ubuntu ressemble à ceci :

```
$ sudo apt install virt-manager
```

- Téléchargez une image ISO d'une distribution Linux de votre choix et, en suivant la documentation de cette distribution, créez une nouvelle machine virtuelle à l'aide de l'image ISO.

  Cette tâche est gérée facilement par le paquet `virt-manager`. Cependant, une machine virtuelle peut être créée en ligne de commande à l'aide de la commande `virt-install`. Essayez les deux méthodes pour vous faire une idée de la façon dont les machines virtuelles sont déployées.

Linux Professional Institute

Thème 103 : Commandes GNU et Unix

103.1 Travail en ligne de commande

Référence aux objectifs de LPI

LPIC-1 version 5.0, Exam 101, Objective 103.1

Valeur

4

Domaines de connaissance les plus importants

- Utilisation de commandes ou de séquences de commandes pour réaliser des tâches simples en ligne de commande.
- Utilisation et modification de l'environnement du shell, en particulier la définition, l'export et le référencement des variables d'environnement.
- Utilisation et édition de l'historique des commandes.
- Exécution des commandes comprises ou non dans le chemin (path) par défaut.

Liste partielle de termes, fichiers et utilitaires utilisés pour cet objectif

- `bash`
- `echo`
- `env`
- `export`
- `pwd`
- `set`
- `unset`
- `type`
- `which`

- `man`
- `uname`
- `history`
- `.bash_history`
- Protection (quoting)

103.1 Leçon 1

Certification :	LPIC-1
Version :	5.0
Thème :	103 Commandes GNU et Unix
Objectif :	103.1 Travailler en ligne de commande
Leçon :	1 sur 2

Introduction

Les débutants dans le monde de l'administration Linux et de l'interpréteur de commandes (*shell*) Bash se sentent souvent un peu perdus sans le confort rassurant d'une interface graphique. Ils ont l'habitude de pouvoir accéder par un clic droit aux repères visuels et aux informations contextuelles que les gestionnaires de fichiers graphiques mettent à leur disposition. Il est donc important d'apprendre et de maîtriser l'ensemble relativement restreint d'outils en ligne de commande qui vous permettent d'accéder rapidement à toutes les données disponibles dans votre ancienne interface graphique - et bien plus encore.

Obtenir des informations sur le système

Lorsque vous contemplez le rectangle clignotant d'une invite de commande, votre première question sera probablement "Où suis-je ?". Ou, plus précisément, "Où suis-je en ce moment dans le système de fichiers Linux et si, admettons, je créais un nouveau fichier, où se trouverait-il ?". Ce que vous recherchez ici, c'est votre répertoire de travail actuel (*present work directory*), et la commande `pwd` vous dira ce que vous voulez savoir :

```
$ pwd
/home/frank
```

Imaginons que Frank est actuellement connecté au système et qu'il se trouve dans son répertoire personnel : `/home/frank/`. Si Frank crée un fichier vide en utilisant la commande `touch` sans spécifier d'autre emplacement dans le système de fichiers, le fichier sera créé dans `/home/frank/`. En affichant le contenu du répertoire avec `ls`, on peut voir ce nouveau fichier :

```
$ touch newfile
$ ls
newfile
```

En dehors de votre position dans le système de fichiers, vous voudrez souvent obtenir des informations sur le système Linux que vous utilisez. Il peut s'agir du numéro de version exact de votre distribution ou de la version du noyau Linux qui est actuellement chargée. L'outil `uname` est ce que vous recherchez ici. Et, plus exactement, `uname` en utilisant l'option `-a` ("all").

```
$ uname -a
Linux base 4.18.0-18-generic #19~18.04.1-Ubuntu SMP Fri Apr 5 10:22:13 UTC 2019 x86_64
x86_64 x86_64 GNU/Linux
```

Ici, `uname` indique que la machine de Frank a le noyau Linux version 4.18.0 installé et qu'elle utilise Ubuntu 18.04 sur un processeur 64 bits (`x86_64`).

Obtenir des informations sur les commandes

Vous trouverez souvent de la documentation qui traite des commandes Linux avec lesquelles vous n'êtes pas encore familiarisé. La ligne de commande elle-même offre toutes sortes d'informations utiles sur ce que font les commandes et comment les utiliser efficacement. Les informations les plus utiles se trouvent sans doute dans les nombreux fichiers du système `man`.

En règle générale, les développeurs de Linux rédigent des fichiers `man` et les distribuent avec les utilitaires qu'ils créent. Les fichiers `man` sont des documents bien structurés dont le contenu est subdivisé de manière intuitive par des titres de section types. En tapant `man` suivi du nom d'une commande, vous obtiendrez des informations comprenant le nom de la commande, un bref résumé de son utilisation, une description plus détaillée ainsi que des informations importantes sur l'historique et les licences. Voici un exemple :

```
$ man uname
```

```
UNAME(1)             User Commands            UNAME(1)
NAME
   uname - print system information
SYNOPSIS
   uname [OPTION]...
DESCRIPTION
   Print certain system information.  With no OPTION, same as -s.
   -a, --all
      print  all information, in the following order, except omit -p
      and -i if unknown:
   -s, --kernel-name
      print the kernel name
   -n, --nodename
      print the network node hostname
   -r, --kernel-release
      print the kernel release
   -v, --kernel-version
      print the kernel version
   -m, --machine
      print the machine hardware name
   -p, --processor
      print the processor type (non-portable)
   -i, --hardware-platform
      print the hardware platform (non-portable)
   -o, --operating-system
      print the operating system
   --help display this help and exit
   --version
      output version information and exit
AUTHOR
   Written by David MacKenzie.
REPORTING BUGS
   GNU coreutils online help: <http://www.gnu.org/software/coreutils/>
   Report uname translation bugs to
   <http://translationproject.org/team/>
COPYRIGHT
   Copyright©2017 Free Software Foundation, Inc. License  GPLv3+: GNU
   GPL version 3 or later <http://gnu.org/licenses/gpl.html>.
   This is free software: you are free to change and redistribute it.
   There is NO WARRANTY, to the extent permitted by law.
SEE ALSO
   arch(1), uname(2)
   Full documentation at: <http://www.gnu.org/software/coreutils/uname>
   or available locally via: info '(coreutils) uname invocation'
```

```
GNU coreutils 8.28       January 2018            UNAME(1)
```

`man` ne fonctionne que si vous lui fournissez un nom de commande précis. En revanche, si vous n'êtes pas sûr du nom de la commande que vous recherchez, vous pouvez utiliser la commande `apropos` pour effectuer une recherche dans les noms et les descriptions des pages `man`. En supposant, par exemple, que vous ne vous souvenez pas que c'est `uname` qui vous renseigne sur la version actuelle de votre noyau Linux, vous pouvez fournir le terme de recherche `kernel` à `apropos`. Vous obtiendrez probablement de nombreux résultats, mais ils devraient inclure ceux-ci :

```
$ apropos kernel
systemd-udevd-kernel.socket (8) - Device event managing daemon
uname (2)            - get name and information about current kernel
urandom (4)          - kernel random number source devices
```

Si vous n'avez pas besoin de la documentation complète d'une commande, vous pouvez obtenir rapidement des informations de base sur une commande en utilisant `type`. L'exemple qui suit utilise `type` pour interroger quatre commandes distinctes à la fois. Le résultat nous montre que `cp` ("copy") est un programme qui réside dans `/bin/cp` et que `kill` (changer l'état d'un processus en cours) est une primitive du *shell* — ce qui veut dire qu'il fait partie du *shell* Bash lui-même :

```
$ type uname cp kill which
uname is hashed (/bin/uname)
cp is /bin/cp
kill is a shell builtin
which is /usr/bin/which
```

Vous remarquerez que, en plus d'être une commande binaire normale comme `cp`, `uname` est également "haché" ("hashed"). C'est parce que Frank a récemment utilisé `uname` et, pour augmenter l'efficacité du système, il a été ajouté à une table de hachage pour le rendre plus accessible la prochaine fois que vous l'exécutez. S'il lançait `type uname` après le démarrage du système, Frank constaterait que `type` décrit à nouveau `uname` comme un binaire ordinaire.

NOTE Une façon plus rapide de vider la table de hachage est de lancer la commande `hash -d`.

Parfois — en particulier lorsque vous travaillez avec des scripts automatisés — vous aurez besoin d'une source d'information plus simple pour une commande. La commande `which` que notre précédente commande `type` a repérée pour nous ne retournera rien d'autre que le chemin absolu vers une commande. Cet exemple localise à la fois les commandes `uname` et `which`.

```
$ which uname which
/bin/uname
/usr/bin/which
```

NOTE Si vous voulez afficher des informations sur les primitives du *shell*, vous pouvez utiliser la commande `help`.

Utiliser l'historique des commandes

Il vous arrivera souvent de rechercher soigneusement l'usage approprié d'une commande et de l'exécuter avec succès, accompagnée d'une série passablement complexe d'options et d'arguments. Mais que se passe-t-il quelques semaines plus tard lorsque vous devez exécuter la même commande avec les mêmes options et les mêmes arguments mais que vous ne vous souvenez plus des détails ? Plutôt que de devoir recommencer vos recherches depuis le début, vous pourrez souvent retrouver la commande originale en utilisant `history`.

En tapant `history`, vous obtiendrez les commandes les plus récentes que vous avez exécutées, avec la plus récente qui apparaîtra en dernier. Vous pouvez facilement effectuer une recherche dans ces commandes en passant une chaîne donnée à la commande `grep`. Cet exemple recherchera n'importe quelle commande qui inclut le texte `bash_history` :

```
$ history | grep bash_history
1605 sudo find /home -name ".bash_history" | xargs grep sudo
```

Ici, une seule commande est renvoyée avec le numéro de séquence correspondant 1605.

Quant à `bash_history`, c'est en fait le nom d'un fichier caché que vous devriez trouver dans votre répertoire d'utilisateur. Puisque c'est un fichier caché (désigné comme tel par le point qui précède son nom de fichier), il ne sera visible qu'en affichant le contenu du répertoire en utilisant `ls` avec l'option `-a` :

```
$ ls /home/frank
newfile
$ ls -a /home/frank
.  ..  .bash_history  .bash_logout  .bashrc  .profile  .ssh  newfile
```

Que contient le fichier `.bash_history` ? Regardez par vous-même : vous y verrez des centaines et des centaines de vos commandes les plus récentes. Vous pourriez cependant être surpris de constater que certaines de vos commandes les *plus* récentes sont absentes dans cette liste. En effet,

bien que celles-ci soient instantanément ajoutées à la base de données dynamique `history`, les tout derniers ajouts à votre historique de commandes ne sont pas écrits dans le fichier `.bash_history` avant la fin de votre session.

Vous pouvez exploiter le contenu de `history` pour rendre votre expérience de la ligne de commande beaucoup plus rapide et efficace en utilisant les flèches haut et bas de votre clavier. En appuyant plusieurs fois sur la touche haut, la ligne de commande sera renseignée avec les commandes récentes. Lorsque vous arrivez à celle que vous souhaitez lancer une seconde fois, vous pouvez l'exécuter en appuyant sur Entrée. Il est ainsi facile de se rappeler et, le cas échéant, de modifier les commandes plusieurs fois de suite au cours d'une session dans le *shell*.

Exercices guidés

1. Utilisez le système `man` pour déterminer comment indiquer à `apropos` de fournir une commande brève afin qu'il affiche seulement un bref message d'utilisation et qu'il quitte ensuite.

2. Utilisez le système `man` pour déterminer quelle licence est attribuée à la commande `grep`.

Exercices d'approfondissement

1. Identifiez l'architecture matérielle et la version du noyau Linux utilisées sur votre ordinateur dans un format d'affichage facile à lire.

2. Affichez les vingt dernières lignes de la base de données dynamique `history` et du fichier `.bash_history` et comparez les deux résultats.

3. Utilisez l'outil `apropos` pour identifier la page `man` où vous trouverez la commande dont vous aurez besoin pour afficher la taille d'un périphérique bloc matériel connecté en octets plutôt qu'en mégaoctets ou en gigaoctets.

Résumé

Dans cette leçon, vous avez appris à :

- Comment obtenir des informations sur votre emplacement dans le système de fichiers et la pile logicielle de votre système d'exploitation.
- Comment trouver de l'aide pour l'utilisation des commandes.
- Comment identifier l'emplacement dans le système de fichiers et le type de commandes binaires.
- Comment trouver et réutiliser des commandes précédemment invoquées.

Les commandes suivantes ont été abordées dans cette leçon :

`pwd`
: Afficher le chemin d'accès au répertoire de travail actuel.

`uname`
: Afficher l'architecture matérielle de votre système, la version du noyau Linux, la distribution et la version de la distribution.

`man`
: Accéder aux fichiers d'aide qui documentent l'utilisation des commandes.

`type`
: Afficher l'emplacement dans le système de fichiers et le type pour une ou plusieurs commandes.

`which`
: Afficher l'emplacement dans le système de fichiers pour une commande.

`history`
: Afficher ou réutiliser les commandes que vous avez invoquées précédemment.

Réponses aux exercices guidés

1. Utilisez le système `man` pour déterminer comment indiquer à `apropos` de fournir une commande brève afin qu'il affiche seulement un bref message d'utilisation et qu'il quitte ensuite.

   Lancez `man apropos` et faites défiler la section "Options" jusqu'au paragraphe `--usage`.

2. Utilisez le système `man` pour déterminer quelle licence est attribuée à la commande `grep`.

   Lancez `man grep` et descendez jusqu'à la section "Copyright" du document. Notez que le programme utilise un copyright de la Free Software Foundation.

Réponses aux exercices d'approfondissement

1. Identifiez l'architecture matérielle et la version du noyau Linux utilisées sur votre ordinateur dans un format d'affichage facile à lire.

   Lancez `man uname`, lisez la section "Description", et identifiez les options de commande qui vous permettront d'obtenir les résultats précis que vous souhaitez. Notez que `-v` vous fournira la version du noyau et `-i` la plateforme matérielle.

   ```
   $ man uname
   $ uname -v
   $ uname -i
   ```

2. Affichez les vingt dernières lignes de la base de données dynamique `history` et du fichier `.bash_history` et comparez les deux résultats.

   ```
   $ history 20
   $ tail -n 20 .bash_history
   ```

3. Utilisez l'outil `apropos` pour identifier la page `man` où vous trouverez la commande dont vous aurez besoin pour afficher la taille d'un périphérique bloc matériel connecté en octets plutôt qu'en mégaoctets ou en gigaoctets.

   Une façon de procéder consiste à lancer `apropos` avec la chaîne `block`, lire le résultat, noter que `lsblk` affiche les périphériques de type bloc (et constitue donc l'outil le plus adapté à nos besoins), lancer `man lsblk`, faire défiler la section "Description" et noter que `-b` affiche la taille du périphérique en octets. Il ne reste qu'à lancer `lsblk -b` pour voir ce qui en ressort.

   ```
   $ apropos block
   $ man lsblk
   $ lsblk -b
   ```

103.1 Leçon 2

Certification :	LPIC-1
Version :	5.0
Thème :	103 Commandes GNU et Unix
Objectif :	103.1 Travailler en ligne de commande
Leçon :	2 sur 2

Introduction

L'environnement d'un système d'exploitation comprend les outils de base - comme les interpréteurs de commandes et éventuellement une interface graphique - dont vous aurez besoin pour travailler. Mais votre environnement sera également doté d'un ensemble de raccourcis et de valeurs prédéfinies. Nous allons apprendre ici à lister, invoquer et gérer ces valeurs.

Trouver vos variables d'environnement

Comment identifier les valeurs courantes de chacune de nos variables d'environnement ? Une façon de le faire est d'utiliser la commande `env` :

```
$ env
DBUS_SESSION_BUS_ADDRESS=unix:path=/run/user/1000/bus
XDG_RUNTIME_DIR=/run/user/1000
XAUTHORITY=/run/user/1000/gdm/Xauthority
XDG_CONFIG_DIRS=/etc/xdg/xdg-ubuntu:/etc/xdg
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/game
s:/snap/bin
```

```
GJS_DEBUG_TOPICS=JS ERROR;JS LOG
[...]
```

Vous obtiendrez beaucoup de résultats, bien plus que ce qui est inclus dans l'extrait ci-dessus. Mais pour l'instant, notez l'entrée `PATH`, qui contient les répertoires où votre shell (et d'autres programmes) vont chercher d'autres programmes sans avoir à spécifier le chemin complet. Avec cette configuration, vous pouvez exécuter un programme binaire qui se trouve, disons, dans `/usr/local/bin` depuis votre répertoire personnel et il fonctionnera comme si le fichier était local.

Changeons de sujet un instant. La commande `echo` affiche à l'écran tout ce que vous lui fournissez en argument. Eh bien, il y aura bien des fois où faire répéter littéralement quelque chose à `echo` nous sera très utile.

```
$ echo "Hi. How are you?"
Hi. How are you?
```

Mais il y a autre chose que vous pouvez faire avec `echo`. Lorsque vous lui fournissez le nom d'une variable d'environnement — et que vous lui indiquez qu'il s'agit d'une variable en préfixant le nom de la variable par un `$` — alors, au lieu de simplement afficher le nom de la variable, l'interpréteur de commandes la développera en vous donnant sa valeur. Vous ne savez pas si votre répertoire préféré est actuellement dans le chemin ? Vous pouvez le vérifier rapidement en passant par `echo` :

```
$ echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/sn
ap/bin
```

Créer de nouvelles variables d'environnement

Vous pouvez ajouter vos propres variables personnalisées à votre environnement. La méthode la plus simple consiste à utiliser le caractère `=`. La chaîne de caractères à gauche sera le nom de votre nouvelle variable et la chaîne de caractères à droite sera sa valeur. Vous pouvez dorénavant passer le nom de la variable à `echo` pour confirmer que ça a marché :

```
$ myvar=hello
$ echo $myvar
hello
```

NOTE Vous remarquerez qu'il n'y a pas d'espace de part et d'autre du signe égal dans l'affectation des variables.

Mais est-ce que ça a vraiment marché ? Tapez `bash` dans le terminal pour lancer un nouveau shell. Ce nouveau shell ressemble exactement à celui que vous venez d'utiliser, mais il est en fait un *enfant* du shell original (que nous appelons le *parent*). Maintenant, à l'intérieur de ce nouveau shell enfant, essayez de faire fonctionner `echo` comme avant. Rien. Que se passe-t-il ?

```
$ bash
$ echo $myvar

$
```

Une variable que vous créez de la façon que nous venons de décrire ne sera disponible que localement, dans la session actuelle du shell. Si vous lancez un nouveau shell — ou si vous fermez la session en utilisant `exit` — la variable ne vous suivra pas. En tapant `exit` ici, vous retrouvez votre shell parent d'origine qui, pour l'instant, est l'endroit où nous voulons être. Vous pouvez relancer `echo $myvar` si vous voulez, juste pour vérifier que la variable est toujours disponible. Maintenant, tapez `export myvar` pour transmettre la variable à tous les shells enfants que vous pourrez ouvrir par la suite. Essayez : tapez `bash` pour lancer un nouveau shell et ensuite `echo` :

```
$ exit
$ export myvar
$ bash
$ echo $myvar
hello
```

Tout cela peut sembler un peu bête lorsque l'on crée des shells sans objectif valable. Mais comprendre comment les variables de l'interpréteur de commandes se propagent dans votre système deviendra très important lorsque vous commencerez à écrire des scripts concrets.

Supprimer des variables d'environnement

Vous voulez savoir comment faire le ménage dans toutes ces variables éphémères que vous avez créées ? Une solution consiste tout simplement à fermer votre shell parent - ou à redémarrer votre machine. Mais il y a plus simple. Comme `unset`, par exemple. Taper `unset` (sans le `$`) va détruire la variable. `echo` nous le prouvera.

```
$ unset myvar
$ echo $myvar
```

```
$
```

S'il y a une commande `unset`, il y a fort à parier qu'il existe une commande `set` qui va avec. Exécuter `set` tout seul affichera beaucoup de choses, et ce ne sera pas très différent de ce que `env` nous a donné. Regardez la première ligne du résultat lorsque vous filtrez avec le terme `PATH` :

```
$ set | grep PATH
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/game
s:/snap/bin
[...]
```

Quelle est la différence entre `set` et `env` ? En ce qui nous concerne, l'essentiel est que `set` affiche toutes les variables et toutes les fonctions. Prenons un exemple. Nous allons créer une nouvelle variable appelée `mynewvar` et confirmer qu'elle est bien là :

```
$ mynewvar=goodbye
$ echo $mynewvar
goodbye
```

Maintenant, lancer `env` en utilisant `grep` pour filtrer la chaîne `mynewvar` n'affichera rien. Mais en exécutant `set` de la même manière, nous verrons notre variable locale.

```
$ env | grep mynewvar

$ set | grep mynewvar
mynewvar=goodbye
```

Guillemets et échappement des caractères spéciaux

Le moment est venu de vous présenter le problème des caractères spéciaux. En temps normal, les caractères alphanumériques (a-z et 0-9) seront lus littéralement par Bash. Si vous essayez de créer un nouveau fichier appelé `myfile`, il vous suffit de taper `touch` suivi de `myfile` et Bash saura quoi en faire. Mais si vous voulez inclure un caractère spécial dans votre nom de fichier, ça se complique un peu.

Pour illustrer cela, nous allons taper `touch` et enchaîner avec le nom : `my big file`. Le problème, c'est qu'il y a deux espaces entre les mots que Bash va interpréter. Même si, techniquement, un espace n'est pas un "caractère", il en est un dans le sens où Bash ne le lira pas

littéralement. Si vous affichez le contenu de votre répertoire actuel, au lieu d'un fichier appelé `my big file`, vous verrez trois fichiers nommés respectivement `my`, `big` et `file`. En effet, Bash considère que vous souhaitez créer une série de fichiers dont vous fournissez les noms dans une liste :

```
$ touch my big file
$ ls
my big file
```

Les espaces seront interprétés de la même manière si vous supprimez (`rm`) les trois fichiers en une seule commande :

```
$ rm my big file
```

Maintenant, essayons de procéder de la bonne manière. Tapez `touch` et les trois parties de votre nom de fichier, mais cette fois-ci, mettez le nom entre guillemets. Cette fois-ci ça a marché. Lorsque vous affichez le contenu du répertoire, vous voyez un seul fichier avec le nom approprié.

```
$ touch "my big file"
$ ls
'my big file'
```

Il existe d'autres façons d'obtenir le même résultat. Les guillemets simples, par exemple, fonctionnent tout aussi bien que les guillemets doubles. (Notez que les guillemets simples préservent la valeur littérale de tous les caractères, tandis que les guillemets doubles préservent tous les caractères *sauf* `$`, `` ` ``, `\` et, dans certains cas, `!`).

```
$ rm 'my big file'
```

En faisant précéder chaque caractère spécial d'une barre oblique inversée, on "échappe" à la spécificité du caractère et Bash le lira littéralement.

```
$ touch my\ big\ file
```

Exercices guidés

1. Utilisez la commande `export` pour ajouter un nouveau répertoire à votre PATH (ceci ne survivra pas à un redémarrage).

2. Utilisez la commande `unset` pour supprimer la variable `PATH`. Essayez de lancer une commande en utilisant `sudo` (comme `sudo cat /etc/shadow`). Que s'est-il passé ? Pourquoi ? (Quittez votre shell pour rétablir le bon fonctionnement des choses.)

Exercices d’approfondissement

1. Faites des recherches sur Internet pour trouver et étudier la liste complète des caractères spéciaux.
2. Essayez d’exécuter des commandes en utilisant des chaînes composées de caractères spéciaux et en utilisant différentes méthodes d’échappement. Est-ce que ces méthodes se comportent différemment ?

Résumé

Dans cette leçon, vous avez appris à :

- Comment identifier les variables d'environnement de votre système.
- Comment créer vos propres variables d'environnement et les exporter vers d'autres shells.
- Comment supprimer des variables d'environnement et comment utiliser les commandes `env` et `set`.
- Comment échapper les caractères spéciaux pour que Bash les lise littéralement.

Les commandes suivantes ont été abordées dans cette leçon :

`echo`
Affiche les chaînes et les variables données.

`env`
Affiche et modifie vos variables d'environnement.

`export`
Passe une variable d'environnement aux shells enfants.

`unset`
Supprime les valeurs et les attributs des variables et des fonctions du shell.

Réponses aux exercices guidés

1. Utilisez la commande `export` pour ajouter un nouveau répertoire à votre PATH (ceci ne survivra pas à un redémarrage).

   Vous pouvez ajouter temporairement un nouveau répertoire (par exemple un répertoire appelé `myfiles` qui se trouve dans votre répertoire personnel) à votre PATH en utilisant `export PATH="/home/yourname/myfiles:$PATH"`. Créez un script simple dans le répertoire `myfiles/`, rendez-le exécutable, et essayez de le lancer depuis un autre répertoire. Ces commandes partent du principe que vous êtes dans votre répertoire personnel qui contient un répertoire appelé `myfiles`.

   ```
   $ touch myfiles/myscript.sh
   $ echo '#!/bin/bash' >> myfiles/myscript.sh
   $ echo 'echo Hello' >> myfiles/myscript.sh
   $ chmod +x myfiles/myscript.sh
   $ myscript.sh
   Hello
   ```

2. Utilisez la commande `unset` pour supprimer la variable `PATH`. Essayez de lancer une commande en utilisant `sudo` (comme `sudo cat /etc/shadow`). Que s'est-il passé ? Pourquoi ? (Quittez votre shell pour rétablir le bon fonctionnement des choses.)

   En tapant `unset PATH`, vous supprimerez les paramètres actuels du PATH. Toute tentative d'invoquer un binaire sans son chemin complet va échouer. Pour cette raison, la tentative d'exécuter une commande en utilisant sudo (qui est lui-même un programme binaire situé dans `/usr/bin/sudo`) va échouer — à moins que vous ne spécifiez l'emplacement absolu, comme dans : `/usr/bin/sudo /bin/cat /etc/shadow`. Vous pouvez réinitialiser votre `PATH` en utilisant `export` ou en quittant simplement le shell.

Réponses aux exercices d'approfondissement

1. Faites des recherches sur Internet pour trouver et étudier la liste complète des caractères spéciaux.

   Voici la liste : `& ; | * ? " ' [ ] ( ) $ < > { } # / \ ! ~`.

2. Essayez d'exécuter des commandes en utilisant des chaînes composées de caractères spéciaux et en utilisant différentes méthodes d'échappement. Est-ce que ces méthodes se comportent différemment ?

   L'échappement à l'aide des caractères `"` préservera les valeurs spéciales du signe dollar, de la quote inversée et de la barre oblique inversée. L'échappement à l'aide d'un caractère `'`, par contre, rendra *tous* les caractères littéralement.

```
$ echo "$mynewvar"
goodbye
$ echo '$mynewvar'
$mynewvar
```

103.2 Traitement de flux de type texte avec des filtres

Référence aux objectifs de LPI

LPIC-1 v5, Exam 101, Objective 103.2

Valeur

2

Domaines de connaissance les plus importants

- Envoi de fichiers textes ou de sorties de commandes à des filtres textuels pour les modifier en utilisant des commandes UNIX appartenant au paquetage GNU textutils.

Liste partielle de termes, fichiers et utilitaires utilisés pour cet objectif

- `bzcat`
- `cat`
- `cut`
- `head`
- `less`
- `md5sum`
- `nl`
- `od`
- `paste`
- `sed`
- `sha256sum`
- `sha512sum`
- `sort`

- `split`
- `tail`
- `tr`
- `uniq`
- `wc`
- `xzcat`
- `zcat`

Linux Professional Institute

103.2 Leçon 1

Certification :	LPIC-1
Version :	5.0
Thème :	103 Commandes GNU et Unix
Objectif :	103.2 Traiter les flux de texte avec des filtres
Leçon :	1 sur 1

Introduction

Gérer du texte constitue une part considérable du travail de tout administrateur système. Doug McIlroy, membre de l'équipe de développement initiale d'Unix, a résumé la philosophie d'Unix en disant (entre autres choses importantes) : "Écrivez des programmes pour gérer les flux de texte, car c'est une interface universelle." Linux est inspiré par le système d'exploitation Unix et en adopte résolument la philosophie. Un administrateur doit donc s'attendre à trouver une multitude d'outils pour manipuler du texte dans une distribution Linux.

Aperçu rapide des redirections et des pipelines

Un autre extrait de la philosophie Unix :

- Écrivez des programmes qui effectuent une seule chose et qui le font bien.
- Écrivez des programmes qui collaborent.

L'un des principaux moyens de faire travailler les programmes ensemble passe par les *pipelines* (tubes de communication) et les *redirections*. A peu près tous vos programmes de manipulation de texte vont récupérer du texte depuis une entrée standard (*stdin*), le sortir vers une sortie standard

(*stdout*) et envoyer les erreurs éventuelles vers une sortie d'erreur standard (*stderr*). Sauf indication contraire, l'entrée standard sera ce que vous tapez sur votre clavier (le programme le lira une fois que vous aurez appuyé sur la touche Entrée). De même, la sortie standard et les erreurs seront affichées sur l'écran de votre terminal. Voyons de plus près comment ça marche.

Dans votre terminal, tapez `cat` et appuyez sur la touche Entrée. Puis tapez du texte au hasard.

```
$ cat
This is a test
This is a test
Hey!
Hey!
It is repeating everything I type!
It is repeating everything I type!
(I will hit ctrl+c so I will stop this nonsense)
(I will hit ctrl+c so I will stop this nonsense)
^C
```

Pour plus de détails sur la commande `cat` (le terme vient de "concaténer"), n'hésitez pas à lire la page de manuel en ligne correspondante.

NOTE

Si vous travaillez sur une installation très minimale d'un serveur Linux, certaines commandes comme `info` et `less` ne seront peut-être pas disponibles. Si c'est le cas, installez ces programmes en suivant la procédure appropriée pour votre système, comme nous l'avons décrit dans les leçons correspondantes.

Comme nous venons de le voir, si vous ne spécifiez pas où `cat` doit lire, il lira l'entrée standard (ce que vous tapez) et enverra ce qu'il lit vers votre fenêtre de terminal (sa sortie standard).

Maintenant essayez ce qui suit :

```
$ cat > mytextfile
This is a test
I hope cat is storing this to mytextfile as I redirected the output
I will hit ctrl+c now and check this
^C

$ cat mytextfile
This is a test
I hope cat is storing this to mytextfile as I redirected the output
I will hit ctrl+c now and check this
```

Le `>` (plus grand que) indique à `cat` de rediriger sa sortie vers le fichier `mytextfile` au lieu de la sortie standard. Maintenant, essayez ceci :

```
$ cat mytextfile > mynewtextfile
$ cat mynewtextfile
This is a test
I hope cat is storing this to mytextfile as I redirected the output
I will hit ctrl+c now and check this
```

Cela a pour effet de copier `mytextfile` vers `mynewtextfile`. Vous pouvez vérifier que ces deux fichiers ont le même contenu en effectuant un `diff` :

```
$ diff mynewtextfile mytextfile
```

Comme il n'y a pas de résultat, les fichiers sont identiques. Essayez maintenant l'opérateur de redirection pour ajouter des données (`>>`) :

```
$ echo 'This is my new line' >> mynewtextfile
$ diff mynewtextfile mytextfile
4d3
< This is my new line
```

Pour l'instant, nous avons utilisé les redirections pour créer et manipuler des fichiers. Nous pouvons également utiliser les pipelines (représentés par le symbole `|`) pour rediriger la sortie d'un programme vers un autre programme. Recherchons toutes les lignes où figure le mot "this" :

```
$ cat mytextfile | grep this
I hope cat is storing this to mytextfile as I redirected the output
I will hit ctrl+c now and check this

$ cat mytextfile | grep -i this
This is a test
I hope cat is storing this to mytextfile as I redirected the output
I will hit ctrl+c now and check this
```

Ici, nous avons envoyé la sortie de `cat` vers une autre commande : `grep`. Notez que lorsque nous ignorons la casse (en utilisant l'option `-i`), nous obtenons une ligne supplémentaire comme résultat.

Traiter les flux de texte

Lire un fichier compressé

Nous allons créer un fichier nommé `ftu.txt` qui contient la liste des commandes suivantes :

```
bzcat
cat
cut
head
less
md5sum
nl
od
paste
sed
sha256sum
sha512sum
sort
split
tail
tr
uniq
wc
xzcat
zcat
```

Maintenant, nous allons utiliser la commande `grep` pour afficher toutes les lignes qui contiennent la chaine de caractères `cat` :

```
$ cat ftu.txt | grep cat
bzcat
cat
xzcat
zcat
```

Une autre façon d'obtenir cette information consiste à utiliser la commande `grep` pour filtrer le texte directement, sans passer par une autre application pour envoyer le flux de texte vers la sortie standard.

```
$ grep cat ftu.txt
```

```
bzcat
cat
xzcat
zcat
```

NOTE N'oubliez pas qu'il y a plusieurs façons d'effectuer la même tâche sous Linux.

Il existe d'autres commandes qui gèrent les fichiers compressés (`bzcat` pour les fichiers compressés avec `bzip`, `xzcat` pour les fichiers compressés avec `xz` et `zcat` pour les fichiers compressés avec `gzip`) et chacune est utilisée pour visualiser le contenu d'un fichier compressé en fonction de l'algorithme de compression utilisé.

Vérifiez que le fichier `ftu.txt` nouvellement créé est bien le seul dans le répertoire, puis créez une version compressée du fichier avec `gzip` :

```
$ ls ftu*
ftu.txt

$ gzip ftu.txt
$ ls ftu*
ftu.txt.gz
```

Ensuite, utilisez la commande `zcat` pour visualiser le contenu du fichier compressé avec `gzip` :

```
$ zcat ftu.txt.gz
bzcat
cat
cut
head
less
md5sum
nl
od
paste
sed
sha256sum
sha512sum
sort
split
tail
tr
uniq
```

```
wc
xzcat
zcat
```

Notez que `gzip` va compresser `ftu.txt` en `ftu.txt.gz` en supprimant le fichier de départ. Par défaut, la commande `gzip` ne vous retournera rien. Cependant, si vous voulez que `gzip` affiche ce qu'il fait, utilisez l'option `-v` pour la sortie "verbeuse".

Afficher un fichier dans un programme de pagination

Vous savez que `cat` concatène un fichier vers la sortie standard (une fois qu'un fichier est fourni après la commande). Le fichier `/var/log/syslog` est l'endroit où votre système Linux stocke tout ce qui se passe d'important dans votre système. Utilisez la commande `sudo` pour acquérir les privilèges nécessaires pour lire le fichier `/var/log/syslog` :

```
$ sudo cat /var/log/syslog
```

Vous allez voir des messages qui défilent très rapidement dans votre fenêtre de terminal. Vous pouvez envoyer la sortie vers le programme `less` pour afficher les résultats page par page. Avec `less`, vous pouvez utiliser les touches fléchées pour naviguer dans le résultat ainsi que les raccourcis de type `vi` pour vous déplacer dans le texte et effectuer des recherches.

Mais plutôt que de passer la commande `cat` à un programme de pagination, il est plus commode d'utiliser directement le pageur :

```
$ sudo less /var/log/syslog
... (résultat occulté pour plus de lisibilité)
```

Récupérer une partie d'un fichier texte

Si vous ne souhaitez examiner que le début ou la fin d'un fichier, il existe d'autres approches. La commande `head` est utilisée pour lire les dix premières lignes d'un fichier par défaut, et la commande `tail` est utilisée pour lire les dix dernières lignes d'un fichier par défaut. Maintenant essayez :

```
$ sudo head /var/log/syslog
Nov 12 08:04:30 hypatia rsyslogd: [origin software="rsyslogd" swVersion="8.1910.0" x-
pid="811" x-info="https://www.rsyslog.com"] rsyslogd was HUPed
Nov 12 08:04:30 hypatia systemd[1]: logrotate.service: Succeeded.
```

```
Nov 12 08:04:30 hypatia systemd[1]: Started Rotate log files.
Nov 12 08:04:30 hypatia vdr: [928] video directory scanner thread started (pid=882, tid=928,
prio=low)
Nov 12 08:04:30 hypatia vdr: [882] registered source parameters for 'A - ATSC'
Nov 12 08:04:30 hypatia vdr: [882] registered source parameters for 'C - DVB-C'
Nov 12 08:04:30 hypatia vdr: [882] registered source parameters for 'S - DVB-S'
Nov 12 08:04:30 hypatia vdr: [882] registered source parameters for 'T - DVB-T'
Nov 12 08:04:30 hypatia vdr[882]: vdr: no primary device found - using first device!
Nov 12 08:04:30 hypatia vdr: [929] epg data reader thread started (pid=882, tid=929,
prio=high)
$ sudo tail /var/log/syslog
Nov 13 10:24:45 hypatia kernel: [ 8001.679238] mce: CPU7: Core temperature/speed normal
Nov 13 10:24:46 hypatia dbus-daemon[2023]: [session uid=1000 pid=2023] Activating via
systemd: service name='org.freedesktop.Tracker1.Miner.Extract' unit='tracker-
extract.service' requested by ':1.73' (uid=1000 pid=2425 comm="/usr/lib/tracker/tracker-
miner-fs ")
Nov 13 10:24:46 hypatia systemd[2004]: Starting Tracker metadata extractor...
Nov 13 10:24:47 hypatia dbus-daemon[2023]: [session uid=1000 pid=2023] Successfully
activated service 'org.freedesktop.Tracker1.Miner.Extract'
Nov 13 10:24:47 hypatia systemd[2004]: Started Tracker metadata extractor.
Nov 13 10:24:54 hypatia kernel: [ 8010.462227] mce: CPU0: Core temperature above threshold,
cpu clock throttled (total events = 502907)
Nov 13 10:24:54 hypatia kernel: [ 8010.462228] mce: CPU4: Core temperature above threshold,
cpu clock throttled (total events = 502911)
Nov 13 10:24:54 hypatia kernel: [ 8010.469221] mce: CPU0: Core temperature/speed normal
Nov 13 10:24:54 hypatia kernel: [ 8010.469222] mce: CPU4: Core temperature/speed normal
Nov 13 10:25:03 hypatia systemd[2004]: tracker-extract.service: Succeeded.
```

Afin d'illustrer le nombre de lignes affichées, nous pouvons rediriger la sortie de la commande `head` vers la commande `nl`, qui affichera le nombre de lignes de texte transmises à la commande :

```
$ sudo head /var/log/syslog | nl
1   Nov 12 08:04:30 hypatia rsyslogd: [origin software="rsyslogd" swVersion="8.1910.0" x-
pid="811" x-info="https://www.rsyslog.com"] rsyslogd was HUPed
2   Nov 12 08:04:30 hypatia systemd[1]: logrotate.service: Succeeded.
3   Nov 12 08:04:30 hypatia systemd[1]: Started Rotate log files.
4   Nov 12 08:04:30 hypatia vdr: [928] video directory scanner thread started (pid=882,
tid=928, prio=low)
5   Nov 12 08:04:30 hypatia vdr: [882] registered source parameters for 'A - ATSC'
6   Nov 12 08:04:30 hypatia vdr: [882] registered source parameters for 'C - DVB-C'
7   Nov 12 08:04:30 hypatia vdr: [882] registered source parameters for 'S - DVB-S'
8   Nov 12 08:04:30 hypatia vdr: [882] registered source parameters for 'T - DVB-T'
9   Nov 12 08:04:30 hypatia vdr[882]: vdr: no primary device found - using first device!
```

```
10  Nov 12 08:04:30 hypatia vdr: [929] epg data reader thread started (pid=882, tid=929,
prio=high)
```

Et nous pouvons faire de même en redirigeant la sortie de la commande `tail` vers la commande `wc`, qui par défaut va compter le nombre de mots dans un document, et en utilisant l'option `-l` pour afficher le nombre de lignes de texte que la commande a lues :

```
$ sudo tail /var/log/syslog | wc -l
10
```

Au cas où un administrateur aurait besoin de voir plus (ou moins) du début ou de la fin d'un fichier, l'option `-n` pourra être utilisée pour limiter la sortie des commandes :

```
$ sudo tail -n 5 /var/log/syslog
Nov 13 10:37:24 hypatia systemd[2004]: tracker-extract.service: Succeeded.
Nov 13 10:37:42 hypatia dbus-daemon[2023]: [session uid=1000 pid=2023] Activating via
systemd: service name='org.freedesktop.Tracker1.Miner.Extract' unit='tracker-
extract.service' requested by ':1.73' (uid=1000 pid=2425 comm="/usr/lib/tracker/tracker-
miner-fs ")
Nov 13 10:37:42 hypatia systemd[2004]: Starting Tracker metadata extractor...
Nov 13 10:37:43 hypatia dbus-daemon[2023]: [session uid=1000 pid=2023] Successfully
activated service 'org.freedesktop.Tracker1.Miner.Extract'
Nov 13 10:37:43 hypatia systemd[2004]: Started Tracker metadata extractor.
$ sudo head -n 12 /var/log/syslog
Nov 12 08:04:30 hypatia rsyslogd: [origin software="rsyslogd" swVersion="8.1910.0" x-
pid="811" x-info="https://www.rsyslog.com"] rsyslogd was HUPed
Nov 12 08:04:30 hypatia systemd[1]: logrotate.service: Succeeded.
Nov 12 08:04:30 hypatia systemd[1]: Started Rotate log files.
Nov 12 08:04:30 hypatia vdr: [928] video directory scanner thread started (pid=882, tid=928,
prio=low)
Nov 12 08:04:30 hypatia vdr: [882] registered source parameters for 'A - ATSC'
Nov 12 08:04:30 hypatia vdr: [882] registered source parameters for 'C - DVB-C'
Nov 12 08:04:30 hypatia vdr: [882] registered source parameters for 'S - DVB-S'
Nov 12 08:04:30 hypatia vdr: [882] registered source parameters for 'T - DVB-T'
Nov 12 08:04:30 hypatia vdr[882]: vdr: no primary device found - using first device!
Nov 12 08:04:30 hypatia vdr: [929] epg data reader thread started (pid=882, tid=929,
prio=high)
Nov 12 08:04:30 hypatia vdr: [882] no DVB device found
Nov 12 08:04:30 hypatia vdr: [882] initializing plugin: vnsiserver (1.8.0): VDR-Network-
Streaming-Interface (VNSI) Server
```

Prise en main de l'éditeur de flux sed

Jetons un œil aux autres fichiers, termes et outils dont le nom ne contient pas `cat`. Nous pouvons faire cela en passant l'option `-v` à `grep`, qui demande à la commande de ne sortir que les lignes qui ne contiennent pas `cat` :

```
$ zcat ftu.txt.gz | grep -v cat
cut
head
less
md5sum
nl
od
paste
sed
sha256sum
sha512sum
sort
split
tail
tr
uniq
wc
```

La plupart des choses que l'on peut faire avec `grep` peuvent également être effectuées avec `sed` — l'éditeur de flux pour filtrer et transformer du texte (comme indiqué dans la page de manuel de `sed`). Pour commencer, nous allons récupérer notre fichier `ftu.txt` en décompressant notre archive `gzip` du fichier :

```
$ gunzip ftu.txt.gz
$ ls ftu*
ftu.txt
```

Maintenant, nous pouvons utiliser `sed` pour afficher les seules lignes qui contiennent la chaîne de caractères `cat` :

```
$ sed -n /cat/p < ftu.txt
bzcat
cat
xzcat
```

```
zcat
```

Nous avons utilisé le signe moins que `<` pour envoyer le contenu du fichier `ftu.txt` vers notre commande `sed`. Le mot placé entre les barres obliques (en l'occurrence `/cat/`) est le terme recherché. L'option `-n` indique à `sed` de ne pas générer de résultat (sauf ceux requis ultérieurement par la commande `p`). Essayez d'exécuter cette même commande sans l'option `-n` pour voir ce qui se passe. Puis essayez ceci :

```
$ sed /cat/d < ftu.txt
cut
head
less
md5sum
nl
od
paste
sed
sha256sum
sha512sum
sort
split
tail
tr
uniq
wc
```

Si nous n'utilisons pas l'option `-n`, `sed` va afficher tout ce qui se trouve dans le fichier à l'exception de ce que le `d` lui demande de supprimer de sa sortie.

Une utilisation courante de `sed` consiste à rechercher et remplacer du texte dans un fichier. Admettons que vous vouliez changer chaque occurrence de `cat` en `dog`. Vous pouvez utiliser `sed` pour ce faire en utilisant l'option `s` qui remplace chaque instance du premier terme, `cat`, par le second terme, `dog` :

```
$ sed s/cat/dog/ < ftu.txt
bzdog
dog
cut
head
less
md5sum
```

```
nl
od
paste
sed
sha256sum
sha512sum
sort
split
tail
tr
uniq
wc
xzdog
zdog
```

Plutôt que d'utiliser un opérateur de redirection (`<`) pour passer le fichier `ftu.txt` à notre commande `sed`, nous pouvons simplement demander à la commande `sed` d'agir directement sur le fichier. Nous allons essayer cela en créant à la volée une sauvegarde du fichier original :

```
$ sed -i.backup s/cat/dog/ ftu.txt
$ ls ftu*
ftu.txt  ftu.txt.backup
```

L'option `-i` va effectuer une opération `sed` *directement* sur le fichier d'origine. Si vous n'utilisez pas `.backup` après l'option `-i`, vous allez simplement modifier votre fichier d'origine. Le texte invoqué après l'option `-i` va constituer l'extension utilisée pour sauvegarder le fichier original avant les modifications effectuées par `sed`.

Garantir l'intégrité des données

Nous avons montré à quel point il est facile de manipuler des fichiers sous Linux. Il peut arriver que vous souhaitiez transmettre un fichier à quelqu'un d'autre, et vous voulez être sûr que le destinataire se retrouve avec une copie conforme du fichier original. Une utilisation très courante de cette technique est mise en œuvre lorsque des serveurs de distributions Linux hébergent des images CD ou DVD téléchargeables de leurs logiciels ainsi que des fichiers contenant les sommes de contrôle de ces images disques. Voici un exemple de listing provenant d'un miroir de téléchargement Debian :

```
                                                                    
[PARENTDIR] Parent Directory                                    -
[SUM]       MD5SUMS                                 2019-09-08 17:46 274
```

```
[CRT]       MD5SUMS.sign                          2019-09-08 17:52 833
[SUM]       SHA1SUMS                              2019-09-08 17:46 306
[CRT]       SHA1SUMS.sign                         2019-09-08 17:52 833
[SUM]       SHA256SUMS                            2019-09-08 17:46 402
[CRT]       SHA256SUMS.sign                       2019-09-08 17:52 833
[SUM]       SHA512SUMS                            2019-09-08 17:46 658
[CRT]       SHA512SUMS.sign                       2019-09-08 17:52 833
[ISO]       debian-10.1.0-amd64-netinst.iso       2019-09-08 04:37 335M
[ISO]       debian-10.1.0-amd64-xfce-CD-1.iso     2019-09-08 04:38 641M
[ISO]       debian-edu-10.1.0-amd64-netinst.iso   2019-09-08 04:38 405M
[ISO]       debian-mac-10.1.0-amd64-netinst.iso   2019-09-08 04:38 334M
```

Dans le listing ci-dessus, les fichiers ISO de l'installateur Debian sont accompagnés de fichiers texte qui contiennent les sommes de contrôle des fichiers à partir de plusieurs algorithmes (MD5, SHA1, SHA256 et SHA512).

NOTE

Une somme de contrôle est une valeur dérivée d'un calcul mathématique basé sur une fonction de hachage cryptographique et appliqué à un fichier. Il existe différents types de fonctions de hachage cryptographiques qui varient en robustesse. L'examen attendra de vous que vous soyez familiarisé avec l'utilisation de `md5sum`, `sha256sum` et `sha512sum`.

Une fois que vous avez téléchargé un fichier (par exemple, l'image `debian-10.1.0-amd64-netinst.iso`), vous devez comparer la somme de contrôle du fichier téléchargé avec une valeur de somme de contrôle donnée.

Voici un exemple pour illustrer notre propos. Nous allons calculer la valeur SHA256 du fichier `ftu.txt` en utilisant la commande `sha256sum` :

```
$ sha256sum ftu.txt
345452304fc26999a715652543c352e5fc7ee0c1b9deac6f57542ec91daf261c  ftu.txt
```

La longue chaîne de caractères qui précède le nom du fichier est la somme de contrôle SHA256 de ce fichier texte. Nous allons créer un fichier qui contient cette valeur de manière à pouvoir l'utiliser pour vérifier l'intégrité de notre fichier texte original. Nous pouvons effectuer cette opération avec la même commande `sha256sum` en redirigeant la sortie vers un fichier :

```
$ sha256sum ftu.txt > sha256.txt
```

Maintenant, pour vérifier le fichier `ftu.txt`, nous allons utiliser la même commande en

fournissant le nom du fichier qui contient notre somme de contrôle avec l'option `-c` :

```
$ sha256sum -c sha256.txt
ftu.txt: OK
```

La valeur contenue dans le fichier correspond à la somme de contrôle SHA256 calculée pour notre fichier `ftu.txt`, comme il fallait s'y attendre. En revanche, si le fichier d'origine était modifié (par exemple, si quelques octets avaient été perdus lors du téléchargement du fichier ou si quelqu'un l'avait délibérément corrompu), la vérification de la somme de contrôle échouerait. Dans ce cas, nous savons que notre fichier est mauvais ou corrompu, et nous ne pouvons pas faire confiance à l'intégrité de son contenu. Pour illustrer notre propos, nous allons ajouter du texte à la fin du fichier :

```
$ echo "new entry" >> ftu.txt
```

Nous allons donc tenter de vérifier l'intégrité du fichier :

```
$ sha256sum -c sha256.txt
ftu.txt: FAILED
sha256sum: WARNING: 1 computed checksum did NOT match
```

Et nous voyons que la somme de contrôle ne correspond pas à ce qui était attendu pour le fichier. Par conséquent, nous ne pouvons pas faire confiance à l'intégrité de ce fichier. Nous pouvons tenter de télécharger une nouvelle copie du fichier, signaler l'échec de la somme de contrôle à l'expéditeur du fichier ou à l'équipe de sécurité du datacenter, en fonction de l'importance du fichier.

Examiner les fichiers de plus près

La commande de dump octal (od) est souvent utilisée pour déboguer des applications et toutes sortes de fichiers. En soi, la commande `od` va juste afficher le contenu d'un fichier au format octal. Nous pouvons utiliser notre fichier `ftu.txt` de tout à l'heure pour un peu de pratique avec cette commande :

```
$ od ftu.txt
0000000 075142 060543 005164 060543 005164 072543 005164 062550
0000020 062141 066012 071545 005163 062155 071465 066565 067012
0000040 005154 062157 070012 071541 062564 071412 062145 071412
0000060 060550 032462 071466 066565 071412 060550 030465 071462
```

```
0000100 066565 071412 071157 005164 070163 064554 005164 060564
0000120 066151 072012 005162 067165 070551 073412 005143 075170
0000140 060543 005164 061572 072141 000012
0000151
```

La première colonne du résultat correspond au *décalage des octets* dans chaque ligne. Étant donné que `od` affiche les informations au format octal par défaut, chaque ligne commence par un décalage d'octet de huit bits, suivi de huit colonnes, chacune contenant la valeur octale des données dans cette colonne.

TIP Rappelez-vous qu'un *octet* a une longueur de 8 bits.

Pour afficher le contenu d'un fichier au format hexadécimal, utilisez l'option `-x` :

```
$ od -x ftu.txt
0000000 7a62 6163 0a74 6163 0a74 7563 0a74 6568
0000020 6461 6c0a 7365 0a73 646d 7335 6d75 6e0a
0000040 0a6c 646f 700a 7361 6574 730a 6465 730a
0000060 6168 3532 7336 6d75 730a 6168 3135 7332
0000100 6d75 730a 726f 0a74 7073 696c 0a74 6174
0000120 6c69 740a 0a72 6e75 7169 770a 0a63 7a78
0000140 6163 0a74 637a 7461 000a
0000151
```

À présent, chacune des huit colonnes après le décalage des octets est représentée par son équivalent hexadécimal.

Une application pratique de la commande `od` permet de déboguer les scripts. Par exemple, la commande `od` peut nous afficher des caractères que l'on ne voit pas normalement et qui existent dans un fichier, comme les *sauts de ligne*. Nous pouvons utiliser l'option `-c`, de sorte qu'au lieu d'afficher la notation numérique pour chaque octet, les entrées des colonnes seront affichées avec leurs équivalents en caractères :

```
$ od -c ftu.txt
0000000   b   z   c   a   t  \n   c   a   t  \n   c   u   t  \n   h   e
0000020   a   d  \n   l   e   s   s  \n   m   d   5   s   u   m  \n   n
0000040   l  \n   o   d  \n   p   a   s   t   e  \n   s   e   d  \n   s
0000060   h   a   2   5   6   s   u   m  \n   s   h   a   5   1   2   s
0000100   u   m  \n   s   o   r   t  \n   s   p   l   i   t  \n   t   a
0000120   i   l  \n   t   r  \n   u   n   i   q  \n   w   c  \n   x   z
0000140   c   a   t  \n   z   c   a   t  \n
```

```
0000151
```

Tous les sauts de ligne dans le fichier sont représentés par le caractère échappé `\n`. Si vous souhaitez simplement afficher tous les caractères dans un fichier et que vous n'avez pas besoin de voir les informations relatives au décalage des octets, cette colonne peut être supprimée du résultat comme suit :

```
$ od -An -c ftu.txt
   b   z   c   a   t  \n   c   a   t  \n   c   u   t  \n   h   e
   a   d  \n   l   e   s   s  \n   m   d   5   s   u   m  \n   n
   l  \n   o   d  \n   p   a   s   t   e  \n   s   e   d  \n   s
   h   a   2   5   6   s   u   m  \n   s   h   a   5   1   2   s
   u   m  \n   s   o   r   t  \n   s   p   l   i   t  \n   t   a
   i   l  \n   t   r  \n   u   n   i   q  \n   w   c  \n   x   z
   c   a   t  \n   z   c   a   t  \n
```

Exercices guidés

1. Quelqu'un vient de faire don d'un ordinateur portable à votre école et vous souhaitez maintenant installer Linux dessus. Il n'y a pas de manuel et vous avez été contraint de démarrer sur une clé USB sans environnement graphique. Vous disposez d'un shell et vous savez que pour chaque processeur que vous avez, il y aura une ligne correspondante dans le fichier `/proc/cpuinfo` :

```
processor : 0
vendor_id : GenuineIntel
cpu family : 6
model    : 158

(lignes ignorées)

processor : 1
vendor_id : GenuineIntel
cpu family : 6
model    : 158

(plus de lignes ignorées)
```

   - En utilisant les commandes `grep` et `wc`, affichez le nombre de processeurs dont vous disposez.

   - Faites la même chose avec `sed` au lieu de `grep`.

2. Explorez votre fichier local `/etc/passwd` avec les commandes `grep`, `sed`, `head` et `tail` en fonction des tâches ci-dessous :

   - Quels utilisateurs ont accès à un shell Bash ?

   - Votre système comporte un certain nombre d'utilisateurs destinés à gérer des programmes spécifiques ou à des fins administratives. Ils n'ont pas accès à un shell. Combien d'entre eux existent sur votre système ?

   - Combien d'utilisateurs et de groupes existent sur votre système (rappelez-vous : utilisez

uniquement le fichier `/etc/passwd`) ?

- Affichez uniquement la première ligne, la dernière ligne et la dixième ligne de votre fichier `/etc/passwd`.

3. Prenons cet exemple de fichier `/etc/passwd`. Recopiez les lignes ci-dessous dans un fichier local nommé `mypasswd` pour cet exercice.

```
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
nvidia-persistenced:x:121:128:NVIDIA Persistence Daemon,,,:/nonexistent:/sbin/nologin
libvirt-qemu:x:64055:130:Libvirt Qemu,,,:/var/lib/libvirt:/usr/sbin/nologin
libvirt-dnsmasq:x:122:133:Libvirt Dnsmasq,,,:/var/lib/libvirt/dnsmasq:/usr/sbin/nologin
carol:x:1000:2000:Carol Smith,Finance,,,Main Office:/home/carol:/bin/bash
dave:x:1001:1000:Dave Edwards,Finance,,,Main Office:/home/dave:/bin/ksh
emma:x:1002:1000:Emma Jones,Finance,,,Main Office:/home/emma:/bin/bash
frank:x:1003:1000:Frank Cassidy,Finance,,,Main Office:/home/frank:/bin/bash
grace:x:1004:1000:Grace Kearns,Engineering,,,Main Office:/home/grace:/bin/ksh
henry:x:1005:1000:Henry Adams,Sales,,,Main Office:/home/henry:/bin/bash
john:x:1006:1000:John Chapel,Sales,,,Main Office:/home/john:/bin/bash
```

- Affichez tous les utilisateurs du groupe `1000` (utilisez `sed` pour sélectionner le champ approprié) depuis votre fichier `mypasswd`.

- Affichez uniquement les noms complets de tous les utilisateurs de ce groupe (utilisez `sed` et `cut`).

Exercices d'approfondissement

1. En utilisant à nouveau le fichier `mypasswd` des exercices ci-dessus, trouvez une commande Bash qui sélectionnera au hasard une personne du bureau principal (`Main Office`) pour gagner une tombola. Utilisez la commande `sed` pour afficher uniquement les lignes du bureau principal, puis une séquence de commandes `cut` pour extraire le prénom de chaque utilisateur de ces lignes. Enfin, vous allez trier ces noms au hasard et afficher uniquement le premier nom de la liste.

2. Combien de personnes travaillent dans la finance (`Finance`), l'ingénierie (`Engineering`) et les ventes (`Sales`) ? (Pensez à utiliser la commande `uniq`).

3. Maintenant vous souhaitez préparer un fichier CSV (*Comma Separated Values*) afin de pouvoir facilement importer, depuis le fichier `mypasswd` de l'exemple précédent, le fichier `names.csv` dans LibreOffice. Le contenu du fichier aura le format suivant :

   ```
   First Name,Last Name,Position
   Carol,Smith,Finance
   ...
   John,Chapel,Sales
   ```

   Astuce : Utilisez les commandes `sed`, `cut`, et `paste` pour obtenir le résultat souhaité. Notez que la virgule (`,`) servira de délimiteur pour ce fichier.

4. Admettons que la feuille de calcul `names.csv` créée dans l'exercice précédent soit un fichier important et que nous voulions nous assurer que personne ne puisse l'altérer entre le moment où nous l'envoyons à quelqu'un et le moment où notre destinataire le reçoit. Comment assurer l'intégrité de ce fichier en utilisant `md5sum` ?

5. Vous vous êtes promis de lire un livre classique à raison de 100 lignes par jour et vous avez décidé de commencer par *Mariner and Mystic* de Herman Melville. Imaginez une commande en utilisant `split` pour séparer ce livre en segments de 100 lignes chacun. Pour obtenir le livre en format texte simple, recherchez-le sur https://www.gutenberg.org.

6. En utilisant `ls -l` sur le répertoire `/etc`, quel genre d'affichage obtenez-vous ? En utilisant la commande `cut` sur le résultat de la commande `ls` donnée, comment afficheriez-vous

uniquement les noms des fichiers ? Qu'en est-il du nom de fichier et du propriétaire du fichier ? En plus des commandes `ls -l` et `cut`, utilisez la commande `tr` pour compacter plusieurs occurrences d'un espace en un seul espace afin de faciliter le formatage de la sortie avec une commande `cut`.

7. Cet exercice suppose que vous travaillez sur une machine réelle (pas une machine virtuelle). Vous devez également vous munir d'une clé USB. Consultez les pages de manuel de la commande `tail` et voyez comment faire pour lire un fichier à la volée au fur et à mesure que du texte y est ajouté. Tout en surveillant la sortie de la commande `tail` sur le fichier `/var/log/syslog`, insérez une clé USB. Saisissez la commande complète que vous utiliseriez pour obtenir le produit (`Product`), le fabricant (`Manufacturer`) et la quantité totale de mémoire de votre clé USB.

Résumé

La gestion des flux de texte est d'une importance capitale pour l'administration de n'importe quel système Linux. Les flux de texte peuvent être traités à l'aide de scripts pour automatiser les tâches quotidiennes ou pour trouver des informations de débogage pertinentes dans les fichiers journaux. Voici un aperçu des commandes abordées dans cette leçon :

`cat`

Utilisé pour combiner ou pour lire des fichiers texte simples.

`bzcat`

Permet le traitement ou la lecture de fichiers compressés avec `bzip2`.

`xzcat`

Permet le traitement ou la lecture de fichiers compressés avec `xz`.

`zcat`

Permet le traitement ou la lecture de fichiers compressés avec `gzip`.

`less`

Cette commande affiche le contenu d'un fichier page par page et permet également la navigation et la recherche.

`head`

Cette commande affiche les 10 premières lignes d'un fichier par défaut. L'option `-n` permet d'afficher moins ou plus de lignes.

`tail`

Cette commande affiche les 10 dernières lignes d'un fichier par défaut. L'option `-n` permet d'afficher moins ou plus de lignes. L'option `-f` est utilisée pour afficher à la volée la sortie d'un fichier texte au fur et à mesure que de nouvelles données y sont inscrites.

`wc`

Abréviation de "word count" mais selon les options que vous utilisez, il pourra compter les caractères, les mots et les lignes.

`sort`

Utilisé pour classer le contenu d'un listing par ordre alphabétique, inverse ou aléatoire.

`uniq`

Utilisé pour recenser (et compter) les chaînes de caractères correspondantes.

`od`

La commande “octal dump” est utilisée pour afficher un fichier binaire en notation octale, décimale ou hexadécimale.

`nl`

La commande “number line” va afficher le nombre de lignes dans un fichier et recréer un fichier avec chaque ligne précédée de son numéro de ligne.

`sed`

L’éditeur de flux peut être utilisé pour trouver les occurrences correspondantes de chaînes de caractères à l’aide d’expressions régulières, ainsi que pour éditer des fichiers à l’aide de motifs prédéfinis.

`tr`

Cette commande permet de remplacer des caractères et de supprimer ou compresser des caractères récurrents.

`cut`

Cette commande permet de générer des colonnes de texte sous forme de champs basés sur le délimiteur de caractères du fichier.

`paste`

Joindre les fichiers en colonnes en fonction de l’utilisation des séparateurs de champs.

`split`

Cette commande permet de scinder des fichiers volumineux en plusieurs petits fichiers en fonction des critères définis par les options de la commande.

`md5sum`

Utilisé pour calculer la valeur de hachage MD5 d’un fichier. Également utilisé pour vérifier un fichier par rapport à une valeur de hachage existante afin de garantir son intégrité.

`sha256sum`

Utilisé pour calculer la valeur de hachage SHA256 d’un fichier. Également utilisé pour vérifier un fichier par rapport à une valeur de hachage existante afin de garantir son intégrité.

`sha512sum`

Utilisé pour calculer la valeur de hachage SHA512 d'un fichier. Également utilisé pour vérifier un fichier par rapport à une valeur de hachage existante afin de garantir son intégrité.

Réponses aux exercices guidés

1. Quelqu'un vient de faire don d'un ordinateur portable à votre école et vous souhaitez maintenant installer Linux dessus. Il n'y a pas de manuel et vous avez été contraint de démarrer sur une clé USB sans environnement graphique. Vous disposez d'un shell et vous savez que pour chaque processeur que vous avez, il y aura une ligne correspondante dans le fichier `/proc/cpuinfo` :

```
processor : 0
vendor_id : GenuineIntel
cpu family : 6
model    : 158

(lignes ignorées)

processor : 1
vendor_id : GenuineIntel
cpu family : 6
model    : 158

(plus de lignes ignorées)
```

   - En utilisant les commandes `grep` et `wc`, affichez le nombre de processeurs dont vous disposez.

     Voici deux possibilités :

```
$ cat /proc/cpuinfo | grep processor | wc -l
$ grep processor /proc/cpuinfo | wc -l
```

     Maintenant que vous savez qu'il existe plusieurs façons de faire la même chose, à quel moment devez-vous utiliser l'une ou l'autre ? Cela dépend en fait de plusieurs facteurs, les deux plus importants étant la performance et la lisibilité. La plupart du temps, vous utiliserez des commandes shell à l'intérieur de scripts shell pour automatiser vos tâches. Plus vos scripts sont volumineux et complexes, plus vous devez vous soucier de leur rapidité.

   - Faites la même chose avec `sed` au lieu de `grep`.

     Maintenant, au lieu de `grep`, nous allons essayer avec `sed` :

```
$ sed -n /processor/p /proc/cpuinfo | wc -l
```

Ici, nous avons utilisé `sed` avec l'option `-n` pour que `sed` n'affiche rien sauf ce qui correspond à l'expression `processor`, comme indiqué par la commande `p`. Comme nous l'avons fait dans les solutions basées sur `grep`, `wc -l` va compter le nombre de lignes et donc le nombre de processeurs que nous avons.

Regardez bien le prochain exemple :

```
$ sed -n /processor/p /proc/cpuinfo | sed -n '$='
```

Cette séquence de commandes fournit le même résultat que dans l'exemple précédent où la sortie de `sed` était envoyée vers la commande `wc`. La différence ici est qu'au lieu d'utiliser `wc -l` pour compter le nombre de lignes, `sed` est invoqué à nouveau pour fournir une fonctionnalité équivalente. Encore une fois, nous supprimons les résultats de `sed` avec l'option `-n` à l'exception de l'expression que nous appelons explicitement, en l'occurrence `'$='`. Cette expression demande à `sed` de rechercher la dernière ligne (`$`) et affiche le numéro de cette ligne (`=`).

2. Explorez votre fichier local `/etc/passwd` avec les commandes `grep`, `sed`, `head` et `tail` en fonction des tâches ci-dessous :
   - Quels utilisateurs ont accès à un shell Bash ?

     ```
     $ grep ":/bin/bash$" /etc/passwd
     ```

     Nous allons améliorer cette réponse en affichant uniquement le nom de l'utilisateur qui utilise le shell Bash.

     ```
     $ grep ":/bin/bash$" /etc/passwd | cut -d: -f1
     ```

     Le nom d'utilisateur est le premier champ (paramètre `-f1` de la commande `cut`) et le fichier `/etc/passwd` utilise des `:` comme séparateurs (paramètre `-d:` de la commande `cut`). Il suffit d'envoyer la sortie de la commande `grep` vers la commande `cut` appropriée.

   - Votre système comporte un certain nombre d'utilisateurs destinés à gérer des programmes spécifiques ou à des fins administratives. Ils n'ont pas accès à un shell. Combien d'entre eux existent sur votre système ?

Le moyen le plus simple de le savoir est d'afficher les lignes correspondant aux comptes qui n'utilisent pas le shell Bash :

```
$ grep -v ":/bin/bash$" /etc/passwd | wc -l
```

- Combien d'utilisateurs et de groupes existent sur votre système (rappelez-vous : utilisez uniquement le fichier `/etc/passwd`) ?

  Le premier champ de chaque ligne de votre fichier `/etc/passwd` est le nom de l'utilisateur, le second est généralement un `x` indiquant que le mot de passe de l'utilisateur n'est pas stocké ici (il est chiffré dans le fichier `/etc/shadow`). Le troisième est l'identifiant de l'utilisateur (UID) et le quatrième est l'identifiant du groupe (GID). Ceci devrait donc nous donner le nombre d'utilisateurs :

  ```
  $ cut -d: -f3 /etc/passwd | wc -l
  ```

  Disons que c'est vrai dans la plupart des cas. Cependant, il y a des situations où vous définirez différents super-utilisateurs ou d'autres types d'utilisateurs spéciaux partageant le même UID. Donc, pour être sûr, nous allons passer le résultat de notre commande `cut` à la commande `sort` et ensuite compter le nombre de lignes.

  ```
  $ cut -d: -f3 /etc/passwd | sort -u | wc -l
  ```

  Maintenant, pour le nombre de groupes :

  ```
  $ cut -d: -f4 /etc/passwd | sort -u | wc -l
  ```

- Affichez uniquement la première ligne, la dernière ligne et la dixième ligne de votre fichier `/etc/passwd`.

  Cela fera l'affaire :

  ```
  $ sed -n -e '1'p -e '10'p -e '$'p /etc/passwd
  ```

  Rappelez-vous que le paramètre `-n` indique à `sed` de ne pas imprimer autre chose que ce qui est spécifié par la commande `p`. Le signe dollar (`$`) utilisé ici est une expression régulière signifiant la dernière ligne du fichier.

3. Prenons cet exemple de fichier `/etc/passwd`. Recopiez les lignes ci-dessous dans un fichier local nommé `mypasswd` pour cet exercice.

```
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
nvidia-persistenced:x:121:128:NVIDIA Persistence Daemon,,,:/nonexistent:/sbin/nologin
libvirt-qemu:x:64055:130:Libvirt Qemu,,,:/var/lib/libvirt:/usr/sbin/nologin
libvirt-dnsmasq:x:122:133:Libvirt Dnsmasq,,,:/var/lib/libvirt/dnsmasq:/usr/sbin/nologin
carol:x:1000:2000:Carol Smith,Finance,,,Main Office:/home/carol:/bin/bash
dave:x:1001:1000:Dave Edwards,Finance,,,Main Office:/home/dave:/bin/ksh
emma:x:1002:1000:Emma Jones,Finance,,,Main Office:/home/emma:/bin/bash
frank:x:1003:1000:Frank Cassidy,Finance,,,Main Office:/home/frank:/bin/bash
grace:x:1004:1000:Grace Kearns,Engineering,,,Main Office:/home/grace:/bin/ksh
henry:x:1005:1000:Henry Adams,Sales,,,Main Office:/home/henry:/bin/bash
john:x:1006:1000:John Chapel,Sales,,,Main Office:/home/john:/bin/bash
```

   - Affichez tous les utilisateurs du groupe `1000` (utilisez `sed` pour sélectionner le champ approprié) depuis votre fichier `mypasswd`.

     Le GID est le quatrième champ du fichier `/etc/passwd`. Vous pourriez être tenté d'essayer ceci :

     ```
     $ sed -n /1000/p mypasswd
     ```

     Dans ce cas, vous obtiendrez également cette ligne :

     ```
     carol:x:1000:2000:Carol Smith,Finance,,,Main Office:/home/carol:/bin/bash
     ```

     Vous savez que ce n'est pas correct puisque Carol Smith est membre du GID 2000 et que la correspondance a été établie grâce à l'UID. Cependant, vous avez peut-être remarqué qu'après le GID, le champ suivant commence par une majuscule. Nous pouvons utiliser une expression régulière pour résoudre ce problème.

     ```
     $ sed -n /:1000:[A-Z]/p mypasswd
     ```

     L'expression `[A-Z]` correspondra à n'importe quel caractère majuscule unique. Vous en

apprendrez davantage dans la leçon correspondante.

- Affichez uniquement les noms complets de tous les utilisateurs de ce groupe (utilisez `sed` et `cut`).

  Utilisez la même technique que vous avez utilisée pour résoudre la première partie de cet exercice et envoyez le résultat vers une commande `cut`.

```
$ sed -n /:1000:[A-Z]/p mypasswd | cut -d: -f5
Dave Edwards,Finance,,,Main Office
Emma Jones,Finance,,,Main Office
Frank Cassidy,Finance,,,Main Office
Grace Kearns,Engineering,,,Main Office
Henry Adams,Sales,,,Main Office
John Chapel,Sales,,,Main Office
```

  Nous y sommes presque ! Notez que les champs dans les résultats peuvent être séparés par des virgules `,`. Nous allons donc renvoyer la sortie vers une autre commande `cut` en utilisant la virgule `,` comme délimiteur.

```
$ sed -n /:1000:[A-Z]/p mypasswd | cut -d: -f5 | cut -d, -f1
Dave Edwards
Emma Jones
Frank Cassidy
Grace Kearns
Henry Adams
John Chapel
```

Réponses aux exercices d'approfondissement

1. En utilisant à nouveau le fichier `mypasswd` des exercices ci-dessus, trouvez une commande Bash qui sélectionnera au hasard une personne du bureau principal (`Main Office`) pour gagner une tombola. Utilisez la commande `sed` pour afficher uniquement les lignes du bureau principal, puis une séquence de commandes `cut` pour extraire le prénom de chaque utilisateur de ces lignes. Enfin, vous allez trier ces noms au hasard et afficher uniquement le premier nom de la liste.

   Tout d'abord, voyez l'effet de l'option `-R` sur la sortie de la commande `sort`. Répétez cette commande plusieurs fois sur votre machine (notez que vous devrez mettre 'Main Office' entre guillemets simples pour que `sed` le traite comme une seule chaîne) :

   ```
   $ sed -n /'Main Office'/p mypasswd | cut -d: -f5 | cut -d, -f1 | sort -R
   ```

   Voici une solution à ce problème :

   ```
   $ sed -n /'Main Office'/p mypasswd | cut -d: -f5 | cut -d, -f1 | sort -R | head -1
   ```

2. Combien de personnes travaillent dans la finance (`Finance`), l'ingénierie (`Engineering`) et les ventes (`Sales`) ? (Pensez à utiliser la commande `uniq`).

   Développez ce que vous avez appris dans les exercices précédents. Essayez ceci :

   ```
   $ sed -n /'Main Office'/p mypasswd
   $ sed -n /'Main Office'/p mypasswd | cut -d, -f2
   ```

   Notez à présent que nous ne nous soucions pas du caractère `:` comme délimiteur. Nous recherchons juste le deuxième champ lorsque nous séparons les lignes par les virgules `,`.

   ```
   $ sed -n /'Main Office'/p mypasswd | cut -d, -f2 | uniq -c
         4 Finance
         1 Engineering
         2 Sales
   ```

   La commande `uniq` n'affiche que les lignes uniques (pas les lignes répétées) et l'option `-c` indique à `uniq` de compter les occurrences de lignes identiques. Faites bien attention : `uniq` ne prend en compte que les lignes adjacentes. Dans le cas contraire, vous devrez utiliser la

commande `sort`.

3. Maintenant vous souhaitez préparer un fichier CSV (*Comma Separated Values*) afin de pouvoir facilement importer, depuis le fichier `mypasswd` de l'exemple précédent, le fichier `names.csv` dans LibreOffice. Le contenu du fichier aura le format suivant :

```
First Name,Last Name,Position
Carol,Smith,Finance
...
John,Chapel,Sales
```

Astuce : Utilisez les commandes `sed`, `cut`, et `paste` pour obtenir le résultat souhaité. Notez que la virgule (`,`) servira de délimiteur pour ce fichier.

Commencez avec les commandes `sed` et `cut`, en vous basant sur ce que nous avons appris dans les exercices précédents :

```
$ sed -n /'Main Office'/p mypasswd | cut -d: -f5 | cut -d" " -f1 > firstname
```

Nous voilà avec le fichier `firstname` qui contient les prénoms de nos employés.

```
$ sed -n /'Main Office'/p mypasswd | cut -d: -f5 | cut -d" " -f2 | cut -d, -f1 > lastname
```

Maintenant nous avons le fichier `lastname` avec les noms de famille de chaque employé.

Ensuite, nous allons chercher à savoir dans quel département travaille chaque employé :

```
$ sed -n /'Main Office'/p mypasswd | cut -d: -f5 | cut -d, -f2 > department
```

Avant de travailler sur la solution définitive, essayez les commandes suivantes pour voir quel type de résultat elles génèrent :

```
$ cat firstname lastname department
$ paste firstname lastname department
```

Et voici le résultat final :

```
$ paste firstname lastname department | tr '\t' ,
```

```
$ paste firstname lastname department | tr '\t' , > names.csv
```

Ici, nous utilisons la commande `tr` pour *traduire* `\t`, le séparateur de tabulations, par une virgule. `tr` est très utile lorsque nous devons remplacer un caractère par un autre. N'hésitez pas à consulter les pages de manuel de `tr` et `paste`. Par exemple, nous pouvons utiliser l'option `-d` pour le délimiteur afin de simplifier la commande précédente :

```
$ paste -d, firstname lastname department
```

Nous avons utilisé la commande `paste` ici pour vous familiariser avec elle. Cependant, nous aurions pu facilement effectuer toutes les tâches en un seul enchaînement de commandes :

```
$ sed -n /'Main Office'/p mypasswd | cut -d: -f5 | cut -d, -f1,2 | tr ' ' , > names.csv
```

4. Admettons que la feuille de calcul `names.csv` créée dans l'exercice précédent soit un fichier important et que nous voulions nous assurer que personne ne puisse l'altérer entre le moment où nous l'envoyons à quelqu'un et le moment où notre destinataire le reçoit. Comment assurer l'intégrité de ce fichier en utilisant `md5sum` ?

   Si vous consultez les pages de manuel de `md5sum`, `sha256sum` et `sha512sum`, vous verrez qu'elles commencent toutes par le même texte :

   "calcule et vérifie la somme de contrôle XXX d'un fichier"

   Où "XXX" représente l'algorithme qui sera utilisé pour calculer cette *somme de contrôle*.

   Nous allons utiliser `md5sum` pour l'exemple et par la suite vous pourrez essayer avec les autres commandes.

   ```
   $ md5sum names.csv
   61f0251fcab61d9575b1d0cbf0195e25  names.csv
   ```

   Vous pouvez maintenant, à titre d'exemple, mettre le fichier à disposition par le biais d'un service FTP sécurisé et envoyer la somme de contrôle générée par un autre moyen de communication sécurisé. Si le fichier a été légèrement modifié, la somme de contrôle du fichier sera complètement différente. En guise d'illustration, éditez `names.csv` et changez James en Jones comme nous le faisons ici :

   ```
   $ sed -i.backup s/James/Jones/ names.csv
   ```

```
$ md5sum names.csv
f44a0d68cb480466099021bf6d6d2e65  names.csv
```

Lorsque vous mettez des fichiers à disposition pour le téléchargement, c'est toujours une bonne pratique de distribuer en même temps une somme de contrôle correspondante afin que les personnes qui téléchargent votre fichier puissent produire une nouvelle somme de contrôle et la vérifier par rapport à l'original. Si vous parcourez https://kernel.org, vous trouverez la page https://mirrors.edge.kernel.org/pub/linux/kernel/v5.x/sha256sums.asc où vous pouvez obtenir les sommes de contrôle SHA256 pour tous les fichiers disponibles au téléchargement.

5. Vous vous êtes promis de lire un livre classique à raison de 100 lignes par jour et vous avez décidé de commencer par *Mariner and Mystic* de Herman Melville. Imaginez une commande en utilisant `split` pour séparer ce livre en segments de 100 lignes chacun. Pour obtenir le livre en format texte simple, recherchez-le sur https://www.gutenberg.org.

   Tout d'abord, nous allons récupérer le livre dans son intégralité sur le site du Projet Gutenberg, où vous pouvez télécharger ce livre et d'autres ouvrages disponibles dans le domaine public.

   ```
   $ wget https://www.gutenberg.org/files/50461/50461-0.txt
   ```

   Vous devrez peut-être installer `wget` s'il n'est pas déjà présent sur votre système. Alternativement, vous pouvez aussi vous servir de `curl`. Utilisez `less` pour vérifier le livre :

   ```
   $ less 50461-0.txt
   ```

   Nous allons maintenant découper le livre en plusieurs morceaux de 100 lignes chacun :

   ```
   $ split -l 100 -d 50461-0.txt melville
   ```

   `50461-0.txt` est le fichier que nous allons découper. `melville` sera le préfixe pour les fichiers scindés. L'option `-l 100` spécifie le nombre de lignes et l'option `-d` indique à `split` de numéroter les fichiers (en utilisant le suffixe fourni). Vous pouvez utiliser `nl` sur n'importe lequel des fichiers découpés (probablement pas sur le dernier) et confirmer que chacun d'entre eux compte une centaine de lignes.

6. En utilisant `ls -l` sur le répertoire `/etc`, quel genre d'affichage obtenez-vous ? En utilisant la commande `cut` sur le résultat de la commande `ls` donnée, comment afficheriez-vous uniquement les noms des fichiers ? Qu'en est-il du nom de fichier et du propriétaire du fichier ? En plus des commandes `ls -l` et `cut`, utilisez la commande `tr` pour compacter

plusieurs occurrences d'un espace en un seul espace afin de faciliter le formatage de la sortie avec une commande `cut`.

La commande `ls` en elle-même vous fournira juste les noms des fichiers. Nous pouvons cependant préparer le résultat de la commande `ls -l` (la liste détaillée) pour extraire des informations plus spécifiques.

```
$ ls -l /etc | tr -s ' ' ,
drwxr-xr-x,3,root,root,4096,out,24,16:58,acpi
-rw-r--r--,1,root,root,3028,dez,17,2018,adduser.conf
-rw-r--r--,1,root,root,10,out,2,17:38,adjtime
drwxr-xr-x,2,root,root,12288,out,31,09:40,alternatives
-rw-r--r--,1,root,root,401,mai,29,2017,anacrontab
-rw-r--r--,1,root,root,433,out,1,2017,apg.conf
drwxr-xr-x,6,root,root,4096,dez,17,2018,apm
drwxr-xr-x,3,root,root,4096,out,24,16:58,apparmor
drwxr-xr-x,9,root,root,4096,nov,6,20:20,apparmor.d
```

L'option `-s` indique à `tr` de réduire les espaces consécutifs en un seul espace. La commande `tr` fonctionne pour tout type de caractère répétitif que vous spécifiez. Ensuite, nous remplaçons les espaces par une virgule `,`. Nous n'avons pas besoin de remplacer les espaces dans notre exemple, nous allons donc simplement omettre les `,`.

```
$ ls -l /etc | tr -s ' '
drwxr-xr-x 3 root root 4096 out 24 16:58 acpi
-rw-r--r-- 1 root root 3028 dez 17 2018 adduser.conf
-rw-r--r-- 1 root root 10 out 2 17:38 adjtime
drwxr-xr-x 2 root root 12288 out 31 09:40 alternatives
-rw-r--r-- 1 root root 401 mai 29 2017 anacrontab
-rw-r--r-- 1 root root 433 out 1 2017 apg.conf
drwxr-xr-x 6 root root 4096 dez 17 2018 apm
drwxr-xr-x 3 root root 4096 out 24 16:58 apparmor
```

Si je ne veux que les noms de fichiers, il suffit d'afficher le neuvième champ :

```
$ ls -l /etc | tr -s ' ' | cut -d" " -f9
```

Pour le nom du fichier et son propriétaire, nous aurons besoin du neuvième et du troisième champ :

```
$ ls -l /etc | tr -s ' ' | cut -d" " -f9,3
```

Et si nous avons juste besoin du nom des répertoires et de leur propriétaire ?

```
$ ls -l /etc | grep ^d | tr -s ' ' | cut -d" " -f9,3
```

7. Cet exercice suppose que vous travaillez sur une machine réelle (pas une machine virtuelle). Vous devez également vous munir d'une clé USB. Consultez les pages de manuel de la commande `tail` et voyez comment faire pour lire un fichier à la volée au fur et à mesure que du texte y est ajouté. Tout en surveillant la sortie de la commande `tail` sur le fichier `/var/log/syslog`, insérez une clé USB. Saisissez la commande complète que vous utiliseriez pour obtenir le produit (`Product`), le fabricant (`Manufacturer`) et la quantité totale de mémoire de votre clé USB.

```
$ tail -f /var/log/syslog | grep -i 'product\:\|blocks\|manufacturer'
Nov  8 06:01:35 brod-avell kernel: [124954.369361] usb 1-4.3: Product: Cruzer Blade
Nov  8 06:01:35 brod-avell kernel: [124954.369364] usb 1-4.3: Manufacturer: SanDisk
Nov  8 06:01:37 brod-avell kernel: [124955.419267] sd 2:0:0:0: [sdc] 61056064 512-byte
logical blocks: (31.3 GB/29.1 GiB)
```

Bien entendu, ceci n'est qu'un exemple et les résultats peuvent varier en fonction du fabricant de votre clé USB. Notez l'utilisation de l'option `-i` avec la commande `grep` car nous ne sommes pas sûrs que les chaînes de caractères que nous recherchons soient en majuscules ou en minuscules. Nous avons également utilisé le `|` comme un OU logique. Nous recherchons donc les lignes contenant `product` OU `blocks` OU `manufacturer`.

103.3 Gestion élémentaire des fichiers

Référence aux objectifs de LPI

LPIC-1 v5, Exam 101, Objective 103.3

Valeur

4

Domaines de connaissance les plus importants

- Copie, déplacement et suppression des fichiers ou des répertoires individuellement.
- Copie récursive de plusieurs fichiers et répertoires.
- Suppression récursive de fichiers et répertoires.
- Utilisation simple et avancée des caractères génériques (wildcard) dans les commandes.
- Utilisation de find pour localiser et agir sur des fichiers en se basant sur leurs types, leurs tailles ou leurs temps (de création, modification ou accès).
- Utilisation des commandes tar, cpio et dd.

Liste partielle de termes, fichiers et utilitaires utilisés pour cet objectif

- `cp`
- `find`
- `mkdir`
- `mv`
- `ls`
- `rm`
- `rmdir`
- `touch`

- `tar`
- `cpio`
- `dd`
- `file`
- `gzip`
- `gunzip`
- `bzip2`
- `bunzip2`
- Développement des noms de fichiers (file globbing)

103.3 Leçon 1

Certification :	LPIC-1
Version :	5.0
Thème :	103 Commandes GNU et Unix
Objectif :	103.3 Gestion élémentaire des fichiers
Leçon :	1 sur 2

Introduction

Tout est fichier sous Linux, il est donc très important de savoir comment les manipuler. Dans cette leçon, nous allons aborder les opérations de base sur les fichiers.

En règle générale, un utilisateur de Linux sera amené à naviguer dans le système de fichiers, à copier des fichiers d'un emplacement vers un autre et à supprimer des fichiers. Nous verrons également les commandes liées à la gestion des fichiers.

Un fichier est une entité qui contient des données et des programmes. Il est constitué du contenu à proprement parler et des métadonnées (taille du fichier, propriétaire, date de création, permissions). Les fichiers sont organisés en répertoires. Un répertoire est un fichier qui contient d'autres fichiers.

Les différents types de fichiers comprennent :

Les fichiers normaux

qui stockent les données et les programmes.

Les répertoires

qui contiennent d'autres fichiers.

Les fichiers spéciaux

qui sont utilisés pour les entrées et les sorties.

Bien entendu, d'autres types de fichiers existent, mais ils dépassent le cadre de cette leçon. Nous verrons plus tard comment identifier ces différents types de fichiers.

Manipuler les fichiers

Afficher les fichiers avec `ls`

La commande `ls` fait partie des principaux outils en ligne de commande que vous devez apprendre pour naviguer dans le système de fichiers.

Dans son fonctionnement de base, `ls` affiche *uniquement* les noms des fichiers et des répertoires :

```
$ ls
Desktop Downloads   emp_salary  file1   Music   Public  Videos
Documents   emp_name    examples.desktop    file2   Pictures    Templates
```

L'option `-l` désigne le format d'affichage long et nous détaille les permissions des fichiers et des répertoires, le propriétaire, la taille, la date et l'heure de dernière modification ainsi que le nom :

```
$ ls -l
total 60
drwxr-xr-x  2   frank   frank   4096    Apr 1   2018    Desktop
drwxr-xr-x  2   frank   frank   4096    Apr 1   2018    Documents
drwxr-xr-x  2   frank   frank   4096    Apr 1   2018    Downloads
-rw-r--r--  1   frank   frank     21    Sep 7   12:59   emp_name
-rw-r--r--  1   frank   frank     20    Sep 7   13:03   emp_salary
-rw-r--r--  1   frank   frank   8980    Apr 1   2018    examples.desktop
-rw-r--r--  1   frank   frank     10    Sep 1   2018    file1
-rw-r--r--  1   frank   frank     10    Sep 1   2018    file2
drwxr-xr-x  2   frank   frank   4096    Apr 1   2018    Music
drwxr-xr-x  2   frank   frank   4096    Apr 1   2018    Pictures
drwxr-xr-x  2   frank   frank   4096    Apr 1   2018    Public
drwxr-xr-x  2   frank   frank   4096    Apr 1   2018    Templates
drwxr-xr-x  2   frank   frank   4096    Apr 1   2018    Videos
```

Le tout premier caractère de la liste nous renseigne sur le type des fichiers :

-

pour un fichier normal.

d

pour un répertoire.

c

pour un fichier spécial.

Pour afficher la taille des fichiers dans un format humainement lisible, ajoutez l'option `-h` :

```
$ ls -lh
total 60K
drwxr-xr-x  2   frank   frank   4.0K    Apr 1   2018    Desktop
drwxr-xr-x  2   frank   frank   4.0K    Apr 1   2018    Documents
drwxr-xr-x  2   frank   frank   4.0K    Apr 1   2018    Downloads
-rw-r--r--  1   frank   frank     21    Sep 7   12:59   emp_name
-rw-r--r--  1   frank   frank     20    Sep 7   13:03   emp_salary
-rw-r--r--  1   frank   frank   8.8K    Apr 1   2018    examples.desktop
-rw-r--r--  1   frank   frank     10    Sep 1   2018    file1
-rw-r--r--  1   frank   frank     10    Sep 1   2018    file2
drwxr-xr-x  2   frank   frank   4.0K    Apr 1   2018    Music
drwxr-xr-x  2   frank   frank   4.0K    Apr 1   2018    Pictures
drwxr-xr-x  2   frank   frank   4.0K    Apr 1   2018    Public
drwxr-xr-x  2   frank   frank   4.0K    Apr 1   2018    Templates
drwxr-xr-x  2   frank   frank   4.0K    Apr 1   2018    Videos
```

Pour afficher tous les fichiers, y compris les fichiers cachés (ceux qui commencent par `.`), utilisez l'option `-a` :

```
$ ls -a
.               .dbus   file1   .profile
..              Desktop file2   Public
.bash_history   .dmrc   .gconf  .sudo_as_admin_successful
```

Les fichiers de configuration comme `.bash_history` qui sont masqués par défaut sont maintenant visibles.

En règle générale, la syntaxe de la commande `ls` suit le schéma :

```
ls OPTIONS FILE
```

Où `OPTIONS` correspond à l'une des options présentées ci-dessus (pour afficher toutes les options possibles, invoquez `man ls`), et `FILE` est le nom du fichier ou du répertoire dont vous souhaitez obtenir les informations.

NOTE Lorsque `FILE` n'est pas spécifié, c'est le répertoire courant qui est utilisé.

Créer, copier, déplacer et supprimer des fichiers

Créer des fichiers avec `touch`

La commande `touch` est le moyen le plus simple de créer de nouveaux fichiers vides. Vous pouvez également l'utiliser pour modifier l'horodatage (c'est-à-dire la date et l'heure de la dernière modification) des fichiers et des répertoires existants. La syntaxe pour utiliser `touch` est :

```
touch OPTIONS FILE_NAME(S)
```

Invoqué sans option, `touch` va créer des fichiers pour tous les noms de fichiers fournis en argument, à condition que ces fichiers n'existent pas déjà. `touch` peut créer plusieurs fichiers simultanément :

```
$ touch file1 file2 file3
```

La commande ci-dessus crée trois nouveaux fichiers vides nommés respectivement `file1`, `file2` et `file3`.

Certaines options de `touch` sont spécialement conçues pour permettre à l'utilisateur de modifier les horodatages des fichiers. À titre d'exemple, l'option `-a` ne modifie que l'horodatage d'accès, tandis que l'option `-m` ne change que l'horodatage de modification. Invoquées conjointement, les deux options modifient l'horodatage d'accès et de modification en fonction de l'heure actuelle :

```
$ touch -am file3
```

Copier des fichiers avec `cp`

Un utilisateur de Linux est souvent amené à copier des fichiers d'un endroit vers un autre. Qu'il s'agisse de déplacer un fichier audio d'un répertoire vers un autre ou un fichier système, utilisez

`cp` pour toutes les opérations de copie :

```
$ cp file1 dir2
```

Cette commande peut être littéralement interprétée comme copier `file1` dans le répertoire `dir2`. Il en résulte la présence de `file1` dans `dir2`. Pour que cette commande soit exécutée avec succès, `file1` doit exister dans le répertoire courant de l'utilisateur. Dans le cas contraire, le système affiche le message d'erreur `No such file or directory`.

```
$ cp dir1/file1 dir2
```

Dans ce cas, observez que le chemin vers `file1` est plus explicite. Le chemin de la source peut être indiqué sous forme de chemin *relatif* ou *absolu*. Les chemins relatifs sont indiqués en référence à un répertoire spécifique, tandis que les chemins absolus ne sont pas indiqués avec une référence. Nous allons préciser cette notion un peu plus loin.

Pour l'instant, notez simplement que la commande copie `file1` dans le répertoire `dir2`. Le chemin vers `file1` est fourni avec plus de précision puisque l'utilisateur ne se trouve pas actuellement dans `dir1`.

```
$ cp /home/frank/Documents/file2 /home/frank/Documents/Backup
```

Dans ce troisième figure, le fichier `file2` rangé dans `/home/frank/Documents` est copié vers le répertoire `/home/frank/Documents/Backup`. Le chemin de la source fourni en argument ici est *absolu*. Dans les deux exemples précédents, les chemins des sources sont *relatifs*. Lorsqu'un chemin commence par le caractère `/`, c'est un chemin absolu, sinon c'est un chemin relatif.

La syntaxe générale de `cp` est :

```
cp OPTIONS SOURCE DESTINATION
```

`SOURCE` est le fichier à copier et `DESTINATION` le répertoire vers lequel le fichier devra être copié. `SOURCE` et `DESTINATION` peuvent être spécifiés comme des chemins absolus ou relatifs.

Déplacer des fichiers avec `mv`

Tout comme `cp` pour la copie, Linux fournit une commande pour déplacer et renommer des fichiers. Elle s'appelle `mv`.

L'opération de déplacement est analogue à l'opération couper-coller que vous pouvez effectuer dans une interface graphique (GUI).

Si vous souhaitez déplacer un fichier vers un nouvel emplacement, utilisez `mv` de la manière suivante :

```
mv FILENAME DESTINATION_DIRECTORY
```

Voici un exemple :

```
$ mv myfile.txt /home/frank/Documents
```

Il en résulte que `myfile.txt` est déplacé vers la destination `/home/frank/Documents`.

Pour renommer un fichier, on utilise `mv` de la manière suivante :

```
$ mv old_file_name new_file_name
```

Cette opération change le nom du fichier de `old_file_name` en `new_file_name`.

Dans la configuration par défaut, `mv` ne vous demandera pas de confirmer si vous souhaitez écraser (renommer) un fichier existant. Vous pouvez toutefois faire en sorte que le système vous affiche une demande de confirmation, en utilisant l'option `-i` :

```
$ mv -i old_file_name new_file_name
mv: overwrite 'new_file_name'?
```

Cette commande demanderait la permission de l'utilisateur avant d'écraser `old_file_name` et le remplacer par `new_file_name`.

Inversement, utilisez l'option `-f`:

```
$ mv -f old_file_name new_file_name
```

pour forcer l'écrasement du fichier sans afficher une demande de confirmation.

Supprimer des fichiers avec `rm`

`rm` est utilisé pour supprimer des fichiers. Pensez-y comme une forme abrégée du mot "remove".

Notez que la suppression d'un fichier est généralement irréversible et que cette commande doit donc être utilisée avec précaution.

```
$ rm file1
```

Cette commande supprimerait `file1`.

```
$ rm -i file1
rm: remove regular file 'file1'?
```

Cette commande demanderait confirmation à l'utilisateur avant de supprimer `file1`. Souvenez-vous, nous venons de voir l'option `-i` en utilisant `mv` ci-dessus.

```
$ rm -f file1
```

Cette commande force la suppression de `file1` sans vous demander la moindre confirmation.

Plusieurs fichiers peuvent être supprimés en même temps :

```
$ rm file1 file2 file3
```

Dans cet exemple, `file1`, `file2` et `file3` sont supprimés simultanément.

La syntaxe de `rm` est généralement donnée par :

```
rm OPTIONS FILE
```

Créer et supprimer des répertoires

Créer des répertoires avec `mkdir`

Créer des répertoires est essentiel pour organiser vos fichiers et vos dossiers. Les fichiers pourront être regroupés de manière cohérente en les rangeant dans un répertoire. Pour créer un répertoire, utilisez `mkdir` :

```
mkdir OPTIONS DIRECTORY_NAME
```

Ici, `DIRECTORY_NAME` est le nom du répertoire à créer. Vous pouvez créer plusieurs répertoires d'une traite :

```
$ mkdir dir1
```

crée le répertoire `dir1` dans le répertoire courant de l'utilisateur.

```
$ mkdir dir1 dir2 dir3
```

La commande précédente permet de créer trois répertoires `dir1`, `dir2` et `dir3` en même temps.

Pour créer un répertoire avec ses sous-répertoires, utilisez l'option `-p` ("parents") :

```
$ mkdir -p parents/children
```

Cette commande crée la structure de répertoires `parents/children`, c'est-à-dire qu'elle crée les répertoires `parents` et `children`. `children` sera situé à l'intérieur de `parents`.

Supprimer des répertoires avec `rmdir`

`rmdir` supprime un répertoire *s'il est vide*. Sa syntaxe est définie ainsi :

```
rmdir OPTIONS DIRECTORY
```

Ici, `DIRECTORY` peut être un argument unique ou une liste d'arguments.

```
$ rmdir dir1
```

Cette commande supprimerait `dir1`.

```
$ rmdir dir1 dir2
```

Cette commande supprimerait simultanément `dir1` et `dir2`.

Vous pouvez très bien supprimer un répertoire avec son sous-répertoire :

```
$ rmdir -p parents/children
```

Cela supprimera la structure de répertoires `parents/children`. Notez que si l'un des répertoires n'est pas vide, ils ne seront pas supprimés.

Gestion récursive des fichiers et des répertoires

Pour manipuler un répertoire et son contenu, vous devez appliquer la *récursion*. La récursion signifie qu'il faut effectuer une action et la répéter tout le long de l'arborescence des répertoires. Sous Linux, les options `-r` ou `-R` ou `--recursive` sont généralement associées à la récursion.

Le cas de figure suivant vous aidera à mieux comprendre la récursion :

Vous affichez le contenu d'un répertoire `students`, qui contient deux sous-répertoires `level 1` et `level 2` ainsi que le fichier nommé `frank`. En appliquant la récursion, la commande `ls` affichera le contenu de `students`, c'est-à-dire `level 1`, `level 2` et `frank`, mais elle ne s'arrêtera pas là. Elle entrera également dans les sous-répertoires `level 1` et `level 2` et affichera leur contenu, et ainsi de suite dans l'arborescence des répertoires.

Affichage récursif avec `ls -R`

`ls -R` est utilisé pour afficher le contenu d'un répertoire avec ses sous-répertoires et ses fichiers.

```
$ ls -R mydirectory
mydirectory/:
file1   newdirectory

mydirectory/newdirectory:
```

Dans l'exemple ci-dessus, `mydirectory` ainsi que tout son contenu sont affichés. Vous pouvez observer que `mydirectory` contient le sous-répertoire `newdirectory` ainsi que le fichier `file1`. `newdirectory` est vide, c'est pourquoi aucun contenu n'est affiché.

En règle générale, pour afficher le contenu d'un répertoire avec tous ses sous-répertoires, utilisez :

```
ls -R DIRECTORY_NAME
```

L'ajout d'une barre oblique finale à `DIRECTORY_NAME` n'a aucun effet :

```
$ ls -R animal
```

équivaut à

```
$ ls -R animal/
```

Copie récursive avec `cp -r`

`cp -r` (ou `-R` ou `--recursive`) vous permet de copier un répertoire avec tous ses sous-répertoires et tous ses fichiers.

```
$ tree mydir
mydir
|_file1
|_newdir
    |_file2
    |_insidenew
        |_lastdir


3 directories, 2 files
$ mkdir newcopy
$ cp mydir newcopy
cp: omitting directory 'mydir'
$ cp -r mydir newcopy
* tree newcopy
newcopy
|_mydir
    |_file1
    |_newdir
        |_file2
        |_insidenew
            |_lastdir

4 directories, 2 files
```

Dans l'exemple ci-dessus, nous constatons qu'en essayant de copier `mydir` dans `newcopy` en utilisant `cp` sans `-r`, le système affiche le message `cp : omitting directory 'mydir'` (`omission du répertoire 'mydir'`). En revanche, en ajoutant l'option `-r`, tout le contenu de `mydir`, y compris le répertoire lui-même, est copié vers `newcopy`.

Pour copier les répertoires et sous-répertoires, utilisez :

```
cp -r SOURCE DESTINATION
```

Suppression récursive avec `rm -r`

`rm -r` va supprimer un répertoire et tout son contenu (sous-répertoires et fichiers).

WARNING

Soyez très vigilants avec l'option `-r` et les options combinées `-rf` lorsqu'elles sont utilisées conjointement avec la commande `rm`. Une commande de suppression récursive sur un répertoire système important pourrait rendre le système inutilisable. Utilisez la commande de suppression récursive uniquement lorsque vous êtes absolument certain que le contenu d'un répertoire peut être supprimé sans danger de l'ordinateur.

Lorsqu'on essaie de supprimer un répertoire sans utiliser l'option `-r`, le système affiche une erreur :

```
$ rm newcopy/
rm: cannot remove 'newcopy/': Is a directory
$ rm -r newcopy/
```

Il vous faut ajouter l'option `-r` comme dans le deuxième exemple pour que la suppression soit prise en compte.

NOTE

Vous vous demandez peut-être pourquoi nous n'utilisons pas `rmdir` dans ce cas. Il y a une différence subtile entre les deux commandes. `rmdir` ne réussira à effectuer une suppression que si le répertoire donné est vide alors que `rm -r` peut être utilisé indépendamment du fait que le répertoire soit vide ou non.

Ajouter l'option `-i` pour obtenir une confirmation avant la suppression du répertoire :

```
$ rm -ri mydir/
rm: remove directory 'mydir/'?
```

Le système demande une confirmation avant d'essayer de supprimer `mydir`.

Filtres sur les fichiers et caractères de substitution

Le *globbing* de fichiers est une fonctionnalité fournie par l'interpréteur de commandes Unix/Linux pour représenter plusieurs noms de fichiers en utilisant des caractères de substitution appelés *jokers*. Les caractères de substitution sont essentiellement des symboles qui peuvent être utilisés pour remplacer un ou plusieurs caractères. Ils permettent, par exemple, d'afficher tous les fichiers qui commencent par la lettre `A` ou alors tous ceux qui se terminent par les lettres `.conf`.

Les caractères de substitution sont très utiles car ils peuvent être utilisés avec des commandes telles que `cp`, `ls` ou `rm`.

Voici quelques exemples de filtres sur les fichiers (*globbing*) :

`rm *`
: Supprimer tous les fichiers du répertoire courant.

`ls l?st`
: Afficher tous les fichiers dont le nom commence par `l` suivi d'un caractère unique quelconque et qui se termine par `st`.

`rmdir [a-z]*`
: Supprimer tous les répertoires dont le nom commence par une lettre.

Types de caractères de substitution

Il y a trois caractères qui peuvent être utilisés comme jokers sous Linux :

`*` (astérisque)
: qui représente zéro, une ou plusieurs occurrences d'un caractère quelconque.

`?` (point d'interrogation)
: qui représente une seule occurrence de n'importe quel caractère.

`[ ]` (caractères entre crochets)
: qui représente toute occurrence du ou des caractères contenus dans les crochets. Il est possible d'utiliser différents types de caractères, qu'il s'agisse de chiffres, de lettres ou d'autres caractères spéciaux. Par exemple, l'expression `[0-9]` correspond à tous les chiffres.

L'astérisque

Un astérisque (`*`) correspond à zéro, une ou plusieurs occurrences de n'importe quel caractère.

Par exemple :

```
$ find /home -name *.png
```

Cette commande trouverait tous les fichiers qui se terminent par `.png` tels que `photo.png`, `cat.png` ou `frank.png`. La commande `find` sera abordée plus en détail dans une leçon

ultérieure.

De même :

```
$ ls lpic-*.txt
```

affichera tous les fichiers texte commençant par les caractères `lpic-` suivis d'un nombre quelconque de caractères et se terminant par `.txt`, tels que `lpic-1.txt` et `lpic-2.txt`.

Le caractère de substitution astérisque peut être utilisé pour manipuler (copier, supprimer ou déplacer) tout le contenu d'un répertoire :

```
$ cp -r animal/* forest
```

Dans cet exemple, tout le contenu de `animal` est copié vers `forest`.

En règle générale, pour copier tout le contenu d'un répertoire, on utilisera :

```
cp -r SOURCE_PATH/* DEST_PATH
```

où `SOURCE_PATH` peut être omis si nous nous trouvons déjà dans le répertoire en question.

L'astérisque, comme n'importe quel autre caractère de substitution, peut être utilisé plusieurs fois dans la même commande et à n'importe quelle position :

```
$ rm *ate*
```

Les fichiers dont le nom est préfixé par zéro, une ou plusieurs occurrences d'un caractère quelconque, suivis des lettres `ate` et se terminant par zéro, une ou plusieurs occurrences d'un caractère quelconque seront supprimés.

Le point d'interrogation

Le point d'interrogation (`?`) correspond à une seule occurrence d'un caractère.

Considérez la liste de fichiers suivante :

```
$ ls
last.txt    lest.txt    list.txt    third.txt   past.txt
```

Pour afficher uniquement les fichiers dont le nom commence par `l` suivi d'un caractère unique et des caractères `st.txt`, nous utilisons le point d'interrogation (`?`) comme caractère de substitution :

```
$ ls l?st.txt
last.txt    lest.txt    list.txt
```

Seuls les fichiers `last.txt`, `lest.txt` et `list.txt` sont affichés car ils correspondent aux critères donnés.

De même,

```
$ ls ??st.txt
last.txt    lest.txt    list.txt    past.txt
```

affiche les fichiers préfixés de deux caractères quelconques suivis de `st.txt`.

Les caractères entre crochets

Les caractères de substitution entre crochets correspondent à n'importe quelle occurrence du ou des caractères placés entre les crochets :

```
$ ls l[aef]st.txt
last.txt    lest.txt
```

Cette commande affichera la liste de tous les fichiers dont le nom commence par `l` suivi de *n'importe quel* caractère de la série `aef` et qui se termine par `st.txt`.

Les crochets peuvent également correspondre à des intervalles :

```
$ ls l[a-z]st.txt
last.txt    lest.txt    list.txt
```

Cette commande affiche tous les fichiers dont le nom commence par `l` suivi d'une lettre minuscule comprise entre `a` et `z` et qui se termine par `st.txt`.

Les séries d'intervalles peuvent également être exprimées à l'aide des crochets :

```
$ ls
```

```
student-1A.txt  student-2A.txt  student-3.txt
$ ls student-[0-9][A-Z].txt
student-1A.text student-2A.txt
```

La liste affiche l'annuaire d'une école avec la liste des élèves inscrits. Nous souhaitons afficher tous les élèves dont le numéro d'inscription répond aux critères suivants :

- commence par `student-`
- suivi d'un chiffre et d'une majuscule
- finit par `.txt`

Combiner les caractères de substitution

Les caractères de substitution peuvent être combinés comme ici :

```
$ ls
last.txt    lest.txt    list.txt    third.txt   past.txt
$ ls [plf]?st*
last.txt    lest.txt    list.txt    past.txt
```

Le premier élément du groupe de caractères de substitution (`[plf]`) correspond à n'importe lequel des caractères `p`, `l` ou `f`. Le deuxième élément du groupe de caractères de substitution (?) correspond à n'importe quel caractère unique. Le troisième élément du groupe de caractères de substitution (*) correspond à zéro, une ou plusieurs occurrences de n'importe quel caractère.

```
$ ls
file1.txt file.txt file23.txt fom23.txt
$ ls f*[0-9].txt
file1.txt file23.txt fom23.txt
```

La commande précédente affiche tous les fichiers dont le nom commence par la lettre `f` suivie d'un ensemble quelconque de lettres et d'au moins une occurrence d'un chiffre et qui se termine par `.txt`. Notez que `file.txt` n'est pas affiché car il ne correspond pas à ces critères.

Exercices guidés

1. Considérez l'affichage ci-dessous :

```
$ ls -lh
total 60K
drwxr-xr-x  2   frank   frank   4.0K    Apr 1   2018    Desktop
drwxr-xr-x  2   frank   frank   4.0K    Apr 1   2018    Documents
drwxr-xr-x  2   frank   frank   4.0K    Apr 1   2018    Downloads
-rw-r--r--  1   frank   frank     21    Sep 7   12:59   emp_name
-rw-r--r--  1   frank   frank     20    Sep 7   13:03   emp_salary
-rw-r--r--  1   frank   frank   8.8K    Apr 1   2018    examples.desktop
-rw-r--r--  1   frank   frank     10    Sep 1   2018    file1
-rw-r--r--  1   frank   frank     10    Sep 1   2018    file2
drwxr-xr-x  2   frank   frank   4.0K    Apr 1   2018    Music
drwxr-xr-x  2   frank   frank   4.0K    Apr 1   2018    Pictures
drwxr-xr-x  2   frank   frank   4.0K    Apr 1   2018    Public
drwxr-xr-x  2   frank   frank   4.0K    Apr 1   2018    Templates
drwxr-xr-x  2   frank   frank   4.0K    Apr 1   2018    Videos
```

   - Que représente le caractère `d` dans les résultats ?

   - Pourquoi les tailles s'affichent-elles dans un format humainement lisible ?

   - Qu'est-ce qui changerait dans l'affichage si `ls` était utilisé sans argument ?

2. Considérez la commande ci-dessous :

```
$ cp /home/frank/emp_name /home/frank/backup
```

   - Que se passera-t-il avec le fichier `emp_name` si cette commande est exécutée avec succès ?

   - Si `emp_name` était un répertoire, quelle option devrait être ajoutée à `cp` pour exécuter la commande ?

- Si `cp` est remplacé par `mv`, à quoi pouvez-vous vous attendre ?

3. Considérez la liste de fichiers suivante :

```
$ ls
file1.txt file2.txt file3.txt file4.txt
```

   Quel caractère de substitution permettrait de supprimer tout le contenu de ce répertoire ?

4. En partant du listing précédent, quels fichiers seraient affichés par la commande suivante ?

```
$ ls file*.txt
```

5. Complétez la commande en ajoutant les chiffres et les caractères appropriés entre les crochets pour énumérer tout le contenu ci-dessus :

```
$ ls file[].txt
```

Exercices d'approfondissement

1. Dans votre répertoire utilisateur, créez deux fichiers `dog` et `cat`.
2. Toujours dans votre répertoire utilisateur, créez le répertoire `animal`. Déplacez `dog` et `cat` vers `animal`.
3. Allez dans le dossier `Documents` qui se trouve dans votre répertoire utilisateur et créez le répertoire `backup` à l'intérieur de celui-ci.
4. Copiez `animal` et tout son contenu vers `backup`.
5. Renommez `animal` dans `backup` en `animal.bkup`.
6. Le répertoire `/home/lpi/databases` contient de nombreux fichiers dont : `db-1.tar.gz`, `db-2.tar.gz` et `db-3.tar.gz`. Quelle commande simple pouvez-vous utiliser pour afficher uniquement les fichiers en question ?

7. Considérez le listing suivant :

```
$ ls
cne1222223.pdf cne12349.txt cne1234.pdf
```

   En utilisant un seul caractère de substitution, quelle commande permettrait de supprimer uniquement les fichiers `.pdf` ?

Résumé

Dans cette leçon, nous avons appris à visualiser le contenu d'un répertoire avec la commande `ls`, à copier (`cp`) des fichiers et des dossiers et à les déplacer (`mv`). Nous avons vu la création de nouveaux répertoires avec la commande `mkdir`. Les commandes pour la suppression des fichiers (`rm`) et des répertoires (`rmdir`) ont également été abordées.

Dans cette leçon, vous avez également appris à connaître le *globbing* des fichiers et les caractères de substitution. Le *globbing* des fichiers est utilisé pour représenter plusieurs noms de fichiers en utilisant des caractères spéciaux appelés *jokers* ou caractères de substitution. Voici les principaux *jokers* et leur signification respective :

`?` (point d'interrogation)
: représente une seule occurrence d'un caractère.

`[ ]` (crochets)
: représentent toute occurrence du ou des caractères contenus entre les crochets.

`*` (astérisque)
: représente zéro, une ou plusieurs occurrences de n'importe quel caractère.

Vous pouvez très bien combiner tous ces *jokers* dans une seule et même instruction.

Réponses aux exercices guidés

1. Considérez l'affichage ci-dessous :

```
$ ls -lh
total 60K
drwxr-xr-x  2   frank   frank   4.0K    Apr 1   2018    Desktop
drwxr-xr-x  2   frank   frank   4.0K    Apr 1   2018    Documents
drwxr-xr-x  2   frank   frank   4.0K    Apr 1   2018    Downloads
-rw-r--r--  1   frank   frank     21    Sep 7   12:59   emp_name
-rw-r--r--  1   frank   frank     20    Sep 7   13:03   emp_salary
-rw-r--r--  1   frank   frank   8.8K    Apr 1   2018    examples.desktop
-rw-r--r--  1   frank   frank     10    Sep 1   2018    file1
-rw-r--r--  1   frank   frank     10    Sep 1   2018    file2
drwxr-xr-x  2   frank   frank   4.0K    Apr 1   2018    Music
drwxr-xr-x  2   frank   frank   4.0K    Apr 1   2018    Pictures
drwxr-xr-x  2   frank   frank   4.0K    Apr 1   2018    Public
drwxr-xr-x  2   frank   frank   4.0K    Apr 1   2018    Templates
drwxr-xr-x  2   frank   frank   4.0K    Apr 1   2018    Videos
```

   - Que représente le caractère `d` dans les résultats ?

     `d` est le caractère qui identifie un répertoire.

   - Pourquoi les tailles s'affichent-elles dans un format humainement lisible ?

     À cause de l'option `-h`.

   - Qu'est-ce qui changerait dans l'affichage si `ls` était utilisé sans argument ?

     Seuls les noms des répertoires et des fichiers seraient indiqués.

2. Considérez la commande ci-dessous :

```
$ cp /home/frank/emp_name /home/frank/backup
```

   - Que se passera-t-il avec le fichier `emp_name` si cette commande est exécutée avec succès ?

     `emp_name` sera copié dans `backup`.

   - Si `emp_name` était un répertoire, quelle option devrait être ajoutée à `cp` pour exécuter la commande ?

`-r`

- Si `cp` est remplacé par `mv`, à quoi pouvez-vous vous attendre ?

  `emp_name` serait déplacé dans `backup`. Il n'existera plus dans le répertoire personnel de l'utilisateur `frank`.

3. Considérez la liste de fichiers suivante :

```
$ ls
file1.txt file2.txt file3.txt file4.txt
```

   Quel caractère de substitution permettrait de supprimer tout le contenu de ce répertoire ?

   L'astérisque `*`.

4. En partant du listing précédent, quels fichiers seraient affichés par la commande suivante ?

```
$ ls file*.txt
```

   Tous, puisque le caractère astérisque représente un nombre quelconque de caractères.

5. Complétez la commande en ajoutant les chiffres et les caractères appropriés entre les crochets pour énumérer tout le contenu ci-dessus :

```
$ ls file[].txt
```

   `file[0-9].txt`

Réponses aux exercices d'approfondissement

1. Dans votre répertoire utilisateur, créez deux fichiers `dog` et `cat`.

```
$ touch dog cat
```

2. Toujours dans votre répertoire utilisateur, créez le répertoire `animal`. Déplacez `dog` et `cat` vers `animal`.

```
$ mkdir animal
$ mv dog cat -t animal/
```

3. Allez dans le dossier `Documents` qui se trouve dans votre répertoire utilisateur et créez le répertoire `backup` à l'intérieur de celui-ci.

```
$ cd ~/Documents
$ mkdir backup
```

4. Copiez `animal` et tout son contenu vers `backup`.

```
$ cp -r animal ~/Documents/backup
```

5. Renommez `animal` dans `backup` en `animal.bkup`.

```
$ mv animal/ animal.bkup
```

6. Le répertoire `/home/lpi/databases` contient de nombreux fichiers dont : `db-1.tar.gz`, `db-2.tar.gz` et `db-3.tar.gz`. Quelle commande simple pouvez-vous utiliser pour afficher uniquement les fichiers en question ?

```
$ ls db-[1-3].tar.gz
```

7. Considérez le listing suivant :

```
$ ls
cne1222223.pdf cne12349.txt cne1234.pdf
```

En utilisant un seul caractère de substitution, quelle commande permettrait de supprimer uniquement les fichiers `.pdf` ?

```
$ rm *.pdf
```

103.3 Leçon 2

Certification :	LPIC-1
Version :	5.0
Thème :	103 Commandes GNU et Unix
Objectif :	103.3 Gestion élémentaire des fichiers
Leçon :	2 sur 2

Introduction

Chercher des fichiers

Au fil de l'utilisation de votre machine, les fichiers augmentent progressivement en nombre et en taille. Il peut arriver qu'il soit difficile de localiser un fichier particulier. Heureusement pour nous, Linux fournit `find` pour rechercher et localiser rapidement les fichiers. `find` utilise la syntaxe suivante :

```
find STARTING_PATH OPTIONS EXPRESSION
```

`STARTING_PATH`

définit le répertoire où débute la recherche.

`OPTIONS`

contrôle le déroulement et ajoute des critères spécifiques pour optimiser le processus de recherche.

`EXPRESSION`
: définit la requête de la recherche.

```
$ find . -name "myfile.txt"
./myfile.txt
```

Ici, le chemin de départ est le répertoire courant. L'option `-name` spécifie que la recherche se base sur le nom du fichier. `myfile.txt` est le nom du fichier à rechercher. Si vous utilisez des caractères de substitution, pensez à mettre l'expression entre guillemets :

```
$ find /home/frank -name "*.png"
/home/frank/Pictures/logo.png
/home/frank/screenshot.png
```

Cette commande trouve tous les fichiers dont le nom se termine par `.png` en partant du répertoire `/home/frank/` et en dessous. Si vous ne comprenez pas l'utilisation de l'astérisque (`*`), elle a été abordée dans la précédente leçon.

Utiliser des critères pour accélérer la recherche

Utilisez `find` pour localiser des fichiers en fonction du *type*, de la *taille* ou du *temps*. Il suffit de spécifier une ou plusieurs options pour que les résultats escomptés soient obtenus plus rapidement.

Les options pour la recherche de fichiers en fonction du type comprennent :

`-type f`
: recherche de fichiers.

`-type d`
: recherche de répertoires.

`-type l`
: recherche de liens symboliques.

```
$ find . -type d -name "example"
```

Cette commande trouve tous les répertoires dans le répertoire courant et en dessous, qui portent le nom `example`.

Parmi les autres critères utilisables avec `find`, on peut citer :

`-name`
: effectue une recherche basée sur le nom fourni.

`-iname`
: recherche basée sur le nom et insensible à la casse (c'est-à-dire que la recherche retournera `myFile` aussi bien que `MYFILE`).

`-not`
: renvoie les résultats qui ne correspondent *pas* au terme recherché.

`-maxdepth N`
: recherche dans le répertoire courant ainsi que dans les sous-répertoires de `N` niveaux de profondeur.

Localiser des fichiers par date de modification

`find` permet également de passer au peigne fin une arborescence de répertoires en fonction de la date de modification du fichier :

```
$ sudo find / -name "*.conf" -mtime 7
/etc/logrotate.conf
```

Cette commande recherche tous les fichiers du système de fichiers (le chemin de départ est le répertoire racine, c'est-à-dire `/`) dont le nom se termine par les caractères `.conf` et qui ont été modifiés au cours des sept derniers jours. La commande nécessite des privilèges élevés pour accéder aux répertoires qui commencent à la base de la structure des répertoires du système, d'où l'utilisation de `sudo` ici. L'argument fourni à `mtime` représente le *nombre de jours* depuis la dernière modification du fichier.

Localiser des fichiers par taille

`find` peut également localiser des fichiers en fonction de leur *taille*. Par exemple, chercher des fichiers d'une taille supérieure à `2G` dans `/var` :

```
$ sudo find /var -size +2G
/var/lib/libvirt/images/debian10.qcow2
/var/lib/libvirt/images/rhel8.qcow2
```

L'option `-size` affiche les fichiers dont la taille correspond à l'argument fourni. Voici quelques exemples :

`-size 100c`
: fichiers d'une taille de 100 octets exactement.

`-size +100k`
: fichiers de plus de 100 kilooctets.

`-size -20M`
: fichiers de moins de 20 mégaoctets.

`-size +2G`
: fichiers de plus de 2 gigaoctets.

> **NOTE** Pour localiser les fichiers vides, nous pouvons utiliser : `find . -size 0c` ou `find . -empty`.

Effectuer des opérations sur le résultat de la recherche

Une fois qu'une recherche est terminée, il est possible d'effectuer une action sur les fichiers résultants en utilisant l'option `-exec` :

```
$ find . -name "*.conf" -exec chmod 644 '{}' \;
```

Cette commande analyse tous les objets du répertoire courant (`.`) ainsi que tous les sous-répertoires à la recherche de fichiers dont le nom se termine par `.conf` et exécute ensuite la commande `chmod 644` pour modifier les permissions des fichiers résultants.

Pour l'instant, ne vous préoccupez pas de la signification de `'{}' \;` nous l'aborderons un peu plus loin.

Utiliser `grep` pour rechercher des fichiers en fonction de leur contenu

`grep` est utilisé pour rechercher l'occurrence d'un mot clé.

Prenons un cas de figure où nous devons rechercher des fichiers en fonction de leur contenu :

```
$ find . -type f -exec grep "lpi" '{}' \; -print
./.bash_history
Alpine/M
```

```
helping/M
```

Cette commande va rechercher tous les objets dans l'arborescence du répertoire courant (`.`) qui sont des fichiers (`-type f`) pour ensuite exécuter la commande `grep "lpi"` sur chaque fichier qui répond aux critères de recherche. Les fichiers qui remplissent ces conditions sont affichés à l'écran (`-print`). Les accolades (`{}`) symbolisent ici les résultats de la recherche `find`. Les `{}` sont entourés de guillemets simples (`'`) pour éviter de passer à `grep` des fichiers dont les noms contiennent des caractères spéciaux. La commande `-exec` se termine par un point-virgule (`;`) qui doit être échappé (`\;`) pour éviter toute interprétation par le *shell*.

L'ajout de l'option `-delete` à la fin d'une expression permet de supprimer tous les fichiers correspondants. Cette option doit être utilisée uniquement lorsque vous êtes sûr et certain que les résultats correspondent bien aux fichiers que vous souhaitez supprimer.

Dans l'exemple ci-dessous, `find` recherche tous les fichiers dans l'arborescence du répertoire courant puis supprime tous ceux dont le nom se termine par les caractères `.bak` :

```
$ find . -name "*.bak" -delete
```

Archiver des fichiers

La commande `tar` (archivage et compression)

La commande `tar`, une forme abrégée de "tape archive(r)", est utilisée pour créer des archives `tar` en compactant un groupe de fichiers en une archive. Une archive est créée dans le but de faciliter le déplacement ou la sauvegarde d'un groupe de fichiers. Imaginez `tar` comme un outil qui crée un adhésif sur lequel les fichiers peuvent s'attacher, se regrouper et se déplacer facilement.

`tar` permet en outre d'extraire une archive `tar`, d'afficher la liste des fichiers inclus dans l'archive et d'ajouter des fichiers supplémentaires à une archive existante.

Voici la syntaxe de la commande `tar` :

```
tar [OPERATION_AND_OPTIONS] [ARCHIVE_NAME] [FILE_NAME(S)]
```

`OPERATION`

Un seul argument opérationnel est autorisé et requis. Les opérations les plus fréquemment utilisées sont :

`--create` (`-c`)
Créer une nouvelle archive `tar`.

`--extract` (`-x`)
Extraire une archive dans son intégralité ou un ou plusieurs fichiers de cette archive.

`--list` (`-t`)
Afficher la liste des fichiers inclus dans l'archive.

`OPTIONS`
Voici les options les plus courantes :

`--verbose` (`-v`)
Afficher les fichiers en cours de traitement par la commande `tar`.

`--file=archive-name` (`-f archive-name`)
Spécifie le nom du fichier de l'archive.

`ARCHIVE_NAME`
Le nom de l'archive.

`FILE_NAME(S)`
La liste des noms de fichiers à extraire, séparés par des espaces. Si elle n'est pas fournie, l'archive est extraite dans son intégralité.

Créer une archive

Admettons que nous ayons un répertoire nommé `stuff` dans le répertoire courant et que nous voulions le sauvegarder dans un fichier nommé `archive.tar`. Nous devons invoquer la commande suivante :

```
$ tar -cvf archive.tar stuff
stuff/
stuff/service.conf
```

Voici ce que signifient concrètement les options utilisées :

`-c`
Créer une archive.

`-v`

Affiche la progression de la création de l'archive dans le terminal, en mode "bavard". Le `-v` est toujours facultatif dans ces commandes, mais il est tout de même pratique.

`-f`

Permet de spécifier le nom de fichier de l'archive.

En règle générale, pour archiver un seul répertoire ou un seul fichier sous Linux, on utilise :

```
tar -cvf NAME-OF-ARCHIVE.tar /PATH/TO/DIRECTORY-OR-FILE
```

NOTE `tar` fonctionne de manière récursive. Il va effectuer l'action requise sur chaque répertoire subséquent à l'intérieur du répertoire indiqué en argument.

Pour archiver plusieurs répertoires à la fois, il suffit de dresser la liste de tous les répertoires en les délimitant par un espace dans la section `/PATH/TO/DIRECTORY-OR-FILE` :

```
$ tar -cvf archive.tar stuff1 stuff2
```

Cette commande génère une archive de `stuff1` et `stuff2` dans `archive.tar`.

Extraire une archive

Nous pouvons extraire une archive à l'aide de `tar` :

```
$ tar -xvf archive.tar
stuff/
stuff/service.conf
```

Cette commande va extraire le contenu de `archive.tar` vers le répertoire courant.

La commande est la même que celle utilisée pour la création de l'archive ci-dessus, à l'exception de l'option `-x` qui remplace l'option `-c`.

Pour extraire le contenu de l'archive vers un répertoire particulier, on utilise l'option `-C` :

```
$ tar -xvf archive.tar -C /tmp
```

Cette commande va extraire le contenu de `archive.tar` dans le répertoire `/tmp`.

```
$ ls /tmp
stuff
```

La compression avec `tar`.

La commande GNU `tar` fournie avec les distributions Linux permet de créer une archive `.tar` et de la compresser ensuite avec la compression `gzip` ou `bzip2` en une seule et même commande :

```
$ tar -czvf name-of-archive.tar.gz stuff
```

Cette commande va créer un fichier compressé en utilisant l'algorithme de compression `gzip` (`-z`).

Même si la compression `gzip` est utilisée le plus souvent pour créer des fichiers `.tar.gz` ou `.tgz`, `tar` supporte également la compression `bzip2`. Cela permet de créer des fichiers compressés `bzip2`, souvent nommés `.tar.bz2`, `.tar.bz` ou `.tbz`.

Pour ce faire, nous remplaçons `-z` pour `gzip` par `-j` pour `bzip2` :

```
$ tar -cjvf name-of-archive.tar.bz stuff
```

Pour décompresser le fichier, il suffit de remplacer `-c` par `-x`, sachant que `x` signifie "extract" :

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

`gzip` est plus rapide, mais il compresse généralement un peu moins, vous obtenez donc un fichier un peu plus volumineux. `bzip2` est plus lent, mais il compresse un peu plus, de sorte que vous obtenez un fichier un peu plus compact. Mais en règle générale, `gzip` et `bzip2` font pratiquement la même chose et les deux vont fonctionner de manière similaire.

Alternativement, on peut appliquer une compression `gzip` ou `bzip2` en utilisant la commande `gzip` pour une compression `gzip` et la commande `bzip` pour une compression `bzip`. Par exemple, pour appliquer la compression `gzip`, invoquez :

```
gzip FILE-TO-COMPRESS
```

`gzip`
: crée le fichier compressé du même nom mais avec une extension `.gz`.

`gzip`
: supprime le fichier original après la création du fichier compressé.

La commande `bzip2` fonctionne de manière similaire.

Pour décompresser les fichiers, nous utilisons soit `gunzip` soit `bunzip2` en fonction de l'algorithme utilisé pour la compression.

La commande `cpio`

`cpio` signifie "copy in, copy out". Cette commande est utilisée pour traiter les archives telles que les fichiers `*.cpio` ou `*.tar`.

`cpio` effectue les opérations suivantes :

- Copier des fichiers vers une archive.
- Extraire les fichiers d'une archive.

La liste des fichiers est récupérée depuis l'entrée standard (principalement depuis la sortie de `ls`).

Pour créer une archive `cpio`, nous allons invoquer :

```
$ ls | cpio -o > archive.cpio
```

L'option `-o` indique à `cpio` de créer une sortie. Dans cet exemple, le fichier de sortie créé est `archive.cpio`. La commande `ls` liste le contenu du répertoire courant qui doit être archivé.

Pour extraire l'archive, nous utilisons :

```
$ cpio -id < archive.cpio
```

L'option `-i` est utilisée pour effectuer l'extraction. L'option `-d` va créer le répertoire de destination. Le caractère `<` correspond à l'entrée standard. Le fichier d'entrée à extraire est `archive.cpio`.

La commande `dd`

`dd` copie des données d'un emplacement vers un autre. La syntaxe de `dd` n'est pas la même que celle de la plupart des programmes Unix, étant donné que la commande utilise la syntaxe `option=value` pour ses options en ligne de commande plutôt que le format GNU standard

`-option value` ou `--option=value` :

```
$ dd if=oldfile of=newfile
```

Cette commande va copier le contenu de `oldfile` dans `newfile`, sachant que `if=` est le fichier d'entrée (*input file*) et `of=` fait référence au fichier de sortie (*output file*).

NOTE

En temps normal, la commande `dd` n'affiche rien à l'écran avant la fin de son exécution. L'option `status=progress` permet d'afficher la progression des opérations dans la console. Par exemple : `dd status=progress if=oldfile of=newfile`.

`dd` est également utilisé pour convertir les données en majuscules/minuscules ou pour écrire directement sur des périphériques bloc comme `/dev/sdb` :

```
$ dd if=oldfile of=newfile conv=ucase
```

Cette commande va copier tout le contenu de `oldfile` vers `newfile` en mettant tout le texte en majuscules.

La commande ci-dessous va sauvegarder l'ensemble du disque dur `/dev/sda` dans un fichier `backup.dd` :

```
$ dd if=/dev/sda of=backup.dd bs=4096
```

Exercices guidés

1. Considérez le *listing* suivant :

```
$ find /home/frank/Documents/ -type d
/home/frank/Documents/
/home/frank/Documents/animal
/home/frank/Documents/animal/domestic
/home/frank/Documents/animal/wild
```

   - Quel genre de fichiers cette commande afficherait-elle ?

   - Dans quel répertoire la recherche doit-elle débuter ?

2. Un utilisateur veut compresser son répertoire de sauvegarde. Il utilise la commande suivante :

```
$ tar cvf /home/frank/backup.tar.gz /home/frank/dir1
```

   Quelle est l'option qui manque pour compresser la sauvegarde en utilisant l'algorithme `gzip` ?

Exercices d'approfondissement

1. Un administrateur système doit effectuer des vérifications régulières dans le but de nettoyer les fichiers volumineux. Ces fichiers volumineux se trouvent dans `/var` et se terminent par l'extension `.backup`.
   - Écrivez la commande basée sur `find` pour localiser ces fichiers :
   - Une analyse de la taille de ces fichiers révèle qu'ils vont de `100M` à `1000M`. Complétez la commande ci-dessus avec cette nouvelle information afin de pouvoir localiser les fichiers de sauvegarde allant de `100M` à `1000M` :
   - Enfin, complétez la commande avec une action de suppression pour que ces fichiers soient effacés :
2. Le répertoire `/var` contient quatre fichiers de sauvegarde :

```
db-jan-2018.backup
db-feb-2018.backup
db-march-2018.backup
db-apr-2018.backup
```

   - En utilisant `tar`, spécifiez la commande qui créerait un fichier d'archive nommé `db-first-quarter-2018.backup.tar` :
   - En utilisant `tar`, spécifiez la commande qui va créer l'archive et la compresser en utilisant `gzip`. Notez que le nom du fichier résultant devra se terminer par `.gz` :

Résumé

Voici ce que nous avons appris dans cette leçon :

- Trouver des fichiers avec `find`.
- Ajouter des critères de recherche basés sur le temps, le type de fichier ou la taille en fournissant des arguments à `find`.
- Effectuer des opérations sur les fichiers résultants.
- Archiver, compresser et décompresser des fichiers en utilisant `tar`.
- Traiter des archives avec `cpio`.
- Copier des fichiers avec `dd`.

Réponses aux exercices guidés

1. Considérez le *listing* suivant :

```
$ find /home/frank/Documents/ -type d
/home/frank/Documents/
/home/frank/Documents/animal
/home/frank/Documents/animal/domestic
/home/frank/Documents/animal/wild
```

   - Quel genre de fichiers cette commande afficherait-elle ?

     Des répertoires

   - Dans quel répertoire la recherche doit-elle débuter ?

     `/home/frank/Documents`

2. Un utilisateur veut compresser son répertoire de sauvegarde. Il utilise la commande suivante :

```
$ tar cvf /home/frank/backup.tar.gz /home/frank/dir1
```

   Quelle est l'option qui manque pour compresser la sauvegarde en utilisant l'algorithme `gzip` ?

   L'option `-z`.

Réponses aux exercices d'approfondissement

1. Un administrateur système doit effectuer des vérifications régulières dans le but de nettoyer les fichiers volumineux. Ces fichiers volumineux se trouvent dans `/var` et se terminent par l'extension `.backup`.
   - Écrivez la commande basée sur `find` pour localiser ces fichiers :

     ```
     $ find /var -name *.backup
     ```

   - Une analyse de la taille de ces fichiers révèle qu'ils vont de `100M` à `1G`. Complétez la commande ci-dessus avec cette nouvelle information afin de pouvoir localiser les fichiers de sauvegarde allant de `100M` à `1G` :

     ```
     $ find /var -name *.backup -size +100M -size -1G
     ```

   - Enfin, complétez la commande avec une action de suppression pour que ces fichiers soient effacés :

     ```
     $ find /var -name *.backup -size +100M -size -1G -delete
     ```

2. Le répertoire `/var` contient quatre fichiers de sauvegarde :

   ```
   db-jan-2018.backup
   db-feb-2018.backup
   db-march-2018.backup
   db-apr-2018.backup
   ```

   - En utilisant `tar`, spécifiez la commande qui créerait un fichier d'archive nommé `db-first-quarter-2018.backup.tar` :

     ```
     $ tar -cvf db-first-quarter-2018.backup.tar db-jan-2018.backup db-feb-2018.backup db-march-2018.backup db-apr-2018.backup
     ```

   - En utilisant `tar`, spécifiez la commande qui va créer l'archive et la compresser en utilisant `gzip`. Notez que le nom du fichier résultant devra se terminer par `.gz` :

     ```
     $ tar -zcvf db-first-quarter-2018.backup.tar.gz db-jan-2018.backup db-feb-2018.backup
     ```

```
db-march-2018.backup db-apr-2018.backup
```

103.4 Utilisation des flux, des tubes et des redirections

Référence aux objectifs de LPI

LPIC-1 v5, Exam 101, Objective 103.4

Valeur

4

Domaines de connaissance les plus importants

- Redirection de l'entrée standard, de la sortie standard et de l'erreur standard.
- Connexion de la sortie d'une commande à l'entrée d'une autre commande.
- Utilisation de la sortie d'une commande comme paramètres d'une autre commande.
- Envoi simultané du résultat d'une commande vers la sortie standard et vers un fichier.

Liste partielle de termes, fichiers et utilitaires utilisés pour cet objectif

- `tee`
- `xargs`

103.4 Leçon 1

Certification :	LPIC-1 (101)
Version :	5.0
Thème :	103 Commandes GNU et Unix
Objectif :	103.4 Utiliser les flux, les tubes et les redirections
Leçon :	1 sur 2

Introduction

Tous les programmes informatiques reposent sur le même principe de base : les données fournies par une source quelconque sont transformées pour produire un résultat intelligible. Dans le contexte du *shell Linux*, la source de données peut être un fichier local, un fichier distant, un périphérique (comme un clavier), etc. La sortie du programme est généralement affichée sur un écran, mais il est tout aussi courant de stocker les données de sortie dans un système de fichiers local, de les envoyer vers un périphérique distant, de les diffuser via des haut-parleurs, etc.

Les systèmes d'exploitation de type Unix comme Linux offrent une grande variété de méthodes d'entrée/sortie. En l'occurrence, la méthode des *descripteurs de fichiers* permet d'associer dynamiquement des nombres entiers à des canaux de données, de sorte qu'un processus puisse les référencer comme ses flux de données d'entrée/sortie.

Les processus Linux standards disposent de trois canaux de communication ouverts par défaut : le canal *entrée standard* ou *standard input* (le plus souvent simplement appelé `stdin`), le canal *sortie standard* ou *standard output* (`stdout`) et le canal *sortie d'erreur standard* ou *standard error* (`stderr`). Les descripteurs de fichiers numériques respectifs assignés à ces canaux sont `0` à `stdin`,

`1` à `stdout` et `2` à `stderr`. Les canaux de communication sont également accessibles par le biais des périphériques spéciaux `/dev/stdin`, `/dev/stdout` et `/dev/stderr`.

Ces trois canaux de communication standards permettent aux programmeurs d'écrire du code qui lit et écrit des données sans se soucier du média dont elles proviennent ou auquel elles sont destinées. Par exemple, si un programme a besoin d'un jeu de données en entrée, il peut simplement demander des données à l'entrée standard et le dispositif qui fait office d'entrée standard fournira ces données. De même, la méthode la plus simple qu'un programme puisse utiliser pour afficher ses données de sortie est de les envoyer vers la sortie standard. Dans une session *shell* standard, le clavier est défini comme l'entrée standard (`stdin`) et l'écran du moniteur est défini comme la sortie standard (`stdout`) et la sortie d'erreur standard (`stderr`).

Le *shell* Bash permet de réaffecter les canaux de communication lors du chargement d'un programme. Il permet, par exemple, de ne pas utiliser l'écran comme sortie standard et de se servir d'un fichier du système de fichiers local comme `stdout`.

Les redirections

La réaffectation du descripteur de fichier d'un canal dans l'environnement *shell* est appelée une *redirection*. Une redirection est définie par un caractère spécial dans la ligne de commande. Par exemple, pour rediriger la sortie standard d'un processus vers un fichier, le symbole *plus grand que* `>` est placé à la fin de la commande et suivi du chemin d'accès vers le fichier qui recevra la sortie redirigée :

```
$ cat /proc/cpuinfo >/tmp/cpu.txt
```

Par défaut, seul le contenu arrivant sur la sortie standard (`stdout`) est redirigé. Cela est dû au fait que la valeur numérique du descripteur de fichier doit normalement être spécifiée juste avant le symbole *plus grand que* `>` et, lorsqu'elle n'est pas spécifiée, Bash redirige la sortie standard. Par conséquent, l'utilisation de `>` équivaut à `1>` (la valeur du descripteur de fichier de la sortie standard `stdout` est 1).

Pour capturer le contenu de la sortie d'erreur standard `stderr`, on utilisera plutôt la redirection `2>`. La plupart des programmes en ligne de commande envoient des informations de débogage et des messages d'erreur vers le canal d'erreur standard. Il est possible, par exemple, de capturer le message d'erreur déclenché par la tentative de lire un fichier inexistant :

```
$ cat /proc/cpu_info 2>/tmp/error.txt
$ cat /tmp/error.txt
```

```
cat: /proc/cpu_info: No such file or directory
```

La sortie standard `stdout` et la sortie d'erreur standard `stderr` sont toutes les deux redirigées vers la même cible avec `&>` ou `>&`. Évitez de placer des espaces à côté de l'esperluette `&`, faute de quoi Bash va la considérer comme une instruction pour exécuter le processus en arrière-plan au lieu d'effectuer la redirection.

La cible en question sera un chemin d'accès vers un fichier accessible en écriture, comme `/tmp/cpu.txt`, ou un descripteur de fichier accessible en écriture. Un descripteur de fichier cible est représenté par une esperluette `&` suivie de la valeur numérique du descripteur de fichier. Par exemple, `1>&2` redirige la sortie standard `stdout` vers la sortie d'erreur standard `stderr`. Pour faire l'inverse et rediriger la sortie d'erreur standard `stderr` vers la sortie standard `stdout`, il faudra plutôt utiliser `2>&1`.

Même si ce n'est pas très pratique, étant donné qu'il existe un moyen plus efficace d'effectuer la même tâche, il est possible de rediriger `stderr` vers `stdout` puis de rediriger le tout vers un fichier. Par exemple, une redirection pour écrire à la fois `stderr` et `stdout` dans un fichier nommé `log.txt` peut s'écrire sous la forme `>log.txt 2>&1`. Cependant, la principale motivation pour rediriger stderr vers stdout consiste à analyser les messages d'erreur et de débogage. On peut très bien rediriger la sortie standard d'un programme vers l'entrée standard d'un autre programme, mais il n'est pas possible de rediriger directement l'erreur standard vers l'entrée standard d'un autre programme. Ainsi, les messages d'un programme envoyés vers `stderr` doivent d'abord être redirigés vers `stdout` afin d'être lus par le `stdin` d'un autre programme.

Pour ignorer tout simplement la sortie d'une commande, son contenu peut être redirigé vers le fichier spécial `/dev/null`. Par exemple, `>log.txt 2>/dev/null` enregistre le contenu de `stdout` dans le fichier `log.txt` en rejetant les erreurs. Le fichier `/dev/null` est accessible en écriture à n'importe quel utilisateur, mais aucune donnée ne pourra être récupérée, étant donné qu'elles ne sont stockées nulle part.

Un message d'erreur s'affiche si la cible spécifiée n'est pas accessible en écriture (si le chemin pointe vers un répertoire ou un fichier en lecture seule) et aucune modification n'est apportée à la cible. En revanche, une redirection de sortie écrase une cible existante accessible en écriture sans la moindre confirmation. Les fichiers sont écrasés par les redirections de sortie à moins que l'option Bash `noclobber` soit activée, ce qui peut se faire pour la session en cours avec la commande `set -o noclobber` ou `set -C` :

```
$ set -o noclobber
$ cat /proc/cpu_info 2>/tmp/error.txt
-bash: /tmp/error.txt: cannot overwrite existing file
```

Pour désactiver l'option `noclobber` pour la session en cours, il suffit d'invoquer `set +o noclobber` ou `set +C`. Pour rendre l'option `noclobber` persistante, il faut l'inclure dans l'environnement Bash de l'utilisateur ou du système.

Même avec l'option `noclobber` activée, il est possible d'ajouter des données redirigées à un contenu existant. Cela peut se faire grâce à une redirection écrite avec un double chevron `>>` :

```
$ cat /proc/cpu_info 2>>/tmp/error.txt
$ cat /tmp/error.txt
cat: /proc/cpu_info: No such file or directory
cat: /proc/cpu_info: No such file or directory
```

Dans l'exemple précédent, le nouveau message d'erreur a été ajouté au message existant dans le fichier `/tmp/error.txt`. Si le fichier n'existe pas encore, il sera créé avec les nouvelles données.

La source de données de l'entrée standard d'un processus peut également être réaffectée. Le symbole inférieur à `<` est utilisé pour rediriger le contenu d'un fichier vers l'entrée standard `stdin` d'un processus. Dans ce cas, les données transitent de droite à gauche : le descripteur réaffecté est supposé être 0 à gauche du symbole inférieur à `<` et la source de données (un chemin vers un fichier) doit être à droite du symbole inférieur à `<`. La commande `uniq`, comme la plupart des outils de gestion de texte en ligne de commande, accepte par défaut les données envoyées à l'entrée standard `stdin` :

```
$ uniq -c </tmp/error.txt
      2 cat: /proc/cpu_info: No such file or directory
```

L'option `-c` demande à `uniq` d'afficher le nombre de fois qu'une ligne répétée apparaît dans le texte. Comme la valeur numérique du descripteur de fichier redirigé a été supprimée, la commande de l'exemple équivaut à `uniq -c 0</tmp/error.txt`. Utiliser un descripteur de fichier autre que `0` dans une redirection d'entrée n'a de sens que dans des contextes bien spécifiques, étant donné qu'il est possible pour un programme de demander des données aux descripteurs de fichier `3`, `4`, etc. En effet, les programmes peuvent utiliser n'importe quel nombre entier supérieur à 2 comme nouveau descripteur de fichier pour l'entrée/sortie de données. À titre d'exemple, le code C suivant lit les données depuis le descripteur de fichier `3` et les reproduit simplement vers le descripteur de fichier `4` :

NOTE Le programme doit gérer correctement ce genre de descripteurs de fichiers, faute de quoi il pourrait tenter une opération de lecture ou d'écriture invalide et entraîner un plantage.

```
#include <stdio.h>

int main(int argc, char **argv){
  FILE *fd_3, *fd_4;
  // Open file descriptor 3
  fd_3 = fdopen(3, "r");
  // Open file descriptor 4
  fd_4 = fdopen(4, "w");
  // Read from file descriptor 3
  char buf[32];
  while ( fgets(buf, 32, fd_3) != NULL ){
    // Write to file descriptor 4
    fprintf(fd_4, "%s", buf);
  }
  // Close both file descriptors
  fclose(fd_3);
  fclose(fd_4);
}
```

Pour tester ce bout de code, enregistrez-le sous le nom de `fd.c` et compilez-le avec `gcc -o fd fd.c`. Ce programme nécessite que les descripteurs de fichiers 3 et 4 soient disponibles pour qu'il puisse y accéder en lecture et en écriture. Pour donner un exemple, le fichier `/tmp/error.txt` créé plus haut pourra être utilisé comme source pour le descripteur de fichier `3` et le descripteur de fichier `4` pourra être redirigé vers la sortie standard `stdout` :

```
$ ./fd 3</tmp/error.txt 4>&1
cat: /proc/cpu_info: No such file or directory
cat: /proc/cpu_info: No such file or directory
```

Du point de vue du développeur, l'utilisation des descripteurs de fichiers évite d'avoir à gérer l'analyse des options et les chemins du système de fichiers. Un seul et même descripteur de fichier peut même être utilisé comme entrée et sortie. Dans ce cas, le descripteur de fichier est défini en ligne de commande avec les symboles inférieur à `<` et supérieur à `>`, comme dans `3<>/tmp/error.txt`.

Documents en ligne

Une autre manière de rediriger l'entrée fait appel aux méthodes *Here document* et *Here string*, également appelées *documents en ligne*. La redirection *Here document* permet la saisie d'un texte de plusieurs lignes qui sera utilisé comme contenu redirigé. Un double chevron constitué de deux

symboles "inférieur à" `<<` indique une redirection *Here document* :

```
$ wc -c <<EOF
> How many characters
> in this Here document?
> EOF
43
```

À droite du double chevron `<<` figure le terme de clôture `EOF` (*End Of File*). Le mode d'insertion se termine dès qu'une ligne contenant uniquement ce terme final est saisie. On peut très bien utiliser n'importe quel autre terme pour clôturer, à condition de ne pas insérer d'espace entre le double chevron `<<` et le terme final. Dans l'exemple ci-dessus, les deux lignes de texte ont été renvoyées vers l'entrée standard `stdin` de la commande `wc -c`, qui affiche le nombre de caractères. Tout comme pour la redirection d'entrée pour les fichiers, l'entrée standard `stdin` (descripteur de fichier `0`) est supposée si le descripteur de fichier redirigé n'est pas renseigné.

La méthode *Here string* ressemble beaucoup à la méthode *Here document*, mais pour une seule ligne :

```
$ wc -c <<<"How many characters in this Here string?"
41
```

Dans cet exemple, la chaîne de caractères à droite du triple chevron `<<<` est envoyée vers l'entrée standard `stdin` de `wc -c`, qui compte le nombre de caractères. Les chaînes de caractères contenant des espaces doivent être placées entre guillemets, faute de quoi seul le premier mot sera utilisé comme *Here string* et les autres seront considérés comme des arguments pour la commande.

Exercices guidés

1. En dehors des fichiers texte, la commande `cat` peut également fonctionner avec des données binaires, pour envoyer le contenu d'un périphérique de type bloc vers un fichier par exemple. En utilisant la redirection, comment `cat` peut-il envoyer le contenu du périphérique `/dev/sdc` vers le fichier `sdc.img` dans le répertoire courant ?

2. Quel est le nom du canal standard redirigé par la commande `date 1> now.txt` ?

3. Après avoir tenté d'écraser un fichier en utilisant la redirection, un utilisateur se retrouve confronté à un message d'erreur indiquant que l'option `noclobber` est activée. Comment peut-il désactiver l'option `noclobber` pour la session en cours ?

4. Quel sera le résultat de la commande `cat <<.>/dev/stdout` ?

Exercices d'approfondissement

1. La commande `cat /proc/cpu_info` affiche un message d'erreur parce que `/proc/cpu_info` n'existe pas. La commande `cat /proc/cpu_info 2>1` redirige le message d'erreur vers où ?

2. Est-ce qu'il sera toujours possible de rejeter du contenu renvoyé vers `/dev/null` si l'option `noclobber` est activée pour la session courante du shell ?

3. Sans utiliser `echo`, comment peut-on rediriger le contenu de la variable `$USER` vers l'entrée standard `stdin` de la commande `sha1sum` ?

4. Le noyau Linux conserve des liens symboliques dans `/proc/PID/fd/` vers chaque fichier ouvert par un processus, où *PID* est le numéro d'identification du processus correspondant. Comment l'administrateur système pourrait-il utiliser ce répertoire pour vérifier l'emplacement des fichiers logs ouverts par `nginx`, en supposant que son PID est `1234` ?

5. Il est possible d'effectuer des calculs arithmétiques en utilisant les seules commandes intégrées à l'interpréteur de commandes, mais les calculs en virgule flottante nécessitent des programmes spécifiques comme `bc` (*basic calculator*). `bc` permet même de spécifier le nombre de décimales, grâce au paramètre `scale`. En revanche, `bc` n'accepte les opérations que par l'entrée standard, généralement saisies en mode interactif. En utilisant un *Here string*, comment peut-on envoyer l'opération en virgule flottante `scale=6 ; 1/3` vers l'entrée standard de `bc` ?

Résumé

Cette leçon aborde les méthodes qui permettent d'exécuter un programme en redirigeant ses canaux de communication standard. Les processus Linux utilisent ces canaux standard comme *descripteurs de fichiers* génériques pour lire et écrire des données, ce qui permet de les convertir à volonté en fichiers ou en périphériques. La leçon comporte les étapes suivantes :

- Que sont les descripteurs de fichiers et quel est leur rôle sous Linux.
- Les canaux de communication standard de chaque processus : *stdin*, *stdout* et *stderr*.
- Comment exécuter correctement une commande en utilisant la redirection des données, en entrée comme en sortie.
- Comment utiliser les documents en ligne dans les redirections d'entrée.

Voici les procédures et les commandes abordées :

- Opérateurs de redirection : `>`, `<`, `>>`, `<<`, `<<<`.
- Commandes `cat`, `set`, `uniq` et `wc`.

Réponses aux exercices guidés

1. En dehors des fichiers texte, la commande `cat` peut également fonctionner avec des données binaires, pour envoyer le contenu d'un périphérique de type bloc vers un fichier par exemple. En utilisant la redirection, comment `cat` peut-il envoyer le contenu du périphérique `/dev/sdc` vers le fichier `sdc.img` dans le répertoire courant ?

   ```
   $ cat /dev/sdc > sdc.img
   ```

2. Quel est le nom du canal standard redirigé par la commande `date 1> now.txt` ?

   Sortie standard ou `stdout`

3. Après avoir tenté d'écraser un fichier en utilisant la redirection, un utilisateur se retrouve confronté à un message d'erreur indiquant que l'option `noclobber` est activée. Comment peut-il désactiver l'option `noclobber` pour la session en cours ?

   `set +C` ou `set +o noclobber`

4. Quel sera le résultat de la commande `cat <<.>/dev/stdout` ?

   Bash entrera en mode de saisie *Heredoc*, puis en sortira lorsqu'un point apparaîtra tout seul sur une ligne. Le texte saisi sera redirigé vers la sortie standard `stdout` (affiché à l'écran).

Réponses aux exercices d'approfondissement

1. La commande `cat /proc/cpu_info` affiche un message d'erreur parce que `/proc/cpu_info` n'existe pas. La commande `cat /proc/cpu_info 2>1` redirige le message d'erreur vers où ?

   Vers un fichier nommé `1` dans le répertoire courant.

2. Est-ce qu'il sera toujours possible de rejeter du contenu renvoyé vers `/dev/null` si l'option `noclobber` est activée pour la session courante du shell ?

   Oui. `/dev/null` est un fichier spécial non affecté par `noclobber`.

3. Sans utiliser `echo`, comment peut-on rediriger le contenu de la variable `$USER` vers l'entrée standard `stdin` de la commande `sha1sum` ?

   ```
   $ sha1sum <<<$USER
   ```

4. Le noyau Linux conserve des liens symboliques dans `/proc/PID/fd/` vers chaque fichier ouvert par un processus, où *PID* est le numéro d'identification du processus correspondant. Comment l'administrateur système pourrait-il utiliser ce répertoire pour vérifier l'emplacement des fichiers logs ouverts par `nginx`, en supposant que son PID est `1234` ?

   En exécutant la commande `ls -l /proc/1234/fd`, qui va afficher les cibles de chaque lien symbolique dans le répertoire.

5. Il est possible d'effectuer des calculs arithmétiques en utilisant les seules commandes intégrées à l'interpréteur de commandes, mais les calculs en virgule flottante nécessitent des programmes spécifiques comme `bc` (*basic calculator*). `bc` permet même de spécifier le nombre de décimales, grâce au paramètre `scale`. En revanche, `bc` n'accepte les opérations que par l'entrée standard, généralement saisies en mode interactif. En utilisant un *Here string*, comment peut-on envoyer l'opération en virgule flottante `scale=6 ; 1/3` vers l'entrée standard de `bc` ?

   ```
   $ bc <<<"scale=6; 1/3"
   ```

103.4 Leçon 2

Certification :	LPIC-1 (101)
Version :	5.0
Thème :	103 Commandes GNU et Unix
Objectif :	103.4 Utiliser les flux, les tubes et les redirections
Leçon :	2 sur 2

Introduction

Un des aspects de la philosophie Unix stipule que chaque programme doit avoir une fonction spécifique et ne pas essayer d'incorporer des fonctionnalités en dehors de son champ d'application. Le fait de garder les choses simples ne signifie pas pour autant que les résultats seront moins élaborés, étant donné que différents programmes peuvent s'enchaîner pour produire un résultat combiné. La barre verticale `|`, également connu sous le nom de symbole *pipe* (ou "tube"), peut être utilisé pour créer un pipeline qui relie la sortie d'un programme directement à l'entrée d'un autre programme, tandis que la substitution de commande permet de stocker la sortie d'un programme dans une variable ou de l'utiliser directement comme argument d'une autre commande.

Les tubes

Contrairement aux redirections, les tubes font circuler les données de gauche à droite dans la ligne de commande. La cible est un autre processus et non pas un chemin du système de fichiers, un descripteur de fichier ou un document en ligne (`Here document`). Le caractère *pipe* `|` indique au shell de lancer toutes les commandes distinctes en même temps et de connecter la sortie de la

commande précédente à l'entrée de la commande suivante, de gauche à droite. Par exemple, au lieu d'utiliser des redirections, le contenu du fichier `/proc/cpuinfo` envoyé vers la sortie standard par `cat` peut être redirigé vers l'entrée standard de `wc` avec la commande suivante :

```
$ cat /proc/cpuinfo | wc
   208    1184    6096
```

En l'absence d'un chemin vers un fichier, `wc` compte le nombre de lignes, de mots et de caractères qu'il reçoit sur son entrée standard, comme c'est le cas dans l'exemple. Plusieurs *pipes* peuvent être présents dans une commande combinée. L'exemple suivant utilise deux *pipes* :

```
$ cat /proc/cpuinfo | grep 'model name' | uniq
model name      : Intel(R) Xeon(R) CPU           X5355  @ 2.66GHz
```

Le contenu du fichier `/proc/cpuinfo` produit par `cat /proc/cpuinfo` a été envoyé vers la commande `grep 'model name'`, qui sélectionne alors uniquement les lignes contenant le terme `model name`. La machine qui exécute l'exemple a beaucoup de processeurs, il y a donc plusieurs lignes avec `model name`. Le dernier tube relie `grep 'model name'` à `uniq`, qui se charge de supprimer toute ligne égale à la précédente.

Les tubes peuvent être combinés avec les redirections dans la même ligne de commande. L'exemple précédent peut être réécrit plus simplement :

```
$ grep 'model name' </proc/cpuinfo | uniq
model name      : Intel(R) Xeon(R) CPU           X5355  @ 2.66GHz
```

La redirection d'entrée pour `grep` n'est pas strictement nécessaire puisque `grep` accepte un chemin de fichier comme argument, mais l'exemple montre comment on peut construire ce genre de commandes combinées.

Les tubes et les redirections fonctionnent de manière exclusive, c'est-à-dire qu'une source ne peut être mise en correspondance qu'avec une seule cible. Pourtant, il est possible de rediriger une sortie vers un fichier tout en l'affichant à l'écran avec le programme `tee`. Pour ce faire, le premier programme envoie sa sortie vers l'entrée standard de `tee` et un nom de fichier est fourni à ce dernier pour stocker les données :

```
$ grep 'model name' </proc/cpuinfo | uniq | tee cpu_model.txt
model name      : Intel(R) Xeon(R) CPU           X5355  @ 2.66GHz
$ cat cpu_model.txt
```

```
model name      : Intel(R) Xeon(R) CPU           X5355  @ 2.66GHz
```

Le résultat du dernier programme de la chaîne, généré par `uniq`, est affiché et stocké dans le fichier `cpu_model.txt`. Pour éviter d'écraser le contenu du fichier fourni et pour y ajouter des données, l'option `-a` doit être fournie à `tee`.

Seule la sortie standard d'un processus est capturée par un tube. Imaginons que vous devez suivre un processus de compilation prolongé à l'écran tout en enregistrant à la fois la sortie standard et l'erreur standard dans un fichier pour une inspection ultérieure. En supposant que votre répertoire courant ne possède pas de *Makefile*, la commande suivante va générer une erreur :

```
$ make | tee log.txt
make: *** No targets specified and no makefile found.  Stop.
```

Bien qu'affiché à l'écran, le message d'erreur généré par `make` n'a pas été capturé par `tee` et un fichier `log.txt` vide a été créé. Il faut donc effectuer une redirection avant qu'un tube ne puisse capturer l'erreur standard :

```
$ make 2>&1 | tee log.txt
make: *** No targets specified and no makefile found.  Stop.
$ cat log.txt
make: *** No targets specified and no makefile found.  Stop.
```

Dans cet exemple, l'erreur standard de `make` a été redirigée vers la sortie standard, de sorte que `tee` a pu la capturer avec un tube, l'afficher à l'écran et l'enregistrer dans le fichier `log.txt`. Dans des cas comme celui-ci, il peut être utile de sauvegarder les messages d'erreur pour une inspection ultérieure.

La substitution de commande

Une autre méthode pour capturer la sortie d'une commande, c'est la *substitution de commande*. En plaçant une commande entre des apostrophes inversées, Bash la remplace par sa sortie standard. L'exemple suivant montre comment utiliser la sortie standard d'un programme comme argument d'un autre programme :

```
$ mkdir `date +%Y-%m-%d`
$ ls
2019-09-05
```

La sortie du programme `date`, la date actuelle au format *année-mois-jour*, a été utilisée comme argument pour créer un répertoire avec `mkdir`. Un résultat identique est obtenu en utilisant `$()` au lieu des apostrophes inversées :

```
$ rmdir 2019-09-05
$ mkdir $(date +%Y-%m-%d)
$ ls
2019-09-05
```

La même méthode peut être utilisée pour stocker la sortie d'une commande sous forme de variable :

```
$ OS=`uname -o`
$ echo $OS
GNU/Linux
```

La commande `uname -o` affiche le nom générique du système d'exploitation utilisé, lequel est stocké dans la variable de session `OS`. L'affectation de la sortie d'une commande à une variable est très utile dans les scripts, car elle permet de stocker et d'évaluer les données de plusieurs manières différentes.

En fonction de la sortie générée par la commande remplacée, la substitution de commande intégrée peut ne pas être appropriée. Une méthode plus sophistiquée pour utiliser la sortie d'un programme comme argument d'un autre programme fait appel à un intermédiaire appelé `xargs`. Le programme `xargs` utilise le contenu qu'il reçoit via l'entrée standard pour exécuter une commande donnée avec le contenu comme argument. L'exemple suivant montre `xargs` en train d'exécuter le programme `identify` avec les arguments fournis par le programme `find` :

```
$ find /usr/share/icons -name 'debian*' | xargs identify -format "%f: %wx%h\n"
debian-swirl.svg: 48x48
debian-swirl.png: 22x22
debian-swirl.png: 32x32
debian-swirl.png: 256x256
debian-swirl.png: 48x48
debian-swirl.png: 16x16
debian-swirl.png: 24x24
debian-swirl.svg: 48x48
```

Le programme `identify` fait partie de *ImageMagick*, une suite d'outils en ligne de commande pour inspecter, convertir et retoucher la plupart des types de fichiers images. Dans l'exemple,

`xargs` a pris tous les chemins listés par `find` et les a mis comme arguments à `identify`, qui affiche alors les informations pour chaque fichier avec le format requis par l'option `-format`. Les fichiers trouvés par `find` dans l'exemple sont des images contenant le logo de la distribution dans un système de fichiers Debian. `-format` est un paramètre de `identify`, et non pas de `xargs`.

L'option `-n 1` demande à `xargs` d'exécuter la commande donnée avec un seul argument à la fois. Dans le cas de l'exemple, au lieu de passer tous les chemins trouvés par `find` comme une liste d'arguments à `identify`, l'utilisation de `xargs -n 1` exécutera la commande `identify` séparément pour chaque chemin. L'utilisation de `-n 2` exécutera `identify` avec deux chemins comme arguments, `-n 3` avec trois chemins comme arguments et ainsi de suite. De même, lorsque `xargs` traite des contenus multilignes — comme c'est le cas avec l'entrée fournie par `find` — l'option `-L` peut être utilisée pour limiter le nombre de lignes qui seront utilisées comme arguments par l'exécution de la commande.

NOTE L'utilisation de `xargs` avec l'option `-n 1` ou `-L 1` pour traiter la sortie générée par `find` peut s'avérer inutile. La commande `find` comporte l'option `-exec` pour exécuter une commande donnée pour chaque élément du résultat de la recherche.

Si les chemins contiennent des espaces, il est primordial de lancer `find` avec l'option `-print0`. Cette option indique à `find` d'utiliser un caractère vide entre chaque entrée de manière à ce que la liste puisse être correctement analysée par `xargs` (la sortie a été supprimée) :

```
$ find . -name '*avi' -print0 -o -name '*mp4' -print0 -o -name '*mkv' -print0 | xargs -0 du
| sort -n
```

L'option `-0` demande à `xargs` d'utiliser le caractère vide comme séparateur. Ainsi, les chemins d'accès aux fichiers fournis par `find` sont correctement analysés, même s'ils contiennent des espaces ou d'autres caractères spéciaux. L'exemple précédent montre l'utilisation de la commande `du` qui permet de connaître l'utilisation du disque pour chaque fichier trouvé, puis de trier les résultats par taille. La sortie a été supprimée pour plus de concision. Notez que pour chacun des critères de recherche, il est nécessaire de passer l'option `-print0` à `find`.

Par défaut, `xargs` place les arguments de la commande exécutée en dernier. Pour changer ce comportement, il faut utiliser l'option `-I` :

```
$ find . -mindepth 2 -name '*avi' -print0 -o -name '*mp4' -print0 -o -name '*mkv' -print0 |
xargs -0 -I PATH mv PATH ./
```

Dans le dernier exemple, chaque fichier trouvé par `find` est déplacé vers le répertoire courant. Comme le ou les chemins source doivent être communiqués à `mv` avant le chemin cible, un terme

de substitution est donné à l'option `-I` de `xargs` qui est ensuite placé de manière appropriée juste après `mv`. En utilisant le caractère vide comme séparateur, il n'est pas nécessaire de mettre le terme de substitution entre guillemets.

Exercices guidés

1. Il est pratique d'enregistrer la date d'exécution des actions effectuées par des scripts automatisés. La commande `date +%Y-%m-%d` affiche la date actuelle au format *année-mois-jour*. Comment peut-on stocker le résultat de cette commande dans une variable shell appelée `TODAY` en utilisant la substitution de commande ?

2. En utilisant la commande `echo`, comment peut-on envoyer le contenu de la variable `TODAY` vers l'entrée standard de la commande `sed s/-/./g` ?

3. Comment la sortie de la commande `date +%Y-%m-%d` peut-elle être utilisée comme chaîne de caractères en ligne (*Here string*) pour la commande `sed s/-/./g` ?

4. La commande `convert image.jpeg -resize 25% small/image.jpeg` crée une version réduite de `image.jpeg` et place l'image résultante dans un fichier de même nom dans le sous-répertoire `small`. En utilisant `xargs`, comment peut-on exécuter la même commande pour chaque image listée dans le fichier `filelist.txt` ?

Exercices d'approfondissement

1. Une routine de sauvegarde simple génère régulièrement une image de la partition `/dev/sda1` avec `dd < /dev/sda1 > sda1.img`. Pour effectuer des vérifications ultérieures de l'intégrité des données, la routine génère également une empreinte SHA1 du fichier avec `sha1sum < sda1.img > sda1.sha1`. En ajoutant des tubes et la commande `tee`, comment ces deux commandes pourraient-elles être combinées en une seule ?

2. La commande `tar` est utilisée pour archiver plusieurs fichiers au sein d'un seul fichier, tout en préservant la structure des répertoires. L'option `-T` permet de spécifier un fichier contenant les chemins à archiver. Par exemple, `find /etc -type f | tar -cJ -f /srv/backup/etc.tar.xz -T -` crée un fichier tar compressé `etc.tar.xz` à partir de la liste fournie par la commande `find` (l'option `-T -` indique l'entrée standard comme liste de chemins). Afin d'éviter d'éventuelles erreurs d'analyse dues à des chemins contenant des espaces, quelles options de commande doivent être présentes pour `find` et `tar` ?

3. Au lieu d'ouvrir une nouvelle session shell distante, la commande `ssh` permet tout simplement d'exécuter une commande spécifiée en argument : `ssh user@storage "remote command"`. Étant donné que `ssh` permet également de rediriger la sortie standard d'un programme local vers l'entrée standard du programme distant, comment la commande `cat` pourrait-elle faire passer un fichier local nommé `etc.tar.gz` vers `/srv/backup/etc.tar.gz` sur le système `user@storage` via `ssh` ?

Résumé

Cette leçon aborde les techniques traditionnelles de communication inter-processus utilisées par Linux. Le *pipelining de commande* crée un canal de communication unidirectionnel entre deux processus et la *substitution de commande* permet de stocker la sortie d'un processus dans une variable shell. La leçon passe en revue les points suivants :

- Comment les *tubes* peuvent être utilisés pour transmettre la sortie d'un processus à l'entrée d'un autre processus.
- L'utilité des commandes `tee` et `xargs`.
- Comment capturer la sortie d'un processus avec la *substitution de commande*, en la stockant dans une variable ou en l'utilisant directement comme paramètre d'une autre commande.

Voici les procédures et les commandes abordées :

- Le *pipelining* de commande avec `|`.
- La substitution de commande avec des apostrophes inversées et `$()`.
- Les commandes `tee`, `xargs` et `find`.

Réponses aux exercices guidés

1. Il est pratique d'enregistrer la date d'exécution des actions effectuées par des scripts automatisés. La commande `date +%Y-%m-%d` affiche la date actuelle au format *année-mois-jour*. Comment peut-on stocker le résultat de cette commande dans une variable shell appelée `TODAY` en utilisant la substitution de commande ?

   ```
   $ TODAY=`date +%Y-%m-%d`
   ```

   ou bien

   ```
   $ TODAY=$(date +%Y-%m-%d)
   ```

2. En utilisant la commande `echo`, comment peut-on envoyer le contenu de la variable `TODAY` vers l'entrée standard de la commande `sed s/-/./g` ?

   ```
   $ echo $TODAY | sed s/-/./g
   ```

3. Comment la sortie de la commande `date +%Y-%m-%d` peut-elle être utilisée comme chaîne de caractères en ligne (*Here string*) pour la commande `sed s/-/./g` ?

   ```
   $ sed s/-/./g <<< `date +%Y-%m-%d`
   ```

   ou bien

   ```
   $ sed s/-/./g <<< $(date +%Y-%m-%d)
   ```

4. La commande `convert image.jpeg -resize 25% small/image.jpeg` crée une version réduite de `image.jpeg` et place l'image résultante dans un fichier de même nom dans le sous-répertoire `small`. En utilisant `xargs`, comment peut-on exécuter la même commande pour chaque image listée dans le fichier `filelist.txt` ?

   ```
   $ xargs -I IMG convert IMG -resize 25% small/IMG < filelist.txt
   ```

   ou bien

```
$ cat filelist.txt | xargs -I IMG convert IMG -resize 25% small/IMG
```

Réponses aux exercices d'approfondissement

1. Une routine de sauvegarde simple génère régulièrement une image de la partition `/dev/sda1` avec `dd < /dev/sda1 > sda1.img`. Pour effectuer des vérifications ultérieures de l'intégrité des données, la routine génère également une empreinte SHA1 du fichier avec `sha1sum < sda1.img > sda1.sha1`. En ajoutant des tubes et la commande `tee`, comment ces deux commandes pourraient-elles être combinées en une seule ?

   ```
   # dd < /dev/sda1 | tee sda1.img | sha1sum > sda1.sha1
   ```

2. La commande `tar` est utilisée pour archiver plusieurs fichiers au sein d'un seul fichier, tout en préservant la structure des répertoires. L'option `-T` permet de spécifier un fichier contenant les chemins à archiver. Par exemple, `find /etc -type f | tar -cJ -f /srv/backup/etc.tar.xz -T -` crée un fichier tar compressé `etc.tar.xz` à partir de la liste fournie par la commande `find` (l'option `-T -` indique l'entrée standard comme liste de chemins). Afin d'éviter d'éventuelles erreurs d'analyse dues à des chemins contenant des espaces, quelles options de commande doivent être présentes pour `find` et `tar` ?

   Les options `-print0` et `--null`:

   ```
   $ find /etc -type f -print0 | tar -cJ -f /srv/backup/etc.tar.xz --null -T -
   ```

3. Au lieu d'ouvrir une nouvelle session shell distante, la commande `ssh` permet tout simplement d'exécuter une commande spécifiée en argument : `ssh user@storage "remote command"`. Étant donné que `ssh` permet également de rediriger la sortie standard d'un programme local vers l'entrée standard du programme distant, comment la commande `cat` pourrait-elle faire passer un fichier local nommé `etc.tar.gz` vers `/srv/backup/etc.tar.gz` sur le système `user@storage` via `ssh` ?

   ```
   $ cat etc.tar.gz | ssh user@storage "cat > /srv/backup/etc.tar.gz"
   ```

   ou bien

   ```
   $ ssh user@storage "cat > /srv/backup/etc.tar.gz" < etc.tar.gz
   ```

103.5 Création, contrôle et interruption des processus

Référence aux objectifs de LPI

LPIC-1 v5, Exam 101, Objective 103.5

Valeur

4

Domaines de connaissance les plus importants

- Exécution de tâches au premier plan et en arrière plan.
- Indiquer à un programme qu'il doit continuer à s'exécuter après la déconnexion.
- Contrôle des processus actifs.
- Sélection et tri des processus à afficher.
- Envoi de signaux aux processus.

Liste partielle de termes, fichiers et utilitaires utilisés pour cet objectif

- `&`
- `bg`
- `fg`
- `jobs`
- `kill`
- `nohup`
- `ps`
- `top`
- `free`

- `uptime`
- `pgrep`
- `pkill`
- `killall`
- `watch`
- `screen`
- `tmux`

103.5 Leçon 1

Certification :	LPIC-1
Version :	5.0
Thème :	103 Commandes GNU et Unix
Objectif :	103.5 Créer, surveiller et arrêter des processus
Leçon :	1 sur 2

Introduction

Chaque fois que nous invoquons une commande, un ou plusieurs processus sont lancés. Un administrateur système compétent doit non seulement créer des processus, mais aussi être capable d'en assurer le suivi et de leur envoyer différents types de signaux lorsque cela est nécessaire. Dans cette leçon, nous allons examiner le contrôle des tâches et la supervision des processus.

Gérer les tâches

Les *tâches* (*jobs*) sont des processus qui ont été lancés de manière interactive via un terminal, envoyés en arrière-plan et dont l'exécution n'est pas encore terminée. Vous pouvez obtenir des informations sur les tâches actives sur votre système Linux (ainsi que leur état) en exécutant `jobs` :

```
$ jobs
```

La commande `jobs` ci-dessus n'a rien affiché, ce qui signifie qu'il n'y a aucune tâche active pour le

moment. Créons notre première tâche en lançant une commande qui prend un certain temps pour s'exécuter (la commande `sleep` avec un paramètre de `60`) et — pendant l'exécution — appuyons sur `Ctrl` + `Z` :

```
$ sleep 60
^Z
[1]+  Stopped                 sleep 60
```

L'exécution de la commande a été arrêtée (ou — plus exactement — suspendue) et l'invite de commande est à nouveau disponible. Vous pouvez rechercher à nouveau des tâches et vous verrez maintenant celle qui est *suspendue* :

```
$ jobs
[1]+  Stopped                 sleep 60
```

Voici une explication du résultat :

`[1]`
: Ce numéro correspond à l'identifiant (ID) de la tâche et peut être utilisé — précédé d'un symbole pourcent (%) — pour contrôler l'état de la tâche avec les outils `fg`, `bg` et `kill` (comme on vous le montrera plus tard).

`+`
: Le signe plus indique la tâche actuelle, par défaut (c'est-à-dire la dernière à être suspendue ou envoyée en arrière-plan). La tâche précédente est identifiée par le signe moins (`-`). Les autres tâches antérieures ne sont pas marquées.

`Stopped`
: Description de l'état de la tâche.

`sleep 60`
: La commande ou la tâche en elle-même.

L'option `-l` permet d'afficher en plus l'ID du processus (PID) juste avant l'état :

```
$ jobs -l
[1]+  1114 Stopped                 sleep 60
```

Voici les autres options disponibles pour `jobs` :

`-n`

Affiche uniquement les processus qui ont changé d'état depuis la dernière notification. Les différents états possibles sont : `Running` (en cours d'exécution), `Stopped` (arrêté), `Terminated` (complété) ou `Done` (fini).

`-p`

Affiche les ID des processus.

`-r`

Affiche uniquement les tâches en cours d'exécution.

`-s`

Affiche uniquement les tâches arrêtées ou suspendues.

NOTE Gardez à l'esprit qu'une tâche dispose d'un *ID de tâche* et d'un *ID de processus* (PID).

Spécifier une tâche

La commande `jobs` ainsi que d'autres outils comme `fg`, `bg` et `kill` (que vous verrez dans la section suivante) ont besoin d'une spécification de tâche (ou `jobspec`) pour agir sur une tâche en particulier. Comme nous venons de le voir, cela peut être - et c'est normalement le cas - l'ID de la tâche précédé de `%`. Cependant, d'autres spécifications sont également possibles. Voyons cela :

`%n`

Tâche dont l'identifiant est `n` :

```
$ jobs %1
[1]+  Stopped                 sleep 60
```

`%str`

Tâche dont la commande commence par `str` :

```
$ jobs %sl
[1]+  Stopped                 sleep 60
```

`%?str`

Tâche dont la commande contient `str` :

```
$ jobs %?le
[1]+  Stopped                 sleep 60
```

`%+` ou `%%`

Tâche actuelle (celle qui a été lancée en dernier en arrière-plan ou suspendue depuis le premier plan) :

```
$ jobs %+
[1]+  Stopped                 sleep 60
```

`%-`

Tâche précédente (celle qui était `%+` avant la tâche actuelle, par défaut) :

```
$ jobs %-
[1]+  Stopped                 sleep 60
```

Dans notre cas, étant donné qu'il n'y a qu'une seule tâche, elle est à la fois la plus récente et la précédente.

État d'une tâche : Suspension, premier plan et arrière-plan

Lorsqu'une tâche est en arrière-plan ou qu'elle a été suspendue, nous pouvons lui faire subir l'une des trois actions suivantes :

1. L'amener au premier plan avec `fg` :

   ```
   $ fg %1
   sleep 60
   ```

   `fg` fait passer la tâche spécifiée au premier plan et en fait la tâche actuelle. Maintenant nous pouvons attendre qu'elle se termine, la stopper à nouveau avec `Ctrl` + `Z` ou la terminer avec `Ctrl` + `C`.

2. L'amener en arrière-plan avec `bg` :

   ```
   $ bg %1
   [1]+ sleep 60 &
   ```

Une fois qu'elle est en arrière plan, la tâche peut être ramenée au premier plan avec `fg` ou tuée (voir ci-dessous). Notez l'esperluette (&) qui signifie que la tâche a été envoyée en arrière-plan. En fait, vous pouvez également utiliser l'esperluette pour démarrer un processus directement en arrière-plan :

```
$ sleep 100 &
[2] 970
```

En plus de l'ID de la nouvelle tâche (`[2]`), nous disposons maintenant de son ID de processus (`970`). Les deux tâches s'exécutent désormais en arrière-plan :

```
$ jobs
[1]-  Running                 sleep 60 &
[2]+  Running                 sleep 100 &
```

Un peu plus tard, la première tâche termine son exécution :

```
$ jobs
[1]-  Done                    sleep 60
[2]+  Running                 sleep 100 &
```

3. La terminer par le biais d'un signal `SIGTERM` avec `kill` :

```
$ kill %2
```

Pour être sûr que la tâche a été terminée, relancez `jobs` :

```
$ jobs
[2]+  Terminated              sleep 100
```

NOTE Lorsqu'aucune tâche n'est spécifiée, `fg` et `bg` vont agir sur la tâche actuelle par défaut. En revanche, `kill` a toujours besoin d'une spécification de tâche.

Détâcher des tâches : `nohup`

_ Les tâches que nous avons vues dans les sections ci-dessus étaient toutes attachées à la session de l'utilisateur qui les avait lancées. Cela signifie que si la session est terminée, les tâches disparaissent. Cependant, il est possible de détacher les tâches des sessions et de les faire tourner

même après la fermeture de la session. Ceci est réalisé grâce à la commande `nohup` (*no hangup*, ne pas raccrocher). La syntaxe est la suivante :

```
nohup COMMAND &
```

N'oubliez pas que le "&" envoie le processus en arrière-plan et libère le terminal dans lequel vous travaillez.

Détachons la tâche en arrière-plan `ping localhost` de la session en cours :

```
$ nohup ping localhost &
[1] 1251
$ nohup: ignoring input and appending output to 'nohup.out'
^C
```

L'affichage nous montre l'ID de la tâche (`[1]`) et le PID (`1251`), suivis d'un message qui nous indique le fichier `nohup.out`. C'est le fichier par défaut où `stdout` et `stderr` seront enregistrés. Maintenant, nous pouvons appuyer sur `Ctrl` + `C` pour libérer l'invite de commande, fermer la session, en démarrer une autre et utiliser `tail -f` pour vérifier si la commande s'exécute et si la sortie est enregistrée dans le fichier par défaut :

```
$ exit
logout
$ tail -f /home/carol/nohup.out
64 bytes from localhost (::1): icmp_seq=3 ttl=64 time=0.070 ms
64 bytes from localhost (::1): icmp_seq=4 ttl=64 time=0.068 ms
64 bytes from localhost (::1): icmp_seq=5 ttl=64 time=0.070 ms
^C
```

TIP Au lieu d'utiliser le fichier par défaut `nohup.out`, vous auriez pu spécifier un fichier de sortie de votre choix avec `nohup ping localhost > /chemin/vers/votre/fichier &`.

Si nous voulons terminer le processus, nous devons spécifier son PID :

```
$ kill 1251
```

Surveiller les processus

Un processus ou une tâche est une instance d'un programme en cours d'exécution. Ainsi, on crée de nouveaux processus chaque fois que l'on tape des commandes dans le terminal.

La commande `watch` exécute un programme à intervalles réguliers (2 secondes par défaut) et nous permet de garder à l'œil l'évolution de la sortie du programme dans le temps. Par exemple, nous pouvons surveiller les variations de la charge moyenne au fur et à mesure que des processus sont exécutés en tapant `watch uptime` :

```
Every  2.0s: uptime          debian: Tue Aug 20 23:31:27 2019

 23:31:27 up 21 min,  1 user,  load average: 0.00, 0.00, 0.00
```

La commande s'exécute jusqu'à ce qu'elle soit interrompue, nous devons donc l'arrêter avec `Ctrl` + `C`. Nous obtenons deux lignes en sortie : la première correspond à `watch` et nous indique la fréquence d'exécution de la commande (`Every 2.0s : uptime`), la commande ou le programme à surveiller (`uptime`) ainsi que le nom d'hôte et la date (`debian : Tue Aug 20 23:31:27 2019`). La deuxième ligne de sortie est celle de `uptime` et comprend l'heure (`23:31:27`), la durée de fonctionnement du système (`up 21 min`), le nombre d'utilisateurs actifs (`1 user`) et la charge moyenne du système ou le nombre de processus en exécution ou en état d'attente pour les 1, 5 et 15 dernières minutes (`load average : 0.00, 0.00, 0.00`).

De même, vous pouvez vérifier l'utilisation de la mémoire lors de la création de nouveaux processus avec `watch free` :

```
Every  2.0s: free            debian: Tue Aug 20 23:43:37 2019

 23:43:37 up 24 min,  1 user,  load average: 0.00, 0.00, 0.00
              total        used        free      shared  buff/cache   available
Mem:       16274868      493984    14729396       35064     1051488    15462040
Swap:      16777212           0    16777212
```

Pour modifier l'intervalle de mise à jour de `watch`, utilisez les options `-n` ou `--interval` avec le nombre de secondes en argument :

```
$ watch -n 5 free
```

A présent, la commande `free` s'exécute toutes les 5 secondes.

Pour plus d'informations sur les options de `uptime`, `free` et `watch`, reportez-vous à leurs pages de manuel en ligne.

NOTE Les informations fournies par `uptime` et `free` sont également intégrées dans des outils plus élaborés comme `top` et `ps` (voir ci-dessous).

Envoyer des signaux aux processus : `kill`

Chaque processus dispose d'un identifiant de processus unique ou PID. Une façon de trouver le PID d'un processus consiste à utiliser la commande `pgrep` suivie du nom du processus :

```
$ pgrep sleep
1201
```

NOTE L'identifiant d'un processus peut également être obtenu par la commande `pidof` (par exemple `pidof sleep`).

Tout comme la commande `pgrep`, la commande `pkill` termine un processus en se basant sur son nom :

```
$ pkill sleep
[1]+  Terminated              sleep 60
```

La commande `killall` permet de terminer plusieurs instances d'un même processus :

```
$ sleep 60 &
[1] 1246
$ sleep 70 &
[2] 1247
$ killall sleep
[1]-  Terminated              sleep 60
[2]+  Terminated              sleep 70
```

`pkill` et `killall` fonctionnent de la même manière que `kill` en envoyant un signal au(x) processus spécifié(s). En l'absence de signal, c'est le signal par défaut `SIGTERM` qui est envoyé. Par contre, `kill` ne prend comme argument qu'un identifiant de tâche ou de processus.

Les signaux peuvent être spécifiés par :

- Le nom :

```
$ kill -SIGHUP 1247
```

- Le nombre :

```
$ kill -1 1247
```

- L'option :

```
$ kill -s SIGHUP 1247
```

Pour faire fonctionner `kill` de la même manière que `pkill` ou `killall` (et nous épargner la recherche préalable des PIDs), nous pouvons utiliser la substitution de commande :

```
$ kill -1 $(pgrep sleep)
```

Comme vous le savez déjà, une syntaxe alternative est `kill -1 $(pgrep sleep)`.

TIP Pour une liste exhaustive de tous les signaux `kill` et de leurs codes, tapez `kill -l` dans le terminal. Utilisez `-SIGKILL` (`-9` ou `-s SIGKILL`) pour tuer les processus récalcitrants en cas d'échec de tous les autres signaux.

`top` et `ps`

Lorsqu'il s'agit de surveiller un processus, `top` et `ps` sont deux outils indispensables. Le premier produit des résultats dynamiques, tandis que le second le fait de manière statique. Quoi qu'il en soit, ce sont là deux excellents outils pour avoir une vue d'ensemble sur tous les processus du système.

Interagir avec `top`

Pour lancer `top`, tapez simplement `top` :

```
$ top

top - 11:10:29 up  2:21,  1 user,  load average: 0,11, 0,20, 0,14
Tasks:  73 total,   1 running,  72 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0,0 us,  0,3 sy,  0,0 ni, 99,7 id,  0,0 wa,  0,0 hi,  0,0 si,  0,0 st
KiB Mem :  1020332 total,   909492 free,    38796 used,    72044 buff/cache
KiB Swap:  1046524 total,  1046524 free,        0 used.   873264 avail Mem
```

```
  PID USER      PR  NI    VIRT    RES    SHR S %CPU %MEM     TIME+ COMMAND
  436 carol     20   0   42696   3624   3060 R  0,7  0,4   0:00.30 top
    4 root      20   0       0      0      0 S  0,3  0,0   0:00.12 kworker/0:0
  399 root      20   0   95204   6748   5780 S  0,3  0,7   0:00.22 sshd
    1 root      20   0   56872   6596   5208 S  0,0  0,6   0:01.29 systemd
    2 root      20   0       0      0      0 S  0,0  0,0   0:00.00 kthreadd
    3 root      20   0       0      0      0 S  0,0  0,0   0:00.02 ksoftirqd/0
    5 root       0 -20       0      0      0 S  0,0  0,0   0:00.00 kworker/0:0H
    6 root      20   0       0      0      0 S  0,0  0,0   0:00.00 kworker/u2:0
    7 root      20   0       0      0      0 S  0,0  0,0   0:00.08 rcu_sched
    8 root      20   0       0      0      0 S  0,0  0,0   0:00.00 rcu_bh
    9 root      rt   0       0      0      0 S  0,0  0,0   0:00.00 migration/0
   10 root       0 -20       0      0      0 S  0,0  0,0   0:00.00 lru-add-drain
   (...)
```

`top` permet une certaine interaction à l'utilisateur. La sortie par défaut est classée par ordre décroissant du pourcentage de temps CPU utilisé par chaque processus. Ce comportement peut être modifié en appuyant sur les touches suivantes à partir de `top` :

`M`

Classer par utilisation de *mémoire*.

`N`

Classer par *numéro* de PID.

`T`

Classer par *temps d'exécution*.

`P`

Classer par *pourcentage* d'utilisation CPU.

TIP Pour basculer entre le classement décroissant et croissant, il suffit d'appuyer sur `R`.

Voici d'autres raccourcis intéressants pour interagir avec `top` :

`?` ou `h`

Aide.

`k`

Envoyer un signal à un processus. `top` vous demandera le `PID` du processus à tuer ainsi que le signal à envoyer (`SIGTERM` ou `15` par défaut).

`r`

Modifier la priorité d'un processus (`renice`). `top` vous demandera la valeur de `nice`. Les valeurs possibles vont de -20 à 19, mais seul le super-utilisateur (`root`) peut définir une valeur négative ou inférieure à la valeur actuelle.

`u`

Afficher les processus d'un utilisateur en particulier (par défaut, les processus de tous les utilisateurs sont affichés).

`c`

Affiche les chemins absolus des programmes et distingue les processus en espace utilisateur de ceux du noyau (entre crochets).

`V`

Affichage en arborescence hiérarchique des processus.

`t` et `m`

Modifie l'aspect des relevés respectifs du CPU et de la mémoire selon un cycle en quatre étapes : les deux premières pressions affichent des barres de progression, la troisième masque ces barres et la quatrième les fait réapparaître.

`W`

Sauvegarder les paramètres de configuration dans `~/.toprc`.

TIP

Une version plus sophistiquée et plus conviviale de `top` est `htop`. Une autre alternative, peut-être plus exhaustive, est `atop`. Si ces outils ne sont pas déjà installés sur votre système, utilisez votre gestionnaire de paquets pour les installer et essayez-les.

Petite explication de l'affichage de `top`.

Les informations fournies par `top` sont réparties en deux zones : la *zone de synthèse* et la *zone des tâches*.

La zone de synthèse de `top`

La zone de synthèse comprend les cinq lignes du haut et nous fournit les informations suivantes :

- `top - 11:10:29 up 2:21, 1 user, load average: 0,11, 0,20, 0,14`
  - l'heure actuelle (au format 24 heures) : `11:20:29`

  - uptime (durée de fonctionnement du système) : `up 2:21`
  - nombre d'utilisateurs connectés et charge moyenne du CPU pour les 1, 5 et 15 dernières minutes, respectivement : `load average: 0,11, 0,20, 0,14`
- `Tasks: 73 total, 1 running, 72 sleeping, 0 stopped, 0 zombie` (informations sur les processus)
  - nombre total de processus en mode actif : `73 total`
  - en cours d'exécution : `1 running`
  - en veille (en attente de reprendre l'exécution) : `72 sleeping`
  - arrêtés (par un signal de contrôle de tâche) : `0 stopped`
  - zombie (ceux qui ont terminé leur exécution mais qui attendent que leur processus parent les supprime de la table des processus) :
- `%Cpu(s): 0,0 us, 0,3 sy, 0,0 ni, 99,7 id, 0,0 wa, 0,0 hi, 0,0 si, 0,0 st` (pourcentage du temps CPU passé sur…)
  - les processus utilisateurs : `0,0 us`
  - les processus du système/noyau : `0,4 sy`
  - les processus définis à une valeur *nice* — plus la valeur de `nice` est élevée, plus la priorité est faible : `0,0 ni`
  - rien — temps d'inactivité du CPU : `99,7 id`
  - les processus en attente d'opérations d'E/S : `0,0 wa`
  - les processus qui répondent aux interruptions matérielles - périphériques qui envoient au processeur des signaux qui requièrent son attention : `0,0 hi`
  - les processus qui répondent aux interruptions logicielles : `0,0 si`
  - les processus qui répondent aux tâches d'autres machines virtuelles dans un environnement virtuel, d'où une perte de temps : `0,0 st`
- `KiB Mem : 1020332 total, 909492 free, 38796 used, 72044 buff/cache` (informations sur la mémoire en kilo-octets)
  - la quantité totale de mémoire : `1020332 total`
  - la mémoire non utilisée : `909492 free`
  - la mémoire utilisée : `38796 used`
  - la mémoire mise en tampon et dans le cache pour éviter les accès excessifs au disque : `72044 buff/cache`

Notez comment le `total` est la somme des trois autres valeurs — `free`, `used` et `buff/cache` — (environ 1 Go dans notre cas).

- `KiB Swap: 1046524 total, 1046524 free, 0 used. 873264 avail Mem` (informations sur le swap en kilo-octets)
  - la quantité totale d'espace swap : `1046524 total`
  - l'espace swap non utilisé : `1046524 free`
  - l'espace swap utilisé : `0 used`
  - la quantité de mémoire swap qui peut être allouée aux processus sans provoquer davantage de swapping : `873264 avail Mem`

La zone des tâches dans `top` : les champs et les colonnes

En dessous de la *zone de synthèse* se trouve la *zone des tâches*, où l'on trouve une série de *champs* et de *colonnes* qui fournissent des informations sur les processus en cours d'exécution :

`PID`
: Identifiant du processus.

`USER`
: Utilisateur ayant invoqué la commande qui a généré le processus.

`PR`
: Priorité du processus pour le noyau.

`NI`
: Valeur `nice` du processus. Les valeurs inférieures ont une priorité plus élevée que les valeurs supérieures.

`VIRT`
: Quantité totale de mémoire utilisée par le processus (swap compris).

`RES`
: Mémoire RAM utilisée par le processus.

`SHR`
: Mémoire partagée par le processus avec d'autres processus.

S

État du processus. Les valeurs incluent : `S` (*sleep interruptible* — attente de la fin d'un événement), `R` (*runnable* — soit en cours d'exécution, soit dans la file d'attente pour être exécuté) ou `Z` (*zombie* — processus enfants terminés dont les structures de données n'ont pas encore été retirées de la table des processus).

%CPU

Pourcentage de CPU utilisé par le processus.

%MEM

Pourcentage de RAM utilisée par le processus, c'est-à-dire la valeur `RES` exprimée en pourcents.

TIME+

Durée totale de l'activité du processus.

COMMAND

Nom de la commande/du programme qui a généré le processus.

Afficher les processus en mode statique : `ps`

Comme nous l'avons dit plus haut, `ps` affiche un instantané des processus. Pour voir tous les processus dans un terminal (tty), tapez `ps a` :

```
$ ps a
  PID TTY      STAT   TIME COMMAND
  386 tty1     Ss+    0:00 /sbin/agetty --noclear tty1 linux
  424 tty7     Ssl+   0:00 /usr/lib/xorg/Xorg :0 -seat seat0 (...)
  655 pts/0    Ss     0:00 -bash
 1186 pts/0    R+     0:00 ps a
 (...)
```

Explication de la syntaxe des options et de l'affichage de `ps`

En ce qui concerne les options, `ps` accepte trois styles différents : BSD, UNIX et GNU. Voyons le fonctionnement de chacun de ces styles lors de la présentation d'informations sur un ID de processus donné :

BSD

Les options ne suivent pas un tiret initial :

```
$ ps p 811
  PID TTY      STAT   TIME COMMAND
  811 pts/0    S      0:00 -su
```

UNIX

Les options suivent un tiret initial :

```
$ ps -p 811
  PID TTY          TIME CMD
  811 pts/0    00:00:00 bash
```

GNU

Les options suivent un double tiret initial :

```
$ ps --pid 811
  PID TTY          TIME CMD
  811 pts/0    00:00:00 bash
```

Dans les trois cas, `ps` renvoie des informations sur le processus dont le `PID` est `811` — en l'occurrence `bash`.

De même, on pourra utiliser `ps` pour rechercher les processus lancés par un utilisateur donné :

- `ps U carol` (BSD)
- `ps -u carol` (UNIX)
- `ps --user carol` (GNU)

Vérifions les processus lancés par `carol` :

```
$ ps U carol
  PID TTY      STAT   TIME COMMAND
  811 pts/0    S      0:00 -su
  898 pts/0    R+     0:00 ps U carol
```

Elle a lancé deux processus : `bash` (`-su`) et `ps` (`ps U carol`). La colonne `STAT` nous indique l'état du processus (voir ci-dessous).

Nous pouvons tirer le meilleur parti de `ps` en combinant certaines de ses options. Une commande

très utile (avec une sortie similaire à celle de `top`) est `ps aux` (style BSD). Dans ce cas, les processus de tous les shells (et pas seulement le shell courant) sont affichés. Voici la signification des options :

`a`

Affiche les processus attachés à un `tty` ou terminal.

`u`

Afficher le format orienté utilisateur.

`x`

Affiche les processus qui ne sont pas attachés à un `tty` ou terminal.

```
$ ps aux
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.1 204504  6780 ?        Ss   14:04   0:00 /sbin/init
root         2  0.0  0.0      0     0 ?        S    14:04   0:00 [kthreadd]
root         3  0.0  0.0      0     0 ?        S    14:04   0:00 [ksoftirqd/0]
root         5  0.0  0.0      0     0 ?        S<   14:04   0:00 [kworker/0:0H]
root         7  0.0  0.0      0     0 ?        S    14:04   0:00 [rcu_sched]
root         8  0.0  0.0      0     0 ?        S    14:04   0:00 [rcu_bh]
root         9  0.0  0.0      0     0 ?        S    14:04   0:00 [migration/0]
(...)
```

Voici une explication des colonnes :

`USER`

Propriétaire du processus.

`PID`

Identifiant du processus.

`%CPU`

Pourcentage d'utilisation du CPU.

`%MEM`

Pourcentage d'utilisation de la mémoire physique.

`VSZ`

Mémoire virtuelle du processus en Kio.

`RSS`

Mémoire physique hors swap utilisée par le processus en Kio.

`TT`

Terminal (`tty`) qui contrôle le processus.

`STAT`

Code représentant l'état du processus. En dehors de `S`, `R` et `Z` (que nous avons vus en décrivant la sortie de `top`), d'autres valeurs possibles incluent : `D` (*uninterruptible sleep* — généralement en attente d'E/S), `T` (*stopped* — normalement par un signal de contrôle). Quelques modificateurs supplémentaires incluent : `<` (haute priorité par rapport aux autres processus), `N` (basse priorité par rapport aux autres processus), ou `+` (dans le groupe de processus de premier plan).

`STARTED`

Heure à laquelle le processus a démarré.

`TIME`

Temps CPU accumulé.

`COMMAND`

Commande qui a démarré le processus.

Exercices guidés

1. `oneko` est un programme amusant qui affiche un chat en train de poursuivre le curseur de votre souris. S'il n'est pas déjà installé sur votre PC, installez-le en utilisant le gestionnaire de paquets de votre distribution. Nous allons l'utiliser pour étudier le contrôle des tâches.
   - Lancez le programme. Comment faites-vous cela ?
   - Déplacez le curseur de la souris pour voir comment le chat le prend en chasse. Maintenant, suspendez le processus. Comment faites-vous cela ? Quel est le résultat ?
   - Vérifiez le nombre de tâches que vous avez actuellement. Qu'est-ce que vous devez taper ? Quel est le résultat ?
   - Maintenant, envoyez-le en arrière-plan en spécifiant son ID de tâche. Quel est le résultat ? Comment pouvez-vous dire que la tâche est exécutée en arrière-plan ?
   - Enfin, terminez la tâche en spécifiant son ID. Qu'est-ce que vous devez taper ?
2. Découvrez les PIDs de tous les processus engendrés par le *serveur web Apache HTTPD* (`apache2`) à l'aide de deux commandes distinctes :
3. Terminez tous les processus `apache2` sans utiliser leurs PIDs et en utilisant deux commandes distinctes :
4. Supposons que vous devez mettre fin à toutes les instances de `apache2` et que vous n'avez pas le temps de trouver leurs PIDs. Comment pouvez-vous faire cela en utilisant `kill` avec le signal `SIGTERM` par défaut en une seule ligne :

5. Démarrez `top` et interagissez avec lui en effectuant ce qui suit :
   - Afficher une vue arborescente des processus :
   - Afficher les chemins complets des processus en différenciant l'espace utilisateur et l'espace noyau :
6. Tapez la commande `ps` pour afficher tous les processus démarrés par l'utilisateur du *serveur web Apache HTTPD* (`www-data`) :
   - En utilisant la syntaxte BSD :
   - En utilisant la syntaxe UNIX :
   - En utilisant la syntaxe GNU :

Exercices d'approfondissement

1. Le signal `SIGHUP` peut être utilisé comme un moyen de redémarrer certains démons. Avec le *serveur web Apache HTTPD* — par exemple — l'envoi de `SIGHUP` au processus parent (celui démarré par `init`) tue ses enfants. Le processus parent, cependant, relit ses fichiers de configuration, rouvre les fichiers journaux et crée un nouvel ensemble de processus enfants. Effectuez les tâches suivantes :
   - Démarrez le serveur web :
   - Assurez-vous de connaître le PID du processus parent :
   - Faites redémarrer le serveur web Apache HTTPD en envoyant le signal `SIGHUP` à son processus parent :
   - Vérifiez que le processus parent n'a pas été tué et que de nouveaux processus enfants ont été créés :
2. Bien que statique à la base, la sortie de `ps` peut être rendue dynamique en combinant `ps` et `watch`. Nous allons surveiller le *serveur web Apache HTTPD* pour détecter les nouvelles connexions. Avant d'effectuer les tâches décrites ci-dessous, il est recommandé de lire la description de la directive `MaxConnectionsPerChild` dans Apache MPM Common Directives.
   - Ajoutez la directive `MaxConnectionsPerChild` avec une valeur de `1` dans le fichier de configuration de `apache2` — dans la famille *Debian* et dérivées on le trouve dans `/etc/apache2/apache2.conf` ; dans la famille *CentOS*, c'est dans `/etc/httpd/conf/httpd.conf`. N'oubliez pas de redémarrer `apache2` pour que les changements soient pris en compte.
   - Tapez une commande qui utilise `watch`, `ps` et `grep` pour les connexions `apache2`.
   - Maintenant, ouvrez un navigateur web ou utilisez un navigateur en ligne de commande comme `lynx` pour établir une connexion au serveur web via son adresse IP. Que voyez-vous

dans l'affichage de `watch` ?

3. Comme vous l'avez vu, dans sa configuration par défaut, `top` trie les tâches par pourcentage d'utilisation du CPU par ordre décroissant (les valeurs les plus élevées en haut). Ce comportement peut être modifié avec les raccourcis interactifs `M` (utilisation de la mémoire), `N` (identifiant unique du processus), `T` (temps d'exécution) et `P` (pourcentage du temps CPU). Cependant, vous pouvez également trier la liste des tâches à votre convenance en lançant `top` avec l'option `-o` (pour plus d'informations, consultez la page `man` de top). Maintenant, effectuez les tâches suivantes :
   - Lancez `top` de façon à ce que les tâches soient triées par utilisation de la mémoire :
   - Vérifiez si vous avez tapé la bonne commande en mettant la colonne mémoire en surbrillance :
4. `ps` dispose également d'une option `o` pour spécifier les colonnes que vous souhaitez afficher. Examinez cette alternative et effectuez les tâches suivantes :
   - Lancez `ps` de façon à ce que seules les informations sur *l'utilisateur*, *le pourcentage de mémoire utilisée*, *le pourcentage de temps CPU utilisé* et *la commande complète* soient affichées :
   - Maintenant, lancez `ps` pour que les seules informations affichées soient celles de l'utilisateur et le nom des programmes qu'il utilise :

Résumé

Dans cette leçon, vous avez découvert les tâches et le contrôle des tâches. Voici les notions et les concepts importants que vous devez retenir :

- Les tâches sont des processus qui sont envoyés en arrière-plan.
- En dehors de l'*ID du processus*, un *ID de tâche* est affecté aux tâches lors de leur création.
- Pour contrôler les tâches, il faut une spécification de tâche (`jobspec`).
- Les tâches peuvent être mises au premier plan, envoyées à l'arrière-plan, suspendues et terminées (ou *tuées*).
- Une tâche peut être détachée du terminal et de la session dans laquelle elle a été créée.

De même, nous avons également abordé le concept de *processus* et de *surveillance des processus*. Voici les idées principales :

- Les processus sont des programmes en cours d'exécution.
- Les processus peuvent être surveillés.
- Différents outils nous permettent de connaître l'ID des processus et de leur envoyer des signaux pour les arrêter.
- Les signaux peuvent être spécifiés par un nom (comme `-SIGTERM`), un nombre (comme `-15`) ou une option (comme `-s SIGTERM`).
- `top` et `ps` sont très efficaces lorsqu'il s'agit de surveiller des processus. La sortie du premier est dynamique et se met constamment à jour ; de son côté, `ps` affiche les résultats de manière statique.

Les commandes suivantes ont été abordées dans cette leçon :

`jobs`
: Affiche les tâches actives et leur état.

`sleep`
: Retarder pour une durée déterminée.

`fg`
: Amener la tâche au premier plan.

`bg`
: Déplacer la tâche vers l'arrière-plan.

`kill`

Envoyer un signal à la tâche.

`nohup`

Détacher la tâche de la session / du terminal.

`exit`

Quitter le shell en cours.

`tail`

Afficher les dernières lignes d'un fichier.

`watch`

Exécuter une commande de manière répétée (cycle de 2 secondes par défaut).

`uptime`

Afficher la durée de fonctionnement du système, le nombre d'utilisateurs actifs et la charge moyenne du système.

`free`

Afficher l'utilisation de la mémoire.

`pgrep`

Trouver l'ID d'un processus en fonction du nom.

`pidof`

Trouver l'ID d'un processus en fonction du nom.

`pkill`

Envoyer un signal à un processus par son nom.

`killall`

Envoyer un signal à un ou plusieurs processus par leur nom.

`top`

Afficher les processus Linux.

`ps`

Afficher un instantané des processus en cours.

Réponses aux exercices guidés

1. `oneko` est un programme amusant qui affiche un chat en train de poursuivre le curseur de votre souris. S'il n'est pas déjà installé sur votre PC, installez-le en utilisant le gestionnaire de paquets de votre distribution. Nous allons l'utiliser pour étudier le contrôle des tâches.

   - Lancez le programme. Comment faites-vous cela ?

     En tapant `oneko` dans le terminal.

   - Déplacez le curseur de la souris pour voir comment le chat le prend en chasse. Maintenant, suspendez le processus. Comment faites-vous cela ? Quel est le résultat ?

     En appuyant sur la combinaison de touches `Ctrl` + `z`:

     ```
     [1]+  Stopped                 oneko
     ```

   - Vérifiez le nombre de tâches que vous avez actuellement. Qu'est-ce que vous devez taper ? Quel est le résultat ?

     ```
     $ jobs
     [1]+  Stopped                 oneko
     ```

   - Maintenant, envoyez-le en arrière-plan en spécifiant son ID de tâche. Quel est le résultat ? Comment pouvez-vous dire que la tâche est exécutée en arrière-plan ?

     ```
     $ bg %1
     [1]+ oneko &
     ```

     Le chat se déplace à nouveau.

   - Enfin, terminez la tâche en spécifiant son ID. Qu'est-ce que vous devez taper ?

     ```
     $ kill %1
     ```

2. Découvrez les PIDs de tous les processus engendrés par le *serveur web Apache HTTPD* (`apache2`) à l'aide de deux commandes distinctes :

   ```
   $ pgrep apache2
   ```

ou

```
$ pidof apache2
```

3. Terminez tous les processus `apache2` sans utiliser leurs PIDs et en utilisant deux commandes distinctes :

```
$ pkill apache2
```

ou

```
$ killall apache2
```

4. Supposons que vous devez mettre fin à toutes les instances de `apache2` et que vous n'avez pas le temps de trouver leurs PIDs. Comment pouvez-vous faire cela en utilisant `kill` avec le signal `SIGTERM` par défaut en une seule ligne :

```
$ kill $(pgrep apache2)
$ kill `pgrep apache2`
```

ou

```
$ kill $(pidof apache2)
$ kill `pidof apache2`
```

> **NOTE** Puisque `SIGTERM` (`15`) est le signal par défaut, ce n'est pas la peine de passer des options à `kill`.

5. Démarrez `top` et interagissez avec lui en effectuant ce qui suit :
   - Afficher une vue arborescente des processus :

     Taper `V`.

   - Afficher les chemins complets des processus en différenciant l'espace utilisateur et l'espace noyau :

     Taper `c`.

6. Tapez la commande `ps` pour afficher tous les processus démarrés par l'utilisateur du *serveur web Apache HTTPD* (`www-data`) :
   - En utilisant la syntaxte BSD :

```
$ ps U www-data
```

   - En utilisant la syntaxe UNIX :

```
$ ps -u www-data
```

   - En utilisant la syntaxe GNU :

```
$ ps --user www-data
```

Réponses aux exercices d'approfondissement

1. Le signal `SIGHUP` peut être utilisé comme un moyen de redémarrer certains démons. Avec le *serveur web Apache HTTPD* — par exemple — l'envoi de `SIGHUP` au processus parent (celui démarré par `init`) tue ses enfants. Le processus parent, cependant, relit ses fichiers de configuration, rouvre les fichiers journaux et crée un nouvel ensemble de processus enfants. Effectuez les tâches suivantes :

   - Démarrez le serveur web :

     ```
     $ sudo systemctl start apache2
     ```

   - Assurez-vous de connaître le PID du processus parent :

     ```
     $ ps aux | grep apache2
     ```

     Le processus parent est celui lancé par l'utilisateur `root`. Dans notre cas, celui dont le PID est `1653`.

   - Faites redémarrer le serveur web Apache HTTPD en envoyant le signal `SIGHUP` à son processus parent :

     ```
     $ sudo kill -SIGHUP 1653
     ```

   - Vérifiez que le processus parent n'a pas été tué et que de nouveaux processus enfants ont été créés :

     ```
     $ ps aux | grep apache2
     ```

     Maintenant vous devriez voir le processus parent `apache2` avec deux nouveaux processus enfants.

2. Bien que statique à la base, la sortie de `ps` peut être rendue dynamique en combinant `ps` et `watch`. Nous allons surveiller le *serveur web Apache HTTPD* pour détecter les nouvelles connexions. Avant d'effectuer les tâches décrites ci-dessous, il est recommandé de lire la description de la directive `MaxConnectionsPerChild` dans Apache MPM Common Directives.

   - Ajoutez la directive `MaxConnectionsPerChild` avec une valeur de `1` dans le fichier de configuration de `apache2` — dans la famille *Debian* et dérivées on le trouve dans

`/etc/apache2/apache2.conf` ; dans la famille *CentOS*, c'est dans `/etc/httpd/conf/httpd.conf`. N'oubliez pas de redémarrer `apache2` pour que les changements soient pris en compte.

La ligne à inclure dans le fichier de configuration est `MaxConnectionsPerChild 1`. Une façon de redémarrer le serveur web est de taper `sudo systemctl restart apache2`.

- Tapez une commande qui utilise `watch`, `ps` et `grep` pour les connexions `apache2`.

```
$ watch 'ps aux | grep apache2'
```

ou

```
$ watch "ps aux | grep apache2"
```

- Maintenant, ouvrez un navigateur web ou utilisez un navigateur en ligne de commande comme `lynx` pour établir une connexion au serveur web via son adresse IP. Que voyez-vous dans l'affichage de `watch` ?

  Un des processus enfants appartenant à `www-data` disparaît.

3. Comme vous l'avez vu, dans sa configuration par défaut, `top` trie les tâches par pourcentage d'utilisation du CPU par ordre décroissant (les valeurs les plus élevées en haut). Ce comportement peut être modifié avec les raccourcis interactifs `M` (utilisation de la mémoire), `N` (identifiant unique du processus), `T` (temps d'exécution) et `P` (pourcentage du temps CPU). Cependant, vous pouvez également trier la liste des tâches à votre convenance en lançant `top` avec l'option `-o` (pour plus d'informations, consultez la page `man` de top). Maintenant, effectuez les tâches suivantes :

   - Lancez `top` de façon à ce que les tâches soient triées par utilisation de la mémoire :

```
$ top -o %MEM
```

   - Vérifiez si vous avez tapé la bonne commande en mettant la colonne mémoire en surbrillance :

     Appuyez sur `x`.

4. `ps` dispose également d'une option `o` pour spécifier les colonnes que vous souhaitez afficher. Examinez cette alternative et effectuez les tâches suivantes :

- Lancez `ps` de façon à ce que seules les informations sur *l'utilisateur*, *le pourcentage de mémoire utilisée*, *le pourcentage de temps CPU utilisé* et *la commande complète* soient affichées :

```
$ ps o user,%mem,%cpu,cmd
```

- Maintenant, lancez `ps` pour que les seules informations affichées soient celles de l'utilisateur et le nom des programmes qu'il utilise :

```
$ ps o user,comm
```

103.5 Leçon 2

Certification :	LPIC-1
Version :	5.0
Thème :	103 Commandes GNU et Unix
Objectif :	103.5 Créer, surveiller et arrêter des processus
Leçon :	2 sur 2

Introduction

Les outils et programmes abordés dans la leçon précédente sont très utiles pour la surveillance des processus au sens large. Cependant, un administrateur système peut avoir besoin d'aller plus loin. Dans cette leçon, nous allons parler du concept de multiplexeur de terminal et découvrir *GNU Screen* et *tmux*. Même si les émulateurs de terminal actuels sont modernes et performants, les multiplexeurs conservent des fonctionnalités intéressantes et puissantes pour un administrateur système productif.

Caractéristiques des multiplexeurs de terminal

En électronique, un multiplexeur (ou *mux*) est un dispositif qui permet de connecter plusieurs entrées à une seule sortie. Par conséquent, un multiplexeur de terminal nous permet de passer d'une entrée à l'autre selon les besoins. Bien qu'ils ne soient pas tout à fait identiques, `screen` et `tmux` partagent une série de caractéristiques communes :

- Toute invocation valide aboutit au moins à une session qui, à son tour, comprend au moins une fenêtre. Les fenêtres contiennent des programmes.

- Les fenêtres peuvent être subdivisées en régions ou en volets, ce qui peut améliorer la productivité lorsque l'on travaille avec plusieurs programmes en même temps.
- Facilité d'utilisation : pour exécuter la plupart des commandes, il suffit d'utiliser une combinaison de touches appelée *préfixe de commande* ou *touche de commande* suivie d'un autre caractère.
- Les sessions peuvent être détachées de leur terminal en cours (c'est-à-dire que les programmes sont envoyés en arrière-plan et continuent à s'exécuter). Cela garantit l'exécution complète des programmes, même si nous fermons accidentellement un terminal, si le terminal reste ponctuellement bloqué ou si la connexion à distance est perdue.
- Connexion par socket.
- Mode copier/coller.
- Les deux sont hautement configurables.

GNU Screen

Dans les débuts d'Unix (années 70-80), les ordinateurs étaient principalement constitués de terminaux connectés à un ordinateur central. C'était tout, pas de fenêtres multiples ni d'onglets. C'est la raison pour laquelle GNU Screen a été créé en 1987 : émuler plusieurs écrans *VT100* indépendants sur un seul terminal physique.

Les fenêtres

GNU Screen est invoqué en tapant simplement `screen` dans le terminal. Vous verrez alors un message de bienvenue :

```
GNU Screen version 4.05.00 (GNU) 10-Dec-16

Copyright (c) 2010 Juergen Weigert, Sadrul Habib Chowdhury
Copyright (c) 2008, 2009 Juergen Weigert, Michael Schroeder, Micah Cowan, Sadrul Habib
Chowdhury
Copyright (c) 1993-2002, 2003, 2005, 2006, 2007 Juergen Weigert, Michael Schroeder
Copyright (c) 1987 Oliver Laumann
(...)
```

Appuyez sur Espace ou Entrée pour fermer le message et vous vous retrouverez face à une invite de commande :

```
$
```

On pourrait croire que rien ne s'est passé, mais en fait, `screen` a déjà créé et gère sa première session et sa première fenêtre. Le préfixe de commande de Screen est `Ctrl` + `a`. Pour voir toutes les fenêtres en bas de l'écran du terminal, tapez `Ctrl` + `a`-`w` :

```
0*$ bash
```

La voilà, notre seule et unique fenêtre pour le moment ! Notez que le compteur commence à 0. Pour créer une autre fenêtre, tapez `Ctrl` + `a`-`c`. Vous verrez apparaître une nouvelle invite de commande. Lancez `ps` dans cette nouvelle fenêtre :

```
$ ps
  PID TTY          TIME CMD
  974 pts/2    00:00:00 bash
  981 pts/2    00:00:00 ps
```

et tapez à nouveau `Ctrl` + `a`-`w` :

```
0-$ bash  1*$ bash
```

Voilà nos deux fenêtres (notez l'astérisque indiquant la fenêtre qui s'affiche actuellement). Cependant, comme elles ont été lancées avec Bash, elles portent toutes les deux le même nom. Puisque nous avons invoqué `ps` dans notre fenêtre actuelle, renommons-la avec ce même nom. Pour ce faire, vous devez taper `Ctrl` + `a`-`A` et saisir le nouveau nom de la fenêtre (`ps`) à l'invite :

```
Set window's title to: ps
```

Maintenant, nous allons créer une autre fenêtre en lui attribuant un nom dès le départ : `yetanotherwindow`. Cela se fait en invoquant `screen` avec l'option `-t` :

```
$ screen -t yetanotherwindow
```

Vous avez plusieurs possibilités pour passer d'une fenêtre à l'autre :

- En utilisant `Ctrl` + `a`-`n` (*next* - aller à la fenêtre suivante) et `Ctrl` + `a`-`p` (*previous* - aller à la fenêtre précédente).
- En utilisant `Ctrl` + `a`-`num` (aller à la fenêtre numéro *num*).
- En utilisant `Ctrl` + `a`-`"` pour afficher une liste de toutes les fenêtres. Vous pouvez vous déplacer

vers le haut et vers le bas en utilisant les touches fléchées et sélectionner celle que vous voulez avec la touche Entrée :

```
Num Name                                                Flags

   0 bash                                                  $
   1 ps                                                    $
   2 yetanotherwindow
```

Lorsque vous travaillez avec des fenêtres, il est important de garder en tête quelques principes de base :

- Les fenêtres exécutent leurs programmes de manière totalement indépendante les unes des autres.
- Les programmes continuent à s'exécuter même si leur fenêtre n'est pas visible (y compris lorsque la session de Screen est détachée, comme nous allons le voir).

Pour supprimer une fenêtre, il suffit de terminer le programme qui s'y trouve (une fois la dernière fenêtre supprimée, `screen` se terminera lui-même). Vous pouvez également utiliser `Ctrl` + `a` - `k` lorsque vous êtes dans la fenêtre que vous voulez supprimer ; un message de confirmation s'affichera à l'écran :

```
Really kill this window [y/n]

Window 0 (bash) killed.
```

Les régions

`screen` permet de segmenter l'écran d'un terminal en plusieurs régions dans lesquelles on peut placer des fenêtres. Ces subdivisions peuvent être horizontales (`Ctrl` + `a` - `S`) ou verticales (`Ctrl` + `a` - `|`).

La seule chose qui s'affichera dans la nouvelle région sera un simple `--` en bas, ce qui veut dire qu'elle est vide :

```
   1 ps                                                    --
```

Pour vous déplacer vers la nouvelle région, tapez `Ctrl` + `a` - `Tab`. Vous pouvez maintenant insérer une fenêtre par l'une des méthodes que nous avons déjà vues, par exemple : `Ctrl` + `a` - `2`. Le `--` devrait maintenant se transformer en `2 yetanotherwindow` :

```
$ ps                                                    $
  PID TTY          TIME CMD
 1020 pts/2    00:00:00 bash
 1033 pts/2    00:00:00 ps
$ screen -t yetanotherwindow




   1 ps                                                          2 yetanotherwindow
```

Voici quelques points importants qu'il faut garder à l'esprit lorsqu'on utilise les régions :

- Vous vous déplacez entre les régions en tapant `Ctrl` + `a` - `Tab`.
- Vous pouvez faire disparaître toutes les régions à l'exception de celle qui est en cours avec `Ctrl` + `a` - `Q`.
- Vous pouvez faire disparaître la région en cours avec `Ctrl` + `a` - `X`.
- La fermeture d'une région n'entraîne pas la fermeture de la fenêtre qui lui est associée.

Les sessions

Pour l'instant, nous avons manipulé les fenêtres et les régions, qui appartenaient toutes à la même et unique session. Il est temps de commencer à jouer avec les sessions. Pour afficher la liste de toutes les sessions, tapez `screen -list` ou `screen -ls` :

```
$ screen -list
There is a screen on:
        1037.pts-0.debian       (08/24/19 13:53:35)     (Attached)
1 Socket in /run/screen/S-carol.
```

C'est notre seule session pour l'instant :

PID

`1037`

Nom

`pts-0.debian` (indiquant le terminal — dans notre cas un *pseudo-terminal secondaire* — et le nom de l'hôte).

État

`Attached` (attachée)

Nous allons créer une nouvelle session en lui donnant un nom plus parlant :

```
$ screen -S "second session"
```

L'écran du terminal s'efface et une nouvelle invite s'affiche. Vous pouvez à nouveau vérifier les sessions :

```
$ screen -ls
There are screens on:
        1090.second session     (08/24/19 14:38:35)     (Attached)
        1037.pts-0.debian       (08/24/19 13:53:36)     (Attached)
2 Sockets in /run/screen/S-carol.
```

Pour supprimer une session, quittez toutes ses fenêtres ou tapez simplement la commande `screen -S SESSION-PID -X quit` (alternativement vous pouvez fournir le nom de la session). Débarrassons-nous de notre première session :

```
$ screen -S 1037 -X quit
```

Vous serez renvoyé à l'invite de votre terminal en dehors de `screen`. Mais n'oubliez pas que notre deuxième session est toujours active :

```
$ screen -ls
There is a screen on:
    1090.second session (08/24/19 14:38:35) (Detached)
1 Socket in /run/screen/S-carol.
```

En revanche, comme nous avons tué la session parent, une nouvelle étiquette lui a été attribuée : `Detached` (détachée).

Détacher une session

Vous pouvez être amené à détacher une session de `screen` de son terminal pour plusieurs raisons :

- Pour laisser votre ordinateur au travail faire ce qu'il a à faire et vous reconnecter à distance

plus tard depuis chez vous.

- Pour partager une session avec d'autres utilisateurs.

Vous détachez une session avec la combinaison de touches `Ctrl` + `a`-`d`. Vous revenez alors dans votre terminal :

```
[detached from 1090.second session]
$
```

Pour vous rattacher à la session, vous utilisez la commande `screen -r SESSION-PID` (PID de la session). Alternativement, vous pouvez utiliser le `SESSION-NAME` (nom de la session) comme nous l'avons vu plus haut. S'il n'y a qu'une seule session détachée, aucun des deux arguments n'est obligatoire :

```
$ screen -r
```

Cette commande suffit pour nous rattacher à notre deuxième session :

```
$ screen -ls
There is a screen on:
        1090.second session      (08/24/19 14:38:35)     (Attached)
1 Socket in /run/screen/S-carol.
```

Voici quelques options importantes pour rattacher une session :

`-d -r`

Rattacher une session et — si nécessaire — la détacher au préalable.

`-d -R`

Identique à `-d -r` mais `screen` va même créer la session auparavant si elle n'existe pas.

`-d -RR`

Identique à `-d -R`. En revanche, si plusieurs sessions sont disponibles, c'est la première qui sera utilisée.

`-D -r`

Rattacher une session. En cas de besoin, détachez-la et déconnectez-vous à distance dans un premier temps.

`-D -R`

Si une session est en cours, rattachez-la (détachez-la et déconnectez-vous d'abord à distance si nécessaire). Si elle n'était pas active, créez-la et affichez une notification à l'utilisateur.

`-D -RR`

Identique à `-D -R` — mais en insistant davantage.

`-d -m`

Démarre `screen` en *mode détaché*. Cela permet de créer une nouvelle session sans s'y attacher. Cette option est utile pour les scripts de démarrage du système.

`-D -m`

Identique à `-d -m`, mais ne crée pas de nouveau processus. La commande se termine dès que la session se termine.

Lisez les pages du manuel de `screen` pour en savoir plus sur les autres options.

Copier & Coller : le mode défilement

GNU Screen dispose d'un mode copie ou défilement (*scrollback*). Une fois activé, vous pouvez déplacer le curseur dans la fenêtre actuelle et dans son historique à l'aide des touches fléchées. Vous pouvez marquer du texte et le copier d'une fenêtre à l'autre. Voici les étapes à suivre :

1. Activez le mode copie/défilement : `Ctrl` + `a`-`[`.
2. Déplacez le curseur au début du texte à copier à l'aide des touches fléchées.
3. Marquez le début du texte à copier : Espace.
4. Déplacez le curseur à la fin du texte à copier à l'aide des touches fléchées.
5. Marquez la fin du texte à copier : Espace.
6. Basculez vers la fenêtre de votre choix et collez le bout de texte : `Ctrl` + `a`-`]`.

Personnalisation de screen

Le fichier de configuration système pour screen est `/etc/screenrc`. Alternativement, un fichier de configuration utilisateur `~/.screenrc` peut être utilisé. Le fichier comprend quatre sections de configuration principales :

`SCREEN SETTINGS`

Vous pouvez définir des paramètres globaux en spécifiant la *directive* suivie d'une espace et de la *valeur*, comme dans l'exemple suivant : `defscrollback 1024`.

SCREEN KEYBINDINGS

Cette section est assez intéressante puisqu'elle vous permet de redéfinir les combinaisons de touches qui peuvent interférer avec votre utilisation quotidienne du terminal. Utilisez le mot-clé `bind` suivi d'une espace, du caractère à utiliser après le préfixe de la commande, d'un autre espace et de la commande, comme dans : `bind l kill` (ce paramètre changera la façon par défaut de tuer une fenêtre en `Ctrl` + `a`-`l`).

Pour afficher tous les raccourcis clavier de screen, tapez `Ctrl` + `a`-`?` ou consultez la page de manuel en ligne.

TIP Bien entendu, vous pouvez également modifier le préfixe de la commande lui-même. Par exemple, pour passer de `Ctrl` + `a` à `Ctrl` + `b`, il suffit d'ajouter cette ligne : `escape ^Bb`.

TERMINAL SETTINGS

Cette section comprend les paramètres relatifs à la taille des fenêtres du terminal et à la mémoire tampon, entre autres. Pour activer le mode non-bloquant afin de mieux gérer les connexions SSH instables, par exemple, la configuration suivante est utilisée : `defnonblock 5`.

STARTUP SCREENS

Vous pouvez inclure des commandes pour faire tourner certains programmes au démarrage de `screen` ; par exemple : `screen -t top top` (screen ouvrira une fenêtre nommée `top` avec le programme `top` à l'intérieur).

tmux

`tmux` a été publié en 2007. Bien que très similaire à `screen`, il comporte néanmoins quelques différences notables :

- Modèle client-serveur : le serveur fournit une série de sessions, chacune d'entre elles pouvant être associée à un certain nombre de fenêtres qui peuvent, à leur tour, être partagées par différents clients.
- Sélection interactive des sessions, des fenêtres et des clients via des menus.
- Une même fenêtre peut être liée à plusieurs sessions.
- Dispositions clavier pour *vim* et *Emacs*.
- Prise en charge des terminaux UTF-8 et 256 couleurs.

Les fenêtres

`tmux` peut être invoqué en tapant simplement `tmux` à l'invite de commande. Vous verrez apparaître une invite de commande et une barre d'état en bas de la fenêtre :

```
[0] 0:bash*                                                    "debian" 18:53 27-Aug-19
```

Outre le nom d'hôte, l'heure et la date, la barre d'état fournit les informations suivantes :

Nom de la session

`[0]`

Numéro de la fenêtre

`0:`

Nom de la fenêtre

`bash*`. Dans la configuration par défaut, il s'agit du nom du programme qui s'exécute dans la fenêtre et — contrairement à `screen` — `tmux` le mettra automatiquement à jour pour refléter le programme en cours d'exécution. Notez l'astérisque indiquant qu'il s'agit de la fenêtre active et visible.

Vous pouvez affecter un nom de session et un nom de fenêtre lorsque vous invoquez `tmux` :

```
$ tmux new -s "LPI" -n "Window zero"
```

La barre d'état sera modifiée en conséquence :

```
[LPI] 0:Window zero*                                           "debian" 19:01 27-Aug-
19
```

Le préfixe de commande de tmux est `Ctrl` + `b`. Pour créer une nouvelle fenêtre, il suffit de taper `Ctrl` + `b` - `c` ; vous verrez apparaître une nouvelle invite et la barre d'état indiquera la nouvelle fenêtre :

```
[LPI] 0:Window zero- 1:bash*                                   "debian" 19:02 27-Aug-
19
```

Étant donné que Bash est l'interpréteur de commandes utilisé, la nouvelle fenêtre recevra ce nom par défaut. Lancez `top` et observez comment le nom se transforme en `top` :

```
[LPI] 0:Window zero- 1:top*                                    "debian" 19:03 27-Aug-19
```

Dans tous les cas, vous pouvez renommer une fenêtre avec `Ctrl` + `b` - `,`. À l'invite, renseignez le nouveau nom et confirmez en appuyant sur la touche Entrée :

```
(rename-window) Window one
```

Vous pouvez afficher toutes les fenêtres pour une sélection avec `Ctrl` + `b` - `w` (utilisez les touches fléchées pour vous déplacer vers le haut et vers le bas et la touche Entrée pour faire la sélection) :

```
(0)  0: Window zero- "debian"
(1)  1: Window one* "debian"
```

De la même manière que pour `screen`, nous pouvons passer d'une fenêtre à l'autre avec :

`Ctrl` + **`b`** - **`n`**
: passer à la fenêtre suivante.

`Ctrl` + **`b`** - **`p`**
: passer à la fenêtre précédente.

`Ctrl` + **`b`** - **`num`**
: passer à la fenêtre numéro *num*.

Pour vous débarrasser d'une fenêtre, utilisez `Ctrl` + `b` - `&`. Il vous sera demandé de confirmer :

```
kill-window Window one? (y/n)
```

D'autres commandes de fenêtres intéressantes sont disponibles :

`Ctrl` + **`b`** - **`f`**
: rechercher une fenêtre par son nom.

`Ctrl` + **`b`** - **`.`**
: modifier le numéro d'index de la fenêtre.

Pour connaître la liste complète des commandes, consultez la page du manuel.

Les panneaux

La possibilité de diviser les fenêtres de `screen` est également présente dans `tmux`. Les subdivisions résultantes ne sont pas appelées *régions* mais *panneaux*, cependant. La principale différence entre les régions et les panneaux est que ces derniers sont des pseudo-terminaux à part entière liés à une fenêtre. Ce qui signifie que tuer un panneau va également tuer son pseudo-terminal ainsi que tous les programmes associés qui s'exécutent à l'intérieur.

Pour diviser une fenêtre horizontalement, nous utilisons `Ctrl` + `b` - `"` :

```
Tasks:  93 total,   1 running,  92 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem :  4050960 total,  3730920 free,   114880 used,   205160 buff/cache
KiB Swap:  4192252 total,  4192252 free,        0 used.  3716004 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S %CPU %MEM     TIME+ COMMAND
 1340 carol     20   0   44876   3400   2800 R  0.3  0.1   0:00.24 top
    1 root      20   0  139088   6988   5264 S  0.0  0.2   0:00.50 systemd
    2 root      20   0       0      0      0 S  0.0  0.0   0:00.00 kthreadd
    3 root      20   0       0      0      0 S  0.0  0.0   0:00.04 ksoftirqd/0
    4 root      20   0       0      0      0 S  0.0  0.0   0:01.62 kworker/0:0
    5 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 kworker/0:0H
    7 root      20   0       0      0      0 S  0.0  0.0   0:00.06 rcu_sched
    8 root      20   0       0      0      0 S  0.0  0.0   0:00.00 rcu_bh
    9 root      rt   0       0      0      0 S  0.0  0.0   0:00.00 migration/0
   10 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 lru-add-drain
   11 root      rt   0       0      0      0 S  0.0  0.0   0:00.01 watchdog/0
   12 root      20   0       0      0      0 S  0.0  0.0   0:00.00 cpuhp/0
$
_____________________________________________________________________________
___
$
```

```
[LPI] 0:Window zero- 1:Window one*                                    "debian" 19:05 27-
Aug-19
```

Pour la diviser verticalement, utilisez `Ctrl` + `b` - `%`:

```
                                                                      |$
    1 root      20   0  139088   6988   5264 S  0.0  0.2   0:00.50 systemd     |
                                                                      |
    2 root      20   0       0      0      0 S  0.0  0.0   0:00.00 kthreadd    |
                                                                      |
    3 root      20   0       0      0      0 S  0.0  0.0   0:00.04 ksoftirqd/0 |
    4 root      20   0       0      0      0 S  0.0  0.0   0:01.62 kworker/0:0 |
    5 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 kworker/0:0H|
    7 root      20   0       0      0      0 S  0.0  0.0   0:00.06 rcu_sched   |
    8 root      20   0       0      0      0 S  0.0  0.0   0:00.00 rcu_bh      |
    9 root      rt   0       0      0      0 S  0.0  0.0   0:00.00 migration/0 |
                                                                      |
   10 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 lru-add-drai|
n                                                                     |
   11 root      rt   0       0      0      0 S  0.0  0.0   0:00.01 watchdog/0  |
                                                                      |
   12 root      20   0       0      0      0 S  0.0  0.0   0:00.00 cpuhp/0     |
                                                                      |
$                                                                     |
______________________________________________________________________|_______
___
$
```

```
[LPI] 0:Window zero- 1:Window one*                                    "debian" 19:05 27-
Aug-19
```

Pour détruire le panneau actif (ainsi que le pseudo-terminal qui s'y trouve et tous les programmes associés), utilisez `Ctrl` + `b` - `x`. La barre d'état vous demandera de confirmer cette opération :

```
kill-pane 1? (y/n)
```

Quelques commandes importantes pour manipuler les panneaux :

`Ctrl` + **`b`** - **`↑`**, **`↓`**, **`←`**, **`→`**
: se déplacer d'un panneau à l'autre.

`Ctrl` + **`b`** - **`;`**
: revenir vers le dernier panneau actif.

`Ctrl` + **`b`** - **`Ctrl`** + **`touche fléchée`**
: redimensionner le panneau d'une ligne.

`Ctrl` + **`b`** - **`Alt`** + **`touche fléchée`**
: redimensionner le panneau de cinq lignes.

`Ctrl` + **`b`** - **`{`**
: permuter les panneaux (actuel vers précédent).

`Ctrl` + **`b`** - **`}`**
: permuter les panneaux (actuel vers suivant).

`Ctrl` + **`b`** - **`z`**
: zoomer/dézoomer le panneau.

`Ctrl` + **`b`** - **`t`**
: `tmux` affiche une horloge fantaisiste à l'intérieur du panneau (supprimez-la en appuyant sur `q`).

`Ctrl` + **`b`** - **`!`**
: transformer le panneau en fenêtre.

Pour connaître la liste complète des commandes, consultez la page du manuel.

Les sessions

Pour lister les sessions dans `tmux`, vous pouvez utiliser `Ctrl` + `b`-`s`:

```
(0) + LPI: 2 windows (attached)
```

Alternativement, vous pouvez utiliser la commande `tmux ls` :

```
$ tmux ls
LPI: 2 windows (created Tue Aug 27 19:01:49 2019) [158x39] (attached)
```

Il n'y a qu'une seule session (`LPI`) qui comprend deux fenêtres. Créez une nouvelle session depuis la session actuelle. Pour ce faire, utilisez `Ctrl` + `b`, tapez `:new` à l'invite, puis appuyez sur Entrée. Vous serez redirigé vers la nouvelle session, comme en témoigne la barre d'état :

```
[2] 0:bash*                                                    "debian" 19:15 27-Aug-19
```

Par défaut, `tmux` a nommé la session `2`. Pour la renommer, utilisez `Ctrl` + `b`-`$`. À l'invite, fournissez le nouveau nom et appuyez sur Entrée :

```
(rename-session) Second Session
```

Vous pouvez basculer entre les sessions avec `Ctrl` + `b`-`s` (utilisez les touches fléchées et la touche Entrée) :

```
(0) + LPI: 2 windows
(1) + Second Session: 1 windows (attached)
```

Pour tuer une session, vous pouvez utiliser la commande `tmux kill-session -t NOM-SESSION`. Si vous tapez la commande depuis la session en cours, vous sortirez de `tmux` pour revenir dans votre session de terminal initiale :

```
$ tmux kill-session -t "Second Session"
[exited]
$
```

Détacher une session

En tuant `Second Session`, nous avons été éjectés de `tmux`. En revanche, nous avons toujours une session active. Demandez à `tmux` une liste des sessions et vous la retrouverez sûrement :

```
$ tmux ls
LPI: 2 windows (created Tue Aug 27 19:01:49 2019) [158x39]
```

Cependant, cette session est détachée de son terminal. Nous pouvons la rattacher avec `tmux attach -t SESSION-NAME` (`attach` peut être remplacé par `at` ou — plus simplement — `a`). Lorsqu'il n'y a qu'une seule session, la spécification du nom est optionnelle :

```
$ tmux a
```

Vous voilà de retour dans votre session ; pour vous en détacher, appuyez sur `Ctrl` + `b` `d` :

```
[detached (from session LPI)]
$
```

TIP La même session peut être attachée à plus d'un terminal. Si vous voulez attacher une session en vous assurant qu'elle est d'abord détachée de tous les autres terminaux, utilisez l'option `-d` : `tmux attach -d -t NOM-SESSION`.

Commandes importantes pour attacher ou détacher une session :

`Ctrl` + `b` `D`
: sélectionner le client à détacher.

`Ctrl` + `b` `r`
: rafraîchir le terminal du client.

Pour connaître la liste complète des commandes, consultez la page du manuel.

Copier & Coller : le mode défilement

`tmux` propose également un mode copie qui fonctionne pratiquement pareil que celui de `screen` (songez à utiliser le préfixe de commande de `tmux` et non pas celui de `screen` !) La seule différence au niveau des commandes, c'est que vous utiliserez `Ctrl` + Espace pour marquer le début de la sélection et `Alt` + `w` pour copier le texte sélectionné.

Personnalisation de tmux

Les fichiers de configuration de `tmux` se trouvent typiquement dans `/etc/tmux.conf` et dans `~/.tmux.conf`. Lors du lancement, `tmux` recherche ces fichiers s'ils existent. Vous pouvez également lancer `tmux` avec l'option `-f` pour fournir un fichier de configuration alternatif. Un exemple de fichier de configuration `tmux` se trouve dans `/usr/share/doc/tmux/example_tmux.conf`. Le niveau de personnalisation que vous pouvez atteindre est considérable. Parmi les choses que vous pouvez faire, on peut citer :

- Changer le préfixe de commande

```
# Changer le raccourci préfixe en C-a
set -g prefix C-a
unbind C-b
bind C-a send-prefix
```

- Définir des raccourcis clavier supplémentaires pour les fenêtres au-delà de 9

```
# Raccourcis clavier pour sélectionner les fenêtres à numérotation plus élevée
bind F1 selectw -t:10
bind F2 selectw -t:11
bind F3 selectw -t:12
```

Pour obtenir une liste complète de tous les raccourcis clavier, tapez `Ctrl` + `b` - `?` (appuyez sur `q` pour quitter) ou consultez la page du manuel.

Exercices guidés

1. Indiquez si les affirmations/caractéristiques suivantes correspondent à GNU Screen, à tmux ou aux deux :

Affirmation/Caractéristique	**GNU Screen**	**tmux**
Le préfixe de commande par défaut est `Ctrl` + `a`		
Modèle client-serveur		
Les panneaux sont des pseudo-terminaux		
La suppression d'une région n'entraîne pas la suppression de la (des) fenêtre(s) associée(s)		
Les sessions comprennent des fenêtres		
Les sessions peuvent être détachées		

2. Installez GNU Screen sur votre ordinateur (nom du paquet : `screen`) et effectuez les tâches suivantes :
   - Lancez le programme. Quelle commande utilisez-vous ?
   - Lancez `top`:
   - En utilisant le préfixe clavier de screen, ouvrez une nouvelle fenêtre ; ensuite, ouvrez `/etc/screenrc` avec `vi` :
   - Affichez la liste des fenêtres en bas de l'écran :
   - Renommez la fenêtre active en `vi` :

- Renommez la fenêtre restante en `top`. Pour ce faire, affichez d'abord une liste de toutes les fenêtres afin de pouvoir vous déplacer vers le haut et vers le bas et sélectionnez la bonne :

- Vérifiez si les noms ont bien changé en affichant à nouveau les noms des fenêtres en bas de l'écran :

- Maintenant, détachez la session et demandez à `screen` d'en créer une autre nommée `ssh` :

- Détachez-vous de la session `ssh` et demandez à `screen` d'afficher la liste des sessions :

- Maintenant, attachez-vous à la première session en utilisant son PID :

- Vous devriez être revenu à la fenêtre qui affiche `top`. Partagez la fenêtre horizontalement et déplacez-vous vers la nouvelle région vide :

- Demandez à `screen` de lister toutes les fenêtres et sélectionnez `vi` pour l'afficher dans la nouvelle région vide :

- Maintenant, scindez la région actuelle dans le sens vertical, déplacez-vous vers la région vide nouvellement créée et associez-la à une toute nouvelle fenêtre :

- Terminez toutes les régions à l'exception de la région actuelle (rappelez-vous que même si vous supprimez les régions, les fenêtres sont toujours actives). Ensuite, quittez toutes les fenêtres de la session en cours jusqu'à ce que la session elle-même soit terminée :

- Enfin, demandez à `screen` de lister ses sessions une dernière fois, tuez la session `ssh` restante par son PID et vérifiez qu'il n'y a effectivement plus de sessions :

3. Installez `tmux` sur votre ordinateur (nom du paquet : `tmux`) et effectuez les tâches suivantes :
   - Lancez le programme. Quelle commande utilisez-vous ?
   - Lancez `top` (notez comment — au bout de quelques secondes — le nom de la fenêtre est remplacé par `top` dans la barre d'état) :
   - En utilisant le préfixe clavier de tmux, ouvrez une nouvelle fenêtre ; puis, créez `~/.tmux.conf` en utilisant `nano` :
   - Scindez la fenêtre verticalement et réduisez à plusieurs reprises la taille du panneau nouvellement créé :
   - Maintenant, changez le nom de la fenêtre courante en `text editing` ; ensuite, demandez à `tmux` d'afficher une liste de toutes ses sessions :
   - Passez à la fenêtre où se trouve `top` et revenez à la fenêtre actuelle en utilisant la même combinaison de touches :
   - Détachez-vous de la session en cours et créez-en une nouvelle nommée `ssh` avec une fenêtre `ssh window` :
   - Détachez-vous également de la session `ssh` et demandez à `tmux` de réafficher la liste des sessions :

NOTE

A partir de là, l'exercice requiert l'utilisation d'une machine *distante* pour les connexions `ssh` à votre hôte local (une machine virtuelle est parfaitement valide et peut s'avérer très pratique). Assurez-vous que `openssh-server` est installé et fonctionne sur votre machine locale et qu'au minimum `openssh-`

> `client` est installé sur la machine distante.

- Maintenant, démarrez une machine distante et connectez-vous via `ssh` à votre hôte local. Une fois la connexion établie, vérifiez la présence de sessions `tmux` :

- Sur l'hôte distant, attachez-vous à la session `ssh` par le nom :

- De retour sur votre machine locale, attachez-vous à la session `ssh` par le nom en vous assurant que la connexion à l'hôte distant est terminée au préalable :

- Affichez la sélection de toutes les sessions et allez à la première session (`[0]`). Une fois que vous y êtes, tuez la session `ssh` par le nom :

- Enfin, détachez-vous de la session en cours et tuez-la par le nom :

Exercices d'approfondissement

1. `screen` et `tmux` peuvent tous deux entrer en mode commande par le biais du *préfixe de commande* + `:` (nous avons déjà vu un bref exemple avec `tmux`). Faites quelques recherches et effectuez les tâches suivantes en mode commande :
   - Faites passer `screen` en mode copie :
   - Renommez la fenêtre en cours avec `tmux` :
   - Fermez toutes les fenêtres de `screen` et terminez la session :
   - Scindez un panneau en deux avec `tmux` :
   - Tuez la fenêtre active avec `tmux` :
2. Lorsque vous activez le mode copie dans `screen`, vous pouvez non seulement utiliser les touches fléchées et `PageHaut` ou `PageBas` pour naviguer dans la fenêtre courante et la mémoire tampon de défilement. Il est également possible d'utiliser un éditeur plein écran de type `vi`. En utilisant cet éditeur, effectuez les tâches suivantes :
   - Affichez `supercalifragilisticexpialidocious` dans votre terminal `screen` :
   - Maintenant, copiez cinq caractères consécutifs (de gauche à droite) dans la ligne située juste au-dessus de votre curseur :
   - Pour finir, on colle la sélection (`stice`) à l'invite de commande :
3. Admettons que vous voulez partager une session `tmux` (`our_session`) avec un autre utilisateur. Vous avez créé le socket (`/tmp/our_socket`) avec les bonnes permissions pour que

vous et l'autre utilisateur puissiez lire et écrire. Quelles sont les deux autres conditions qui doivent être remplies pour que le second utilisateur puisse attacher avec succès la session par le biais de `tmux -S /tmp/our_socket a -t our_session` ?

Résumé

Dans cette leçon, vous avez découvert les *multiplexeurs de terminaux* en général et GNU Screen et tmux en particulier. Voici les concepts importants à retenir :

- Préfixe de commande : `screen` utilise `Ctrl` + `a` + *caractère* ; `tmux` utilise `Ctrl` + `b` + *caractère*.
- Structure des sessions, des fenêtres et des subdivisions de fenêtres (régions ou panneaux).
- Mode copier/coller.
- Détachement de session : l'une des fonctionnalités les plus puissantes des multiplexeurs.

Les commandes suivantes ont été abordées dans cette leçon :

`screen`
: Démarrer une session `screen`.

`tmux`
: Démarrer une session `tmux`.

Réponses aux exercices guidés

1. Indiquez si les affirmations/caractéristiques suivantes correspondent à GNU Screen, à tmux ou aux deux :

Affirmation/Caractéristique	GNU Screen	tmux
Le préfixe de commande par défaut est `Ctrl` + `a`	x	
Modèle client-serveur		x
Les panneaux sont des pseudo-terminaux		x
La suppression d'une région n'entraîne pas la suppression de la (des) fenêtre(s) associée(s)	x	
Les sessions comprennent des fenêtres	x	x
Les sessions peuvent être détachées	x	x

2. Installez GNU Screen sur votre ordinateur (nom du paquet : `screen`) et effectuez les tâches suivantes :
   - Lancez le programme. Quelle commande utilisez-vous ?

     `screen`
   - Lancez `top`:

     `top`
   - En utilisant le préfixe clavier de screen, ouvrez une nouvelle fenêtre ; ensuite, ouvrez `/etc/screenrc` avec `vi` :

     `Ctrl` + `a` - `c`

     `sudo vi /etc/screenrc`
   - Affichez la liste des fenêtres en bas de l'écran :

`Ctrl` + `a` - `w`

- Renommez la fenêtre active en `vi` :

  `Ctrl` + `a` - `A`. Ensuite, il faut taper `vi` et appuyer sur Entrée.

- Renommez la fenêtre restante en `top`. Pour ce faire, affichez d'abord une liste de toutes les fenêtres afin de pouvoir vous déplacer vers le haut et vers le bas et sélectionnez la bonne :

  Tout d'abord, nous tapons `Ctrl` + `a` - `"`. Ensuite, nous utilisons les touches fléchées pour marquer la fenêtre qui correspond à `0 bash` et nous appuyons sur Entrée. Enfin, nous tapons `Ctrl` + `a` - `A`, nous tapons `top` et nous appuyons sur Entrée.

- Vérifiez si les noms ont bien changé en affichant à nouveau les noms des fenêtres en bas de l'écran :

  `Ctrl` + `a` - `w`

- Maintenant, détachez la session et demandez à `screen` d'en créer une autre nommée `ssh` :

  `Ctrl` + `a` - `d` `screen -S "ssh"` et appuyez sur Entrée.

- Détachez-vous de la session `ssh` et demandez à `screen` d'afficher la liste des sessions :

  `Ctrl` + `a` - `d` `screen -list` ou `screen -ls`.

- Maintenant, attachez-vous à la première session en utilisant son PID :

  `screen -r PID-SESSION`

- Vous devriez être revenu à la fenêtre qui affiche `top`. Partagez la fenêtre horizontalement et déplacez-vous vers la nouvelle région vide :

  `Ctrl` + `a` - `S`

  `Ctrl` + `a` - `Tab`

- Demandez à `screen` de lister toutes les fenêtres et sélectionnez `vi` pour l'afficher dans la nouvelle région vide :

  Nous utilisons `Ctrl` + `a` - `"` pour afficher toutes les fenêtres à sélectionner, marquer `vi` et appuyer sur Entrée.

- Maintenant, scindez la région actuelle dans le sens vertical, déplacez-vous vers la région

vide nouvellement créée et associez-la à une toute nouvelle fenêtre :

`Ctrl` + `a` - `|`

`Ctrl` + `a` - `Tab`

`Ctrl` + `a` - `c`

- Terminez toutes les régions à l'exception de la région actuelle (rappelez-vous que même si vous supprimez les régions, les fenêtres sont toujours actives). Ensuite, quittez toutes les fenêtres de la session en cours jusqu'à ce que la session elle-même soit terminée :

  `Ctrl` + `a` - `Q`. `exit` (pour quitter Bash). `Maj` + `:`, puis nous tapons `quit` et appuyons sur Entrée (pour quitter `vi`). Ensuite, nous tapons `exit` (pour quitter le shell Bash en cours) et `q` (pour terminer `top`) ; puis nous tapons `exit` (pour quitter le shell Bash en cours).

- Enfin, demandez à `screen` de lister ses sessions une dernière fois, tuez la session `ssh` restante par son PID et vérifiez qu'il n'y a effectivement plus de sessions :

  `screen -list` ou `screen -ls`

  `screen -S PID-OF-SESSION -X quit`

  `screen -list` or `screen -ls`

3. Installez `tmux` sur votre ordinateur (nom du paquet : `tmux`) et effectuez les tâches suivantes :
   - Lancez le programme. Quelle commande utilisez-vous ?

     `tmux`

   - Lancez `top` (notez comment — au bout de quelques secondes — le nom de la fenêtre est remplacé par `top` dans la barre d'état) :

     `top`

   - En utilisant le préfixe clavier de tmux, ouvrez une nouvelle fenêtre ; puis, créez `~/.tmux.conf` en utilisant `nano` :

     `Ctrl` + `b` - `c` `nano ~/.tmux.conf`

   - Scindez la fenêtre verticalement et réduisez à plusieurs reprises la taille du panneau nouvellement créé :

     `Ctrl` + `b` - `"`

`Ctrl` + `b` - `Ctrl` + `↓`

- Maintenant, changez le nom de la fenêtre courante en `text editing` ; ensuite, demandez à `tmux` d'afficher une liste de toutes ses sessions :

  `Ctrl` + `b` - `,`. Ensuite, nous fournissons le nouveau nom et nous appuyons sur Entrée. `Ctrl` + `b` - `s` ou `tmux ls`.

- Passez à la fenêtre où se trouve `top` et revenez à la fenêtre actuelle en utilisant la même combinaison de touches :

  `Ctrl` + `b` - `n` ou `Ctrl` + `b` - `p`

- Détachez-vous de la session en cours et créez-en une nouvelle nommée `ssh` avec une fenêtre `ssh window` :

  `Ctrl` + `b` - `d` `tmux new -s "ssh" -n "ssh window"`

- Détachez-vous également de la session `ssh` et demandez à `tmux` de réafficher la liste des sessions :

  `Ctrl` + `b` - `d` `tmux ls`

> **NOTE** A partir de là, l'exercice requiert l'utilisation d'une machine *distante* pour les connexions `ssh` à votre hôte local (une machine virtuelle est parfaitement valide et peut s'avérer très pratique). Assurez-vous que `openssh-server` est installé et fonctionne sur votre machine locale et qu'au minimum `openssh-client` est installé sur la machine distante.

- Maintenant, démarrez une machine distante et connectez-vous via `ssh` à votre hôte local. Une fois la connexion établie, vérifiez la présence de sessions `tmux` :

  Sur l'hôte distant : `ssh utilisateur-local@adresse-ip-locale`. Une fois connecté à la machine locale : `tmux ls`.

- Sur l'hôte distant, attachez-vous à la session `ssh` par le nom :

  `tmux a -t ssh` (`a` peut être remplacé par `at` ou `attach`).

- De retour sur votre machine locale, attachez-vous à la session `ssh` par le nom en vous assurant que la connexion à l'hôte distant est terminée au préalable :

  `tmux a -d -t ssh` (`a` peut être remplacé par `at` ou `attach`).

- Affichez la sélection de toutes les sessions et allez à la première session (`[0]`). Une fois que vous y êtes, tuez la session `ssh` par le nom :

  Tapez `Ctrl` + `b` - `s`, utilisez les touches fléchées pour marquer la session `0` et appuyez sur Entrée `tmux kill-session -t ssh`.

- Enfin, détachez-vous de la session en cours et tuez-la par le nom :

  `Ctrl` + `b` - `d` `tmux kill-session -t 0`.

Réponses aux exercices d'approfondissement

1. `screen` et `tmux` peuvent tous deux entrer en mode commande par le biais du *préfixe de commande* + `:` (nous avons déjà vu un bref exemple avec `tmux`). Faites quelques recherches et effectuez les tâches suivantes en mode commande :
   - Faites passer `screen` en mode copie :

     `Ctrl` + `a`-`:` — ensuite on tape `copy`.
   - Renommez la fenêtre en cours avec `tmux` :

     `Ctrl` + `b`-`:` — ensuite on tape `rename-window`.
   - Fermez toutes les fenêtres de `screen` et terminez la session :

     `Ctrl` + `a`-`:` — ensuite on tape `quit`.
   - Scindez un panneau en deux avec `tmux` :

     `Ctrl` + `b`-`:` — ensuite on tape `split-window`.
   - Tuez la fenêtre active avec `tmux` :

     `Ctrl` + `b`-`:` — ensuite on tape `kill-window`.
2. Lorsque vous activez le mode copie dans `screen`, vous pouvez non seulement utiliser les touches fléchées et `PageHaut` ou `PageBas` pour naviguer dans la fenêtre courante et la mémoire tampon de défilement. Il est également possible d'utiliser un éditeur plein écran de type `vi`. En utilisant cet éditeur, effectuez les tâches suivantes :
   - Affichez `supercalifragilisticexpialidocious` dans votre terminal `screen` :

     ```
     echo supercalifragilisticexpialidocious
     ```
   - Maintenant, copiez cinq caractères consécutifs (de gauche à droite) dans la ligne située juste au-dessus de votre curseur :

     On active le mode copie : `Ctrl` + `a`-`[` ou `Ctrl` + `a`-`:` puis on tape `copy`. Ensuite, on se déplace vers la ligne supérieure en utilisant `k` et on appuie sur la touche Espace pour marquer le début de la sélection. Enfin, on avance de quatre caractères en utilisant `l` et on appuie à nouveau sur Espace pour marquer la fin de la sélection.
   - Pour finir, on colle la sélection (`stice`) à l'invite de commande :

`Ctrl` + `a` - `]`

3. Admettons que vous voulez partager une session `tmux` (`our_session`) avec un autre utilisateur. Vous avez créé le socket (`/tmp/our_socket`) avec les bonnes permissions pour que vous et l'autre utilisateur puissiez lire et écrire. Quelles sont les deux autres conditions qui doivent être remplies pour que le second utilisateur puisse attacher avec succès la session par le biais de `tmux -S /tmp/our_socket a -t our_session` ?

   Les deux utilisateurs doivent avoir un groupe en commun, par exemple `multiplexer`. Ensuite, nous devons également attribuer le socket à ce groupe : `chgrp multiplexer /tmp/our_socket`.

103.6 Modification des priorités des processus

Référence aux objectifs de LPI

LPIC-1 v5, Exam 101, Objective 103.6

Valeur

2

Domaines de connaissance les plus importants

- Connaissance de la priorité par défaut affectée à un nouveau processus.
- Exécution de programme avec une priorité plus haute ou plus basse que celle par défaut.
- Changement de la priorité d'un processus en cours d'exécution.

Liste partielle de termes, fichiers et utilitaires utilisés pour cet objectif

- `nice`
- `ps`
- `renice`
- `top`

Linux Professional Institute

103.6 Leçon 1

Certification :	LPIC-1
Version :	5.0
Thème :	103 Commandes GNU et Unix
Objectif :	103.6 Modifier les priorités d'exécution des processus
Leçon :	1 sur 1

Introduction

Les systèmes d'exploitation capables d'exécuter plus d'un processus en même temps sont appelés systèmes multitâches ou multiprocessus. Si la véritable simultanéité ne se produit que lorsque plus d'un processeur (CPU) est disponible, même les systèmes dotés d'un seul processeur peuvent reproduire un semblant de simultanéité en passant très rapidement d'un processus à l'autre. Cette technique est également utilisée dans les systèmes dotés de plusieurs CPUs équivalents ou systèmes à *multiprocesseurs symétriques (SMP)*, étant donné que le nombre de processus simultanés potentiels dépasse largement le nombre de CPUs disponibles.

En fait, un seul processus à la fois peut contrôler le CPU. Cependant, la plupart des activités des processus sont des *appels système*, c'est-à-dire que le processus en cours d'exécution transfère le contrôle du CPU à un processus du système d'exploitation pour qu'il effectue l'opération requise. Les appels système se chargent de toute la communication entre périphériques comme l'allocation de mémoire, la lecture et l'écriture sur les systèmes de fichiers, l'affichage de texte à l'écran, l'interaction avec l'utilisateur, les transferts réseau, etc. Le transfert du contrôle du CPU pendant un appel système permet au système d'exploitation de décider s'il doit rendre le contrôle du CPU au processus précédent ou le céder à un autre processus. Comme les CPU modernes sont capables

d'exécuter des instructions beaucoup plus rapidement que la plupart des périphériques externes ne peuvent communiquer entre eux, un nouveau processus de contrôle peut effectuer une grande partie du travail du CPU alors même que les réponses matérielles précédemment sollicitées ne sont pas encore disponibles. Pour garantir une optimisation de l'utilisation du CPU, les systèmes d'exploitation multiprocessus gèrent une file d'attente dynamique de processus actifs en attente d'un créneau de temps du CPU.

Même s'ils permettent d'améliorer considérablement l'utilisation du temps CPU, le fait de s'appuyer uniquement sur les appels système pour passer d'un processus à l'autre n'est pas suffisant pour obtenir des performances multitâches satisfaisantes. Un processus qui ne fait aucun appel au système pourrait contrôler le CPU indéfiniment. C'est pourquoi les systèmes d'exploitation modernes sont également *préemptifs*, c'est-à-dire qu'un processus en cours d'exécution peut être remis dans la file d'attente afin qu'un processus plus important puisse contrôler le CPU, même si le processus en cours d'exécution n'a pas effectué d'appel système.

L'ordonnanceur Linux

En tant que système d'exploitation multiprocessus préemptif, Linux implémente un ordonnanceur chargé d'organiser la file d'attente des processus. Plus précisément, l'ordonnanceur décide également quel *thread* (processus léger) de la file d'attente sera exécuté — un processus peut engendrer toute une série de *threads* indépendants — mais processus et *thread* sont des termes interchangeables dans ce contexte. Chaque processus est associé à deux paramètres qui déterminent son ordonnancement : la *politique d'ordonnancement* et la *priorité d'ordonnancement*.

On distingue deux principales catégories de politiques d'ordonnancement : les *politiques temps réel* et les *politiques normales*. Les processus soumis à une politique temps réel sont ordonnancés directement en fonction de leurs valeurs de priorité. Si un processus plus important est prêt à s'exécuter, un processus moins important en cours d'exécution est préempté et le processus prioritaire prend le contrôle du CPU. Un processus de priorité inférieure ne prendra le contrôle du CPU que lorsque les processus de priorité supérieure sont inactifs ou en attente d'une réponse du matériel.

Tous les processus temps réel ont une priorité plus élevée que les processus normaux. En tant que système d'exploitation polyvalent, Linux n'exécute que très peu de processus temps réel. La plupart des processus, y compris les programmes système et utilisateur, s'exécutent selon des politiques d'ordonnancement normales. Les processus normaux ont généralement la même priorité, mais les politiques normales peuvent définir des règles de priorité d'exécution à l'aide d'un autre paramètre de processus : la *valeur nice*. Pour éviter toute confusion avec les priorités dynamiques dérivées des valeurs *nice*, les priorités d'ordonnancement sont généralement appelées priorités d'ordonnancement *statiques*.

L'ordonnanceur Linux peut être configuré de plusieurs manières et il existe même des méthodes plus complexes pour établir les priorités, mais ces concepts généraux restent toujours valables. Lorsqu'on inspecte et qu'on règle l'ordonnancement des processus, il est important de garder à l'esprit que seuls les processus soumis à une politique d'ordonnancement normale seront affectés.

Comprendre les priorités

Linux réserve des priorités statiques allant de 0 à 99 aux processus temps réel tandis que les processus ordinaires se voient attribuer des priorités statiques allant de 100 à 139, ce qui implique qu'il existe 39 niveaux de priorité différents pour les processus normaux. Les valeurs plus basses correspondent à une priorité plus élevée. La priorité statique d'un processus actif se trouve dans le fichier `sched`, situé dans le répertoire correspondant du système de fichiers `/proc` :

```
$ grep ^prio /proc/1/sched
prio                  :       120
```

Comme le montre l'exemple, la ligne commençant par `prio` indique la valeur de priorité du processus (le processus PID 1 est le processus *init* ou *systemd*, le premier processus lancé par le noyau lors du démarrage du système). La priorité standard des processus normaux est de 120, elle peut donc être ramenée à 100 ou augmentée à 139. Les priorités de tous les processus en cours d'exécution peuvent être consultées à l'aide de la commande `ps -Al` ou `ps -el` :

```
$ ps -el
F S   UID   PID  PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
4 S     0     1     0  0  80   0 -  9292 -      ?        00:00:00 systemd
4 S     0    19     1  0  80   0 -  8817 -      ?        00:00:00 systemd-journal
4 S   104    61     1  0  80   0 - 64097 -      ?        00:00:00 rsyslogd
4 S     0    63     1  0  80   0 -  7244 -      ?        00:00:00 cron
1 S     0   126     1  0  80   0 -  4031 -      ?        00:00:00 dhclient
4 S     0   154     1  0  80   0 -  3937 -      pts/0    00:00:00 agetty
4 S     0   155     1  0  80   0 -  3937 -      pts/1    00:00:00 agetty
4 S     0   156     1  0  80   0 -  3937 -      pts/2    00:00:00 agetty
4 S     0   157     1  0  80   0 -  3937 -      pts/3    00:00:00 agetty
4 S     0   158     1  0  80   0 -  3937 -      console  00:00:00 agetty
4 S     0   160     1  0  80   0 - 16377 -      ?        00:00:00 sshd
4 S     0   280     0  0  80   0 -  5301 -      ?        00:00:00 bash
0 R     0   392   280  0  80   0 -  7221 -      ?        00:00:00 ps
```

La colonne `PRI` indique la priorité statique assignée par le noyau. On remarquera cependant que la valeur de la priorité affichée par `ps` diffère de celle observée dans l'exemple ci-dessus. Pour des

raisons historiques, les priorités affichées par `ps` vont de -40 à 99 par défaut, si bien que la priorité réelle est calculée en y ajoutant 40 (notamment 80 + 40 = 120).

On peut également surveiller à chaud les processus gérés par le noyau Linux avec le programme `top`. Comme pour `ps`, `top` affiche la valeur de la priorité à sa façon. Pour faciliter l'identification des processus temps réel, `top` soustrait la valeur de la priorité de 100, ce qui rend toutes les priorités temps réel négatives, avec un nombre négatif ou *rt* qui les identifie. Par conséquent, les priorités normales affichées par `top` vont de 0 à 39.

NOTE

Pour obtenir plus de détails de la commande `ps`, on peut utiliser des options supplémentaires. Comparez l'affichage de cette commande à celui de l'exemple précédent :

```
$ ps -e -o user,uid,comm,tty,pid,ppid,pri,pmem,pcpu --sort=-pcpu | head
```

Le degré de priorité des processus

Chaque processus normal démarre avec une valeur `nice` par défaut de 0 (priorité 120). Le nom *nice* (gentil, sympathique) vient de l'idée que les processus "plus sympathiques" permettent à d'autres processus de s'exécuter avant eux dans une file d'attente donnée. Les valeurs `nice` vont de -20 (moins *nice*, haute priorité) à 19 (plus *nice*, basse priorité). Linux permet également d'assigner des valeurs différentes aux *threads* d'un même processus. La colonne `NI` dans la sortie `ps` indique le numéro *nice*.

Seul l'utilisateur root peut réduire la priorité d'un processus en dessous de zéro. Vous pouvez démarrer un processus avec une priorité différente de la normale avec la commande `nice`. Par défaut, `nice` modifie la priorité à 10, mais cette valeur peut être modifiée avec l'option `-n` :

```
$ nice -n 15 tar czf home_backup.tar.gz /home
```

Dans cet exemple, la commande `tar` est exécutée avec une priorité de 15. La commande `renice` peut être utilisée pour modifier la priorité d'un processus en cours d'exécution. L'option `-p` indique le numéro PID du processus cible. Par exemple :

```
# renice -10 -p 2164
2164 (process ID) old priority 0, new priority -10
```

Les options `-g` et `-u` sont utilisées pour modifier tous les processus d'un groupe ou d'un utilisateur donné, respectivement. Avec `renice +5 -g users`, le degré d'agrément des processus

appartenant aux utilisateurs du groupe *users* sera augmenté de cinq.

En dehors de `renice`, la priorité des processus peut être modifiée à l'aide d'autres programmes, comme `top`. Dans l'écran principal de `top`, la priorité d'un processus peut être modifiée en appuyant sur `r` puis en renseignant le numéro PID du processus :

```
top - 11:55:21 up 23:38,  1 user,  load average: 0,10, 0,04, 0,05
Tasks:  20 total,   1 running,  19 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0,5 us,  0,3 sy,  0,0 ni, 99,0 id,  0,0 wa,  0,2 hi,  0,0 si,  0,0 st
KiB Mem :  4035808 total,   774700 free,  1612600 used,  1648508 buff/cache
KiB Swap:  7999828 total,  7738780 free,   261048 used.  2006688 avail Mem
PID to renice [default pid = 1]
  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
    1 root      20   0   74232   7904   6416 S 0,000 0,196   0:00.12 systemd
   15 root      20   0   67436   6144   5568 S 0,000 0,152   0:00.03 systemd-journal
   21 root      20   0   61552   5628   5000 S 0,000 0,139   0:00.01 systemd-logind
   22 message+  20   0   43540   4072   3620 S 0,000 0,101   0:00.03 dbus-daemon
   23 root      20   0   45652   6204   4992 S 0,000 0,154   0:00.06 wickedd-dhcp4
   24 root      20   0   45648   6276   5068 S 0,000 0,156   0:00.06 wickedd-auto4
   25 root      20   0   45648   6272   5060 S 0,000 0,155   0:00.06 wickedd-dhcp6
```

Le message `PID to renice [default pid = 1]` apparaît avec le premier processus listé sélectionné par défaut. Pour modifier la priorité d'un autre processus, saisissez son PID et appuyez sur Entrée. Le message `Renice PID 1 to value` apparaît alors (avec le numéro de PID demandé) et une nouvelle valeur pourra être attribuée.

Exercices guidés

1. Dans un système multitâche préemptif, que se passe-t-il lorsqu'un processus de moindre priorité occupe le processeur et qu'un processus de priorité plus élevée est en attente d'exécution ?

2. Prenons l'écran `top` ci-dessous :

```
top - 08:43:14 up 23 days, 12:29,  5 users,  load average: 0,13, 0,18, 0,21
Tasks: 240 total,   2 running, 238 sleeping,   0 stopped,   0 zombie
%Cpu(s):  1,4 us,  0,4 sy,  0,0 ni, 98,1 id,  0,0 wa,  0,0 hi,  0,0 si,  0,0 st
MiB Mem :   7726,4 total,    590,9 free,   1600,8 used,   5534,7 buff/cache
MiB Swap:  30517,0 total,  30462,5 free,     54,5 used.   5769,4 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
    1 root      20   0  171420  10668   7612 S   0,0   0,1   9:59.15 systemd
    2 root      20   0       0      0      0 S   0,0   0,0   0:02.76 kthreadd
    3 root       0 -20       0      0      0 I   0,0   0,0   0:00.00 rcu_gp
    4 root       0 -20       0      0      0 I   0,0   0,0   0:00.00 rcu_par_gp
    8 root       0 -20       0      0      0 I   0,0   0,0   0:00.00 mm_percpu_wq
    9 root      20   0       0      0      0 S   0,0   0,0   0:49.06 ksoftirqd/0
   10 root      20   0       0      0      0 I   0,0   0,0  18:24.20 rcu_sched
   11 root      20   0       0      0      0 I   0,0   0,0   0:00.00 rcu_bh
   12 root      rt   0       0      0      0 S   0,0   0,0   0:08.17 migration/0
   14 root      20   0       0      0      0 S   0,0   0,0   0:00.00 cpuhp/0
   15 root      20   0       0      0      0 S   0,0   0,0   0:00.00 cpuhp/1
   16 root      rt   0       0      0      0 S   0,0   0,0   0:11.79 migration/1
   17 root      20   0       0      0      0 S   0,0   0,0   0:26.01 ksoftirqd/1
```

   Quels sont les PID qui ont des priorités temps réel ?

3. Considérons le listing `ps -el` ci-dessous :

```
F S   UID   PID  PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
4 S     0     1     0  0  80   0 - 42855 -      ?        00:09:59 systemd
1 S     0     2     0  0  80   0 -     0 -      ?        00:00:02 kthreadd
1 I     0     3     2  0  60 -20 -     0 -      ?        00:00:00 rcu_gp
1 S     0     9     2  0  80   0 -     0 -      ?        00:00:49 ksoftirqd/0
1 I     0    10     2  0  80   0 -     0 -      ?        00:18:26 rcu_sched
```

```
1 I     0    11     2  0  80   0 -     0 -      ?        00:00:00 rcu_bh
1 S     0    12     2  0 -40   - -     0 -      ?        00:00:08 migration/0
1 S     0    14     2  0  80   0 -     0 -      ?        00:00:00 cpuhp/0
5 S     0    15     2  0  80   0 -     0 -      ?        00:00:00 cpuhp/1
```

Quel PID a la plus grande priorité ?

4. Vous essayez de modifier la priorité d'un processus avec `renice` et vous vous retrouvez confronté à l'erreur suivante :

```
$ renice -10 21704
renice: failed to set priority for 21704 (process ID): Permission denied
```

Quelle est la cause probable de l'erreur ?

Exercices d'approfondissement

1. Lorsqu'un processus occupe trop de temps CPU, il faut généralement modifier les priorités de ce processus. En utilisant `ps` avec les options standard pour afficher tous les processus du système dans un format long, quelle option `--sort` va trier les processus par utilisation du CPU, dans un ordre croissant ?

2. La commande `schedtool` peut définir tous les paramètres d'ordonnancement du CPU dont Linux est capable ou afficher des informations pour des processus donnés. Comment peut-on l'utiliser pour afficher les paramètres d'ordonnancement du processus `1750` ? Par ailleurs, comment utiliser `schedtool` pour basculer le processus 1750 en temps réel avec une priorité de -90 (telle qu'affichée par `top`) ?

Résumé

Cette leçon explique comment Linux partage le temps de CPU entre les processus qu'il gère. Pour garantir les meilleures performances, les processus critiques doivent prendre le pas sur les processus moins critiques. La leçon passe en revue les étapes suivantes :

- Concepts de base des systèmes multiprocessus.
- Qu'est-ce qu'un ordonnanceur de processus et comment Linux le met en œuvre.
- Les priorités de Linux, les valeurs *nice* et leur utilité.
- Comment lire et interpréter les priorités des processus sous Linux.
- Modifier la priorité d'un processus, avant et pendant son exécution.

Réponses aux exercices guidés

1. Dans un système multitâche préemptif, que se passe-t-il lorsqu'un processus de moindre priorité occupe le processeur et qu'un processus de priorité plus élevée est en attente d'exécution ?

   Le processus de moindre priorité est mis en attente et le processus de plus haute priorité est exécuté à sa place.

2. Prenons l'écran `top` ci-dessous :

```
top - 08:43:14 up 23 days, 12:29,  5 users,  load average: 0,13, 0,18, 0,21
Tasks: 240 total,   2 running, 238 sleeping,   0 stopped,   0 zombie
%Cpu(s):  1,4 us,  0,4 sy,  0,0 ni, 98,1 id,  0,0 wa,  0,0 hi,  0,0 si,  0,0 st
MiB Mem :   7726,4 total,    590,9 free,   1600,8 used,   5534,7 buff/cache
MiB Swap:  30517,0 total,  30462,5 free,     54,5 used.   5769,4 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
    1 root      20   0  171420  10668   7612 S   0,0   0,1   9:59.15 systemd
    2 root      20   0       0      0      0 S   0,0   0,0   0:02.76 kthreadd
    3 root       0 -20       0      0      0 I   0,0   0,0   0:00.00 rcu_gp
    4 root       0 -20       0      0      0 I   0,0   0,0   0:00.00 rcu_par_gp
    8 root       0 -20       0      0      0 I   0,0   0,0   0:00.00 mm_percpu_wq
    9 root      20   0       0      0      0 S   0,0   0,0   0:49.06 ksoftirqd/0
   10 root      20   0       0      0      0 I   0,0   0,0  18:24.20 rcu_sched
   11 root      20   0       0      0      0 I   0,0   0,0   0:00.00 rcu_bh
   12 root      rt   0       0      0      0 S   0,0   0,0   0:08.17 migration/0
   14 root      20   0       0      0      0 S   0,0   0,0   0:00.00 cpuhp/0
   15 root      20   0       0      0      0 S   0,0   0,0   0:00.00 cpuhp/1
   16 root      rt   0       0      0      0 S   0,0   0,0   0:11.79 migration/1
   17 root      20   0       0      0      0 S   0,0   0,0   0:26.01 ksoftirqd/1
```

   Quels sont les PID qui ont des priorités temps réel ?

   Les PIDs 12 et 16.

3. Considérons le listing `ps -el` ci-dessous :

```
F S   UID   PID  PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
4 S     0     1     0  0  80   0 - 42855 -      ?        00:09:59 systemd
1 S     0     2     0  0  80   0 -     0 -      ?        00:00:02 kthreadd
1 I     0     3     2  0  60 -20 -     0 -      ?        00:00:00 rcu_gp
```

```
1 S     0     9     2  0  80   0 -     0 -      ?        00:00:49 ksoftirqd/0
1 I     0    10     2  0  80   0 -     0 -      ?        00:18:26 rcu_sched
1 I     0    11     2  0  80   0 -     0 -      ?        00:00:00 rcu_bh
1 S     0    12     2  0 -40   - -     0 -      ?        00:00:08 migration/0
1 S     0    14     2  0  80   0 -     0 -      ?        00:00:00 cpuhp/0
5 S     0    15     2  0  80   0 -     0 -      ?        00:00:00 cpuhp/1
```

Quel PID a la plus grande priorité ?

Le PID 12.

4. Vous essayez de modifier la priorité d'un processus avec `renice` et vous vous retrouvez confronté à l'erreur suivante :

```
$ renice -10 21704
renice: failed to set priority for 21704 (process ID): Permission denied
```

Quelle est la cause probable de l'erreur ?

Sur l'utilisateur root peut réduire les valeurs *nice* en-dessous de zéro.

Réponses aux exercices d'approfondissement

1. Lorsqu'un processus occupe trop de temps CPU, il faut généralement modifier les priorités de ce processus. En utilisant `ps` avec les options standard pour afficher tous les processus du système dans un format long, quelle option `--sort` va trier les processus par utilisation du CPU, dans un ordre croissant ?

```
$ ps -el --sort=pcpu
```

2. La commande `schedtool` peut définir tous les paramètres d'ordonnancement du CPU dont Linux est capable ou afficher des informations pour des processus donnés. Comment peut-on l'utiliser pour afficher les paramètres d'ordonnancement du processus `1750` ? Par ailleurs, comment utiliser `schedtool` pour basculer le processus 1750 en temps réel avec une priorité de -90 (telle qu'affichée par `top`) ?

```
$ schedtool 1750
```

```
# schedtool -R -p 89 1750
```

103.7 Recherche dans des fichiers texte avec les expressions rationnelles

Référence aux objectifs de LPI

LPIC-1 v5, Exam 101, Objective 103.7

Valeur

3

Domaines de connaissance les plus importants

- Création d'expressions rationnelles simples contenant différents éléments de notation.
- Compréhension des différences entre les expressions rationnelles simples et étendues.
- Compréhension des concepts de caractères spéciaux, classes de caractères et ancres.
- Utilisation des expressions rationnelles dans des commandes pour effectuer des recherches dans une arborescence ou dans le contenu d'un fichier.
- Utilisation des expressions rationnelles pour supprimer, modifier et substituer du texte.

Liste partielle de termes, fichiers et utilitaires utilisés pour cet objectif

- `grep`
- `egrep`
- `fgrep`
- `sed`
- `regex(7)`

Linux Professional Institute

103.7 Leçon 1

Certification :	LPIC-1
Version :	5.0
Thème :	103 Commandes GNU et Unix
Objectif :	103.7 Recherche dans des fichiers texte avec les expressions régulières
Leçon :	1 sur 2

Introduction

Les algorithmes de recherche de chaînes de caractères sont couramment utilisés dans le traitement de données, si bien que les systèmes d'exploitation de type Unix disposent de leur propre implémentation généralisée : les *expressions régulières*, souvent abrégées en *regex*. Les expressions régulières sont des séquences de caractères qui constituent un motif générique utilisé pour localiser et parfois modifier une séquence correspondante dans une chaîne de caractères plus volumineuse. Les expressions régulières permettent notamment de :

- Écrire des règles d'analyse pour les requêtes dans les serveurs HTTP comme *nginx*.
- Écrire des scripts qui convertissent des jeux de données texte vers un autre format.
- Rechercher des occurrences pertinentes dans les fichiers de journalisation (*logs*) ou les documents.
- Filtrer les documents balisés en conservant le contenu sémantique.

L'expression régulière la plus simple contient au moins un *atome*. Un atome, ainsi nommé parce qu'il est l'élément de base d'une expression régulière, n'est qu'un caractère qui peut ou ne peut

pas avoir une signification particulière. La plupart des caractères ordinaires ne sont pas ambigus, ils conservent leur sens littéral, tandis que d'autres ont une signification particulière :

`.` (point)

L'atome correspond à n'importe quel caractère.

`^` (accent circonflexe)

L'atome correspond au début d'une ligne.

`$` (signe dollar)

L'atome correspond à la fin d'une ligne.

Par exemple, l'expression régulière `bc` composée des atomes littéraux `b` et `c` peut être trouvée dans la chaîne `abcd`, mais ne peut pas être trouvée dans la chaîne `a1cd`. D'un autre côté, l'expression régulière `.c` peut être trouvée dans les deux chaînes `abcd` et `a1cd`, étant donné que le point `.` correspond à n'importe quel caractère.

Les atomes `^` (accent circonflexe) et `$` (signe dollar) sont utilisés lorsque seules les correspondances au début ou à la fin de la chaîne de caractères sont pertinentes. C'est pour cela qu'on les appelle également *ancres*. Par exemple, `cd` peut être trouvé dans `abcd`, mais pas `^cd`. De même, `ab` peut être trouvé dans `abcd`, mais pas `ab$`. L'accent circonflexe `^` est un caractère littéral sauf au début, et le signe dollar `$` est un caractère littéral sauf à la fin de l'expression régulière.

L'expression entre crochets

Il existe un autre type d'atome appelé *expression entre crochets*. Même s'ils ne constituent pas un seul caractère, les crochets `[]` (avec leur contenu) sont considérés comme un seul atome. Une expression entre crochets n'est généralement qu'une liste de caractères littéraux entourés de `[]`, ce qui permet à l'atome de correspondre à n'importe quel caractère de la liste. Par exemple, l'expression `[1b]` peut être trouvée dans les deux chaînes `abcd` et `a1cd`. Pour spécifier les caractères auxquels l'atome ne doit pas correspondre, la liste doit commencer par `^`, comme dans `[^1b]`. Il est également possible de spécifier des plages de caractères dans des expressions entre crochets. Par exemple, `[0-9]` correspond aux chiffres de 0 à 9 et `[a-z]` à toute lettre minuscule. Les plages doivent être utilisées avec précaution, car elles peuvent ne pas être cohérentes selon les différents paramètres régionaux.

Les listes d'expressions entre crochets acceptent également des classes en remplacement des caractères simples et des plages de caractères. Voici les classes de caractères traditionnelles :

`[:alnum:]`

Représente un caractère alphanumérique.

`[:alpha:]`
Représente un caractère alphabétique.

`[:ascii:]`
Représente un caractère qui fait partie du jeu de caractères ASCII.

`[:blank:]`
Représente un caractère d'espacement, c'est-à-dire une espace ou une tabulation.

`[:cntrl:]`
Représente un caractère de contrôle.

`[:digit:]`
Représente un chiffre (0 à 9).

`[:graph:]`
Représente n'importe quel caractère imprimable sauf l'espace.

`[:lower:]`
Représente un caractère minuscule.

`[:print:]`
Représente n'importe quel caractère imprimable y compris l'espace.

`[:punct:]`
Représente n'importe quel caractère imprimable qui n'est pas une espace ou un caractère alphanumérique.

`[:space:]`
Représente les caractères d'espacement : espace, saut de page (`\f`), saut de ligne (`\n`), retour chariot (`\r`), tabulation horizontale (`\t`) et tabulation verticale (`\v`).

`[:upper:]`
Représente une lettre majuscule.

`[:xdigit:]`
Représente les chiffres hexadécimaux (0 à F).

Les classes de caractères peuvent être combinées avec des caractères simples et des plages. En revanche, elles ne peuvent pas être utilisées comme élément final d'une plage. Par ailleurs, les classes de caractères ne peuvent être utilisées que dans des expressions entre crochets, et non en

tant qu'atome indépendant en dehors des crochets.

Les quantificateurs

La portée d'un atome, qu'il s'agisse d'un atome à un seul caractère ou d'un atome entre crochets, peut être ajustée par le biais d'un *quantificateur atomique*. Les quantificateurs atomiques définissent des *séquences* d'atomes, c'est-à-dire que les correspondances ont lieu lorsqu'une répétition contiguë de l'atome a été trouvée dans la chaîne de caractères. La sous-chaîne de caractères associée à la correspondance est appelée une *pièce*. Cela dit, les quantificateurs et d'autres fonctionnalités des expressions régulières sont traités différemment en fonction de la norme utilisée.

D'après la norme POSIX, il existe deux formes d'expressions régulières : les expressions régulières "basiques" et "étendues". La plupart des programmes de traitement de texte installés dans une distribution Linux conventionnelle prennent en charge les deux formes. Il est donc important de connaître leurs différences afin d'éviter les problèmes de compatibilité et de choisir l'implémentation la plus adaptée à la tâche envisagée.

Le quantificateur `*` a la même fonction dans les expressions régulières basiques et étendues (l'atome apparaît zéro fois ou plus) et c'est un caractère littéral s'il apparaît au début de l'expression régulière ou s'il est précédé d'une barre oblique inverse `\`. Le quantificateur plus `` `sélectionnera les éléments contenant une ou plusieurs correspondances d'atomes dans l'ordre. Avec le quantificateur point d'interrogation `?`, une correspondance sera obtenue si l'atome correspondant apparaît une fois ou pas du tout. S'ils sont précédés d'une barre oblique inverse `\`, leur signification particulière n'est pas prise en compte. Les expressions régulières basiques supportent également les quantificateurs ` `` et `?`, mais ils doivent être précédés d'une barre oblique inverse. Contrairement aux expressions régulières étendues, `+` et `?` sont des caractères littéraux dans les expressions régulières basiques.

Les limites

Une *limite* est un quantificateur atomique qui, comme son nom l'indique, permet à l'utilisateur de spécifier des limites quantitatives précises pour un atome. Dans les expressions régulières étendues, une limite peut apparaître sous trois formes :

`{i}`

L'atome doit apparaître exactement `i` fois (`i` étant un nombre entier). Par exemple, `[[:blank:]]{2}` correspond exactement à deux caractères vides.

`{i,}`

L'atome doit apparaître au moins `i` fois (`i` étant un nombre entier). Par exemple, `[[:blank:]]{2,}` correspond à toute séquence d'au moins deux caractères vides.

`{i,j}`

L'atome doit apparaître au minimum `i` et au maximum `j` fois (`i` et `j` étant des nombres entiers, avec `j` supérieur à `i`). Par exemple, `xyz{2,4}` correspond à la chaîne `xy` suivie de deux à quatre fois le caractère `z`.

Dans tous les cas, si une sous-chaîne correspond à une expression régulière et qu'une sous-chaîne plus longue et qui commence au même endroit correspond également, la sous-chaîne la plus longue sera prise en compte.

Les expressions régulières basiques autorisent également les limites, mais les délimiteurs doivent être précédés de `\` : `\{` et `\}`. En eux-mêmes, `{` et `}` sont interprétés comme des caractères littéraux. Un `\{` suivi d'un caractère autre qu'un chiffre est un caractère littéral, et non pas le début d'une délimitation.

Les branches et les renvois

Les expressions régulières basiques se distinguent également des expressions régulières étendues par un autre aspect important : une expression régulière étendue peut être subdivisée en *branches*, chacune d'entre elles étant une expression régulière indépendante. Les branches sont séparées par des `|` et l'expression régulière combinée s'appliquera à tout ce qui correspond à l'une des branches. Par exemple, `he|him` s'appliquera si les sous-chaînes `he` ou `him` figurent dans la chaîne de caractères analysée. Les expressions régulières basiques interprètent `|` comme un caractère littéral. En revanche, la plupart des programmes prenant en charge les expressions régulières basiques autorisent les branches avec `\|`.

Une expression régulière étendue entourée de `()` peut être utilisée dans un *renvoi* vers une référence. Par exemple, `([[:digit:]])\1` correspondra à toute expression régulière qui se répète au moins une fois, parce que le `\1` dans l'expression est le renvoi vers la pièce correspondant à la première sous-expression entre parenthèses. Si plusieurs sous-expressions entre parenthèses existent dans l'expression régulière, elles pourront être référencées par `\2`, `\3` et ainsi de suite.

Pour les regex basiques, les sous-expressions doivent être entourées de `\(` et `\)`, les `(` et `)` étant eux-mêmes des caractères ordinaires. L'indicateur de renvoi est utilisé comme dans les expressions régulières étendues.

Recherche à l'aide d'expressions régulières

L'avantage le plus immédiat des expressions régulières consiste à améliorer les recherches dans les systèmes de fichiers et dans les documents texte. L'option `-regex` de la commande `find` permet de tester chaque chemin dans une hiérarchie de répertoires en fonction d'une expression régulière. Par exemple,

```
$ find $HOME -regex '.*/\..*' -size +100M
```

recherche les fichiers de plus de 100 mégaoctets (100 unités de 1048576 octets), mais uniquement dans les chemins du répertoire personnel de l'utilisateur qui contiennent une correspondance avec `.*/\..*`, c'est-à-dire un `/.` entouré de n'importe quel autre nombre de caractères. En d'autres termes, seuls les fichiers cachés ou les fichiers situés dans des répertoires cachés seront affichés, quelle que soit la position de `/.` dans le chemin d'accès correspondant. Pour des expressions régulières insensibles à la casse, l'option `-iregex` devra être utilisée à la place :

```
$ find /usr/share/fonts -regextype posix-extended -iregex
'.*(dejavu|liberation).*sans.*(italic|oblique).*'
/usr/share/fonts/dejavu/DejaVuSansCondensed-BoldOblique.ttf
/usr/share/fonts/dejavu/DejaVuSansCondensed-Oblique.ttf
/usr/share/fonts/dejavu/DejaVuSans-BoldOblique.ttf
/usr/share/fonts/dejavu/DejaVuSans-Oblique.ttf
/usr/share/fonts/dejavu/DejaVuSansMono-BoldOblique.ttf
/usr/share/fonts/dejavu/DejaVuSansMono-Oblique.ttf
/usr/share/fonts/liberation/LiberationSans-BoldItalic.ttf
/usr/share/fonts/liberation/LiberationSans-Italic.ttf
```

Dans cet exemple, l'expression régulière contient des branches (écrites dans le style étendu) pour afficher certains fichiers de polices dans l'arborescence `/usr/share/fonts`. Les expressions régulières étendues ne sont pas supportées par défaut, mais `find` permet de les activer avec `-regextype posix-extended` ou `-regextype egrep`. Le standard regex par défaut pour `find` est *findutils-default*, qui est virtuellement un clone d'expression régulière basique.

Il est souvent nécessaire de transmettre le résultat d'un programme à la commande `less` lorsqu'il ne tient pas sur l'écran. La commande `less` fractionne son entrée en pages, une page d'écran à la fois, ce qui permet à l'utilisateur de naviguer facilement dans le texte vers le haut et vers le bas. De plus, `less` permet à l'utilisateur d'effectuer des recherches basées sur les expressions régulières. Cette fonctionnalité est particulièrement importante car `less` est le paginateur par défaut utilisé pour de nombreuses tâches quotidiennes, comme l'inspection d'entrées de journal ou la consultation de pages de manuel. Lors de la lecture d'une page de manuel, par exemple, la

touche `/` ouvre une invite de recherche. Il s'agit d'un scénario typique dans lequel les expressions régulières sont utiles, car les options de commande sont affichées juste après une marge dans la mise en page générale du manuel. Cependant, la même option peut apparaître plusieurs fois dans le texte, ce qui rend les recherches littérales impossibles. Quoi qu'il en soit, en tapant `^[[:blank:]]*-o` — ou plus simplement : `^ *-o` — à l'invite de recherche, on passe immédiatement à la section de l'option `-o` (si elle existe) après avoir appuyé sur Entrée, ce qui permet de consulter plus rapidement la description d'une option.

Exercices guidés

1. Quelle expression régulière étendue correspondrait à n'importe quelle adresse électronique, comme `info@example.org` ?

2. Quelle expression régulière étendue correspondrait uniquement à une adresse IPv4 au format standard de type quartet pointillé, comme `192.168.15.1` ?

3. Comment la commande `grep` peut-elle être utilisée pour lister le contenu du fichier `/etc/services`, en ignorant tous les commentaires (lignes commençant par #) ?

4. Le fichier `domains.txt` contient une liste de noms de domaines, un par ligne. Comment utiliser la commande `egrep` pour lister les seuls domaines `.org` ou `.com` ?

Exercices d'approfondissement

1. Depuis le répertoire courant, comment la commande `find` utiliserait-elle une expression régulière étendue pour rechercher tous les fichiers qui ne contiennent pas un suffixe de fichier standard (noms de fichiers ne se terminant pas par `.txt` ou `.c`, par exemple) ?

2. La commande `less` est le programme de pagination par défaut pour l'affichage de fichiers texte longs dans le shell. En tapant `/`, une expression régulière peut être saisie dans l'invite de recherche pour aller au premier résultat correspondant. Quelle combinaison de touches doit être saisie à l'invite de recherche pour rester dans la position actuelle du document et seulement mettre en surbrillance les correspondances ?

3. Dans `less`, comment serait-il possible de filtrer le résultat de sorte que seules les lignes qui correspondent à une expression régulière s'affichent ?

Résumé

Cette leçon aborde la prise en charge générale par Linux des expressions régulières, une norme largement utilisée dont les fonctionnalités de recherche de motifs sont supportées par la plupart des programmes de traitement de texte. La leçon comprend les étapes suivantes :

- Qu'est-ce qu'une expression régulière.
- Les principaux composants d'une expression régulière.
- Les différences entre les expressions régulières basiques et étendues.
- Comment effectuer des recherches de texte et de fichiers simples à l'aide des expressions régulières.

Réponses aux exercices guidés

1. Quelle expression régulière étendue correspondrait à n'importe quelle adresse électronique, comme `info@example.org` ?

```
egrep "\S+@\S+\.\S+"
```

2. Quelle expression régulière étendue correspondrait uniquement à une adresse IPv4 au format standard de type quartet pointillé, comme `192.168.15.1` ?

```
egrep "[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}"
```

3. Comment la commande `grep` peut-elle être utilisée pour lister le contenu du fichier `/etc/services`, en ignorant tous les commentaires (lignes commençant par #) ?

```
grep -v ^# /etc/services
```

4. Le fichier `domains.txt` contient une liste de noms de domaines, un par ligne. Comment utiliser la commande `egrep` pour lister les seuls domaines `.org` ou `.com` ?

```
egrep ".org$|.com$" domains.txt
```

Réponses aux exercices d'approfondissement

1. Depuis le répertoire courant, comment la commande `find` utiliserait-elle une expression régulière étendue pour rechercher tous les fichiers qui ne contiennent pas un suffixe de fichier standard (noms de fichiers ne se terminant pas par `.txt` ou `.c`, par exemple) ?

   ```
   find . -type f -regextype egrep -not -regex '.*\.[[:alnum:]]{1,}$'
   ```

2. La commande `less` est le programme de pagination par défaut pour l'affichage de fichiers texte longs dans le shell. En tapant `/`, une expression régulière peut être saisie dans l'invite de recherche pour aller au premier résultat correspondant. Quelle combinaison de touches doit être saisie à l'invite de recherche pour rester dans la position actuelle du document et seulement mettre en surbrillance les correspondances ?

   Appuyer sur `Ctrl` + `K` avant de saisir le terme de la recherche.

3. Dans `less`, comment serait-il possible de filtrer le résultat de sorte que seules les lignes qui correspondent à une expression régulière s'affichent ?

   En appuyant sur `&` et en saisissant l'expression recherchée.

Linux Professional Institute

103.7 Leçon 2

Certification :	LPIC-1
Version :	5.0
Thème :	103 Commandes GNU et Unix
Objectif :	103.7 Recherche dans des fichiers texte avec les expressions régulières
Leçon :	2 sur 2

Introduction

La transmission de données par une chaîne de commandes groupées permet d'appliquer des filtres complexes basés sur les expressions régulières. Les expressions régulières sont une technique importante utilisée non seulement dans l'administration système, mais aussi dans l'exploration de données (*data mining*) et les domaines connexes. Deux commandes sont particulièrement adaptées à la manipulation de fichiers et de données textuelles à l'aide d'expressions régulières : `grep` et `sed`. `grep` est un outil de recherche de motifs et `sed` est un éditeur de flux. Ces commandes sont utiles en elles-mêmes, mais c'est surtout en travaillant avec d'autres processus qu'elles se distinguent.

La recherche de motifs : grep

L'une des utilisations les plus courantes de `grep` consiste à faciliter l'inspection de fichiers longs, en utilisant l'expression régulière comme filtre appliqué à chaque ligne. On peut l'utiliser pour n'afficher que les lignes commençant par un certain terme. Par exemple, `grep` peut être utilisé pour examiner un fichier de configuration pour les modules du noyau, en n'affichant que les lignes relatives aux options :

```
$ grep '^options' /etc/modprobe.d/alsa-base.conf
options snd-pcsp index=-2
options snd-usb-audio index=-2
options bt87x index=-2
options cx88_alsa index=-2
options snd-atiixp-modem index=-2
options snd-intel8x0m index=-2
options snd-via82xx-modem index=-2
```

Le symbole *pipe* `|` peut être utilisé pour rediriger la sortie d'une commande directement vers l'entrée de `grep`. L'exemple suivant utilise une expression régulière entre crochets pour sélectionner les lignes de la sortie de `fdisk -l` qui commencent par `Disk /dev/sda` ou `Disk /dev/sdb` :

```
# fdisk -l | grep '^Disk /dev/sd[ab]'
Disk /dev/sda: 320.1 GB, 320072933376 bytes, 625142448 sectors
Disk /dev/sdb: 7998 MB, 7998537728 bytes, 15622144 sectors
```

La simple sélection des lignes avec des correspondances peut être inadaptée à une tâche particulière, ce qui nécessite d'ajuster le comportement de `grep` par le biais de ses options. Par exemple, l'option `-c` ou `--count` demande à `grep` d'afficher le nombre de lignes avec des correspondances :

```
# fdisk -l | grep '^Disk /dev/sd[ab]' -c
2
```

L'option peut être placée avant ou après l'expression régulière. Voici d'autres options importantes de `grep` :

`-c` ou `--count`

Au lieu d'afficher les résultats de la recherche, afficher uniquement le nombre total de correspondances dans un fichier donné.

`-i` ou `--ignore-case`

Rendre la recherche insensible à la casse.

`-f FILE` ou `--file=FILE`

Indiquer un fichier qui contient l'expression régulière à utiliser.

`-n` ou `--line-number`

Indiquer le numéro de la ligne.

`-v` ou `--invert-match`

Sélectionner toutes les lignes sauf celles qui contiennent des correspondances.

`-H` ou `--with-filename`

Afficher également le nom du fichier qui contient la ligne.

`-z` ou `--null-data`

Plutôt que de demander à `grep` de traiter les flux de données d'entrée et de sortie comme des lignes séparées (en utilisant le retour à la ligne par défaut), il faut considérer l'entrée ou la sortie comme une séquence de lignes. Lorsque l'on combine la sortie de la commande `find` en utilisant son option `-print0` avec la commande `grep`, l'option `-z` ou `--null-data` devra être utilisée pour traiter le flux de la même manière.

Bien qu'activée par défaut lorsque plusieurs chemins de fichiers sont fournis en entrée, l'option `-H` n'est pas activée pour les fichiers individuels. Cela peut s'avérer critique dans certaines situations, comme lorsque `grep` est appelé directement par `find`, par exemple :

```
$ find /usr/share/doc -type f -exec grep -i '3d modeling' "{}" \; | cut -c -100
artistic aspects of 3D modeling. Thus this might be the application you are
This major approach of 3D modeling has not been supported
oce is a C++ 3D modeling library. It can be used to develop CAD/CAM softwares, for instance
[FreeCad
```

Dans cet exemple, `find` affiche tous les fichiers sous `/usr/share/doc` et transmet chacun à `grep`, qui effectue à son tour une recherche insensible à la casse du terme `3d modeling` à l'intérieur de chaque fichier. Le transfert de données vers `cut` sert uniquement à limiter la longueur de la sortie à 100 colonnes. Notez cependant qu'il n'y a aucun moyen de savoir de quel fichier proviennent les lignes en question. Ce problème est résolu en ajoutant l'option `-H` à `grep` :

```
$ find /usr/share/doc -type f -exec grep -i -H '3d modeling' "{}" \; | cut -c -100
/usr/share/doc/openscad/README.md:artistic aspects of 3D modeling. Thus this might be the
applicatio
/usr/share/doc/opencsg/doc/publications.html:This major approach of 3D modeling has not been
support
```

À présent, on peut identifier les fichiers dans lesquels chaque correspondance a été trouvée. Pour rendre la liste encore plus parlante, les lignes qui précèdent et qui suivent peuvent être ajoutées

aux lignes contenant des correspondances :

```
$ find /usr/share/doc -type f -exec grep -i -H -1 '3d modeling' "{}" \; | cut -c -100
/usr/share/doc/openscad/README.md-application Blender), OpenSCAD focuses on the CAD aspects
rather t
/usr/share/doc/openscad/README.md:artistic aspects of 3D modeling. Thus this might be the
applicatio
/usr/share/doc/openscad/README.md-looking for when you are planning to create 3D models of
machine p
/usr/share/doc/opencsg/doc/publications.html-3D graphics library for Constructive Solid
Geometry (CS
/usr/share/doc/opencsg/doc/publications.html:This major approach of 3D modeling has not been
support
/usr/share/doc/opencsg/doc/publications.html-by real-time computer graphics until recently.
```

L'option `-1` demande à `grep` d'inclure une ligne avant et une ligne après lorsqu'il trouve une ligne avec une correspondance. Ces lignes supplémentaires sont appelées *lignes contextuelles*. Elles sont identifiées dans la sortie par un signe moins après le nom du fichier. Le même résultat peut être obtenu avec `-C 1` ou `--context=1`. D'autres quantités de lignes de contexte peuvent être indiquées.

Il existe deux programmes complémentaires à `grep` : `egrep` et `fgrep`. Le programme `egrep` est équivalent à la commande `grep -E`, qui incorpore des fonctionnalités supplémentaires autres que les expressions régulières de base. Par exemple, avec `egrep` il est possible d'utiliser des expressions régulières étendues, comme les branches :

```
$ find /usr/share/doc -type f -exec egrep -i -H -1 '3d (modeling|printing)' "{}" \; | cut -c
-100
/usr/share/doc/openscad/README.md-application Blender), OpenSCAD focuses on the CAD aspects
rather t
/usr/share/doc/openscad/README.md:artistic aspects of 3D modeling. Thus this might be the
applicatio
/usr/share/doc/openscad/README.md-looking for when you are planning to create 3D models of
machine p
/usr/share/doc/openscad/RELEASE_NOTES.md-* Support for using 3D-Mouse / Joystick / Gamepad
input dev
/usr/share/doc/openscad/RELEASE_NOTES.md:* 3D Printing support: Purchase from a print
service partne
/usr/share/doc/openscad/RELEASE_NOTES.md-* New export file formats: SVG, 3MF, AMF
/usr/share/doc/opencsg/doc/publications.html-3D graphics library for Constructive Solid
Geometry (CS
/usr/share/doc/opencsg/doc/publications.html:This major approach of 3D modeling has not been
```

```
support
/usr/share/doc/opencsg/doc/publications.html-by real-time computer graphics until recently.
```

Dans cet exemple, soit `3D modeling` soit `3D printing` correspondra à l'expression, sans tenir compte des majuscules et des minuscules. Pour afficher les seules parties d'un flux de texte qui correspondent à l'expression utilisée par `egrep`, utilisez l'option `-o`.

Le programme `fgrep` est équivalent à `grep -F`, c'est-à-dire qu'il ne tient pas compte des expressions régulières. Il est utile dans les recherches simples où le but est de trouver une expression littérale. Dans ce cas, les caractères spéciaux tels que le signe du dollar et le point seront considérés au pied de la lettre et non pas en fonction de leur signification dans une expression régulière.

L'éditeur de flux : sed

Le programme `sed` sert à modifier des données textuelles de façon non interactive. Cela signifie que tout le travail d'édition est effectué à partir d'un jeu d'instructions prédéfinies, et non pas en tapant arbitrairement et directement dans un texte affiché à l'écran. En termes modernes, `sed` peut être compris comme un analyseur de motifs : lorsqu'un texte est fourni en entrée, il insère un contenu spécifique à des positions prédéfinies ou lorsqu'il trouve une correspondance pour une expression régulière.

Sed, comme son nom le suggère, est parfaitement adapté à l'édition de texte à travers des flux de données (*pipelines*). Sa syntaxe de base est `sed -f SCRIPT` lorsque les instructions d'édition sont stockées dans le fichier `SCRIPT` ou `sed -e COMMANDS` pour exécuter `COMMANDS` directement depuis la ligne de commande. En l'absence de `-f` ou de `-e`, `sed` utilise le premier paramètre non optionnel comme fichier de script. Il est également possible d'utiliser un fichier en entrée en fournissant son chemin comme argument à `sed`.

Les instructions `sed` sont composées d'un seul caractère, qui peut être précédé d'une adresse ou suivi d'une ou plusieurs options. Elles sont appliquées successivement à chaque ligne. Les adresses peuvent être un numéro de ligne unique, une expression régulière ou une plage de lignes. Par exemple, la première ligne d'un flux de texte peut être supprimée avec `1d`, où `1` spécifie la ligne où la commande de suppression `d` sera appliquée. Pour illustrer l'utilisation de `sed`, prenons le résultat de la commande `factor `seq 12``, qui renvoie les facteurs premiers des nombres de 1 à 12 :

```
$ factor `seq 12`
1:
2: 2
3: 3
```

```
4: 2 2
5: 5
6: 2 3
7: 7
8: 2 2 2
9: 3 3
10: 2 5
11: 11
12: 2 2 3
```

La suppression de la première ligne avec `sed` est effectuée par `1d` :

```
$ factor `seq 12` | sed 1d
2: 2
3: 3
4: 2 2
5: 5
6: 2 3
7: 7
8: 2 2 2
9: 3 3
10: 2 5
11: 11
12: 2 2 3
```

Une série de lignes peut être spécifiée avec une virgule de séparation :

```
$ factor `seq 12` | sed 1,7d
8: 2 2 2
9: 3 3
10: 2 5
11: 11
12: 2 2 3
```

On peut utiliser plus d'une instruction dans la même exécution, en les séparant par des points-virgules. Cependant, il est important dans ce cas de les entourer de guillemets pour éviter que le point-virgule ne soit interprété par le shell :

```
$ factor `seq 12` | sed "1,7d;11d"
8: 2 2 2
9: 3 3
```

```
10: 2 5
12: 2 2 3
```

Dans cet exemple, deux instructions de suppression ont été exécutées, d'abord pour les lignes allant de 1 à 7, puis pour la ligne 11. Une adresse peut également être une expression régulière, de sorte que seules les lignes qui correspondent à cette expression seront affectées par l'instruction :

```
$ factor `seq 12` | sed "1d;/:.*2.*/d"
3: 3
5: 5
7: 7
9: 3 3
11: 11
```

L'expression régulière `:.*2.*` correspond à toute occurrence du nombre 2 n'importe où après un deux-points, ce qui entraîne la suppression des lignes correspondant à des nombres avec le facteur 2. Avec `sed`, tout ce qui est placé entre les barres obliques (`/`) est considéré comme une expression régulière et, par défaut, toutes les expressions régulières de base sont supportées. Par exemple, `sed -e "/^#/d" /etc/services` affiche le contenu du fichier `/etc/services` sans les lignes qui commencent par `#` (lignes de commentaires).

L'instruction de suppression `d` fait partie de la multitude d'instructions d'édition fournies par `sed`. Au lieu de supprimer une ligne, `sed` peut la remplacer par un texte donné :

```
$ factor `seq 12` | sed "1d;/:.*2.*/c REMOVED"
REMOVED
3: 3
REMOVED
5: 5
REMOVED
7: 7
REMOVED
9: 3 3
REMOVED
11: 11
REMOVED
```

L'instruction `c REMOVED` remplace simplement une ligne par le texte `REMOVED`. Dans notre exemple, chaque ligne dont une partie correspond à l'expression régulière `:.*2.*` est affectée par l'instruction `c REMOVED`. L'instruction `a TEXT` copie le texte indiqué par `TEXT` sur une nouvelle ligne après la ligne avec une correspondance. L'instruction `r FILE` fait la même chose, mais en

copiant le contenu du fichier indiqué par `FILE`. L'instruction `w` fait le contraire de `r`, c'est-à-dire que la ligne sera ajoutée au fichier indiqué.

L'instruction `sed` la plus populaire est certainement `s/FIND/REPLACE/`. Elle sert à remplacer une correspondance à l'expression régulière `FIND` par le texte indiqué par `REPLACE`. Par exemple, l'instruction `s/hda/sda/` remplace une chaîne de caractères correspondant à l'expression régulière `hda` par `sda`. Seule la première correspondance trouvée dans la ligne sera remplacée, à moins que l'indicateur `g` ne soit placé après l'instruction, comme dans `s/hda/sda/g`.

Une étude de cas plus concrète permettra d'illustrer les fonctionnalités de `sed`. Admettons qu'une clinique veuille envoyer des textos à ses clients pour leur rappeler leur rendez-vous du lendemain. Un scénario typique s'appuie sur un service professionnel de messagerie instantanée, qui fournit une API permettant d'accéder au système chargé de transmettre les messages. Ces messages proviennent généralement du même système que celui qui exécute l'application contrôlant les rendez-vous des clients, et ils sont déclenchés à une heure précise de la journée ou à la suite d'un autre événement. Dans cette situation hypothétique, l'application pourrait générer un fichier appelé `appointments.csv` contenant des données sous forme de tableau avec tous les rendez-vous pour le lendemain, puis utilisé par `sed` pour générer les messages texte à partir d'un fichier modèle appelé `template.txt`. Les fichiers CSV sont un outil standard pour exporter des données à partir d'une base de données, de sorte que des exemples de rendez-vous peuvent être présentés comme suit :

```
$ cat appointments.csv
"NAME","TIME","PHONE"
"Carol","11am","55557777"
"Dave","2pm","33334444"
```

La première ligne contient les intitulés de chaque colonne, qui serviront à faire correspondre les balises du fichier modèle :

```
$ cat template.txt
Hey <NAME>, don't forget your appointment tomorrow at <TIME>.
```

Les signes inférieur à `<` et supérieur à `>` ont été placés autour des titres pour les identifier comme des balises. Le script Bash ci-dessous analyse tous les rendez-vous en file d'attente en utilisant `template.txt` comme modèle de message :

```
#! /bin/bash

TEMPLATE=`cat template.txt`
```

```
TAGS=(`sed -ne '1s/^"//;1s/","/\n/g;1s/"$//p' appointments.csv`)
mapfile -t -s 1 ROWS < appointments.csv
for (( r = 0; r < ${#ROWS[*]}; r++ ))
do
  MSG=$TEMPLATE
  VALS=(`sed -e 's/^"//;s/","/\n/g;s/"$//' <<<${ROWS[$r]}`)
  for (( c = 0; c < ${#TAGS[*]}; c++ ))
  do
    MSG=`sed -e "s/<${TAGS[$c]}>/${VALS[$c]}/g" <<<"$MSG"`
  done
  echo curl --data message=\"$MSG\" --data phone=\"${VALS[2]}\" https://mysmsprovider/api
done
```

Un véritable script de production devrait également gérer l'authentification, la vérification des erreurs et la journalisation, mais l'exemple comporte des fonctionnalités de base pour le moment. Les premières instructions exécutées par `sed` ne sont appliquées qu'à la première ligne — l'adresse `1` dans `1s/^"//;1s/","/\n/g;1s/"$//p` — pour supprimer les guillemets de début et de fin — `1s/^"//` et `1s/"$//` — et pour remplacer les séparateurs de champs par un caractère de retour à la ligne : `1s/","/\n/g`. Seule la première ligne est nécessaire pour charger les noms des colonnes, et les lignes non correspondantes seront supprimées par l'option `-n`, ce qui nécessite que l'indicateur `p` soit placé après la dernière commande `sed` pour afficher la ligne correspondante. Les balises sont alors stockées dans la variable `TAGS` sous forme de matrice Bash. Une autre matrice Bash est créée par la commande `mapfile` pour stocker les lignes qui contiennent les rendez-vous dans la matrice `ROWS`.

Une boucle `for` est mise en place pour traiter chaque ligne de rendez-vous trouvée dans `ROWS`. Ensuite, les guillemets et les séparateurs dans le rendez-vous - le rendez-vous figure dans la variable `${ROWS[$r]}` utilisée comme un *here string* - sont remplacés par `sed`, de la même manière que les commandes utilisées pour charger les balises. Les valeurs séparées du rendez-vous sont alors stockées dans la matrice `VALS`, où les indices 0, 1 et 2 correspondent aux valeurs de `NAME`, `TIME` et `PHONE`.

Enfin, une boucle `for` imbriquée traverse la matrice `TAGS` et remplace chaque balise trouvée dans le modèle par sa valeur correspondante dans `VALS`. La variable `MSG` contient une copie du modèle affiché, mise à jour par la commande de substitution `s/<${TAGS[$c]}>/${VALS[$c]}/g` à chaque passage de la boucle dans `TAGS`.

Il en résulte un message du type : `"Hey Carol, don't forget your appointment tomorrow at 11am."` Le message généré peut alors être envoyé sous forme de paramètre dans une requête HTTP avec `curl`, en tant que message e-mail ou n'importe quelle autre méthode similaire.

Combiner grep et sed

Les commandes `grep` et `sed` peuvent être utilisées conjointement lorsque des procédures d'exploration de texte plus complexes s'avèrent nécessaires. En tant qu'administrateur système, vous pouvez être amené à inspecter toutes les tentatives de connexion à un serveur, par exemple. Le fichier `/var/log/wtmp` enregistre toutes les connexions et déconnexions, tandis que le fichier `/var/log/btmp` enregistre les tentatives de connexion échouées. Ces données sont écrites dans un format binaire, qui peut être lu respectivement avec les commandes `last` et `lastb`.

La sortie de `lastb` affiche non seulement le nom d'utilisateur utilisé dans la tentative de connexion échouée, mais aussi son adresse IP :

```
# lastb -d -a -n 10 --time-format notime
user     ssh:notty       (00:00)     81.161.63.251
nrostagn ssh:notty       (00:00)     vmd60532.contaboserver.net
pi       ssh:notty       (00:00)     132.red-88-20-39.staticip.rima-tde.net
pi       ssh:notty       (00:00)     132.red-88-20-39.staticip.rima-tde.net
pi       ssh:notty       (00:00)     46.6.11.56
pi       ssh:notty       (00:00)     46.6.11.56
nps      ssh:notty       (00:00)     vmd60532.contaboserver.net
narmadan ssh:notty       (00:00)     vmd60532.contaboserver.net
nominati ssh:notty       (00:00)     vmd60532.contaboserver.net
nominati ssh:notty       (00:00)     vmd60532.contaboserver.net
```

L'option `-d` traduit l'adresse IP en nom d'hôte correspondant. Le nom d'hôte peut fournir des indices sur le fournisseur d'accès ou le service d'hébergement utilisé pour effectuer ces tentatives de connexion douteuses. L'option `-a` affiche le nom d'hôte dans la dernière colonne, ce qui facilite le filtrage à appliquer. L'option `--time-format notime` supprime l'heure à laquelle la tentative de connexion a eu lieu. La commande `lastb` peut mettre un certain temps à se terminer s'il y a eu trop de tentatives de connexion erronées, c'est pourquoi l'affichage a été limité à dix entrées avec l'option `-n 10`.

Toutes les adresses IP distantes ne sont pas associées à un nom d'hôte, de sorte que le reverse DNS ne les concerne pas et qu'elles peuvent être ignorées. Même s'il est possible de rédiger une expression régulière qui correspondrait au format attendu pour un nom d'hôte en fin de ligne, il est probablement plus simple d'écrire une regex qui correspond soit à une lettre de l'alphabet, soit à un seul chiffre en fin de ligne. L'exemple ci-dessous montre comment la commande `grep` récupère la liste sur l'entrée standard et supprime les lignes qui ne contiennent pas de nom d'hôte :

```
# lastb -d -a --time-format notime | grep -v '[0-9]$' | head -n 10
```

```
nvidia   ssh:notty       (00:00)     vmd60532.contaboserver.net
n_tonson ssh:notty       (00:00)     vmd60532.contaboserver.net
nrostagn ssh:notty       (00:00)     vmd60532.contaboserver.net
pi       ssh:notty       (00:00)     132.red-88-20-39.staticip.rima-tde.net
pi       ssh:notty       (00:00)     132.red-88-20-39.staticip.rima-tde.net
nps      ssh:notty       (00:00)     vmd60532.contaboserver.net
narmadan ssh:notty       (00:00)     vmd60532.contaboserver.net
nominati ssh:notty       (00:00)     vmd60532.contaboserver.net
nominati ssh:notty       (00:00)     vmd60532.contaboserver.net
nominati ssh:notty       (00:00)     vmd60532.contaboserver.net
```

La commande `grep` avec l'option `-v` n'affiche que les lignes qui ne correspondent pas à l'expression régulière donnée. Une expression régulière correspondant à toute ligne se terminant par un chiffre (c'est-à-dire `[0-9]$`) ne retiendra que les entrées sans nom d'hôte. Par conséquent, `grep -v '[0-9]$'` n'affichera que les lignes qui se terminent par un nom d'hôte.

Le résultat peut être filtré encore davantage, en ne gardant que le nom de domaine et en supprimant les autres parties de chaque ligne. La commande `sed` peut le faire avec une commande de substitution pour remplacer la ligne entière par une référence au nom de domaine :

```
# lastb -d -a --time-format notime | grep -v '[0-9]$' | sed -e 's/.* \(.*\)$/\1/' | head -n
10
vmd60532.contaboserver.net
vmd60532.contaboserver.net
vmd60532.contaboserver.net
132.red-88-20-39.staticip.rima-tde.net
132.red-88-20-39.staticip.rima-tde.net
vmd60532.contaboserver.net
vmd60532.contaboserver.net
vmd60532.contaboserver.net
vmd60532.contaboserver.net
vmd60532.contaboserver.net
```

La parenthèse échappée dans `.* \(.*\)$` indique à `sed` de se souvenir de cette partie de la ligne, c'est-à-dire la partie entre le dernier caractère d'espacement et la fin de la ligne. Dans l'exemple, cette partie est référencée avec `\1` et utilisée pour remplacer la ligne entière.

Il est clair que la plupart des hôtes distants essaient de se connecter plus d'une fois, ce qui fait que le même nom de domaine se répète. Pour supprimer les entrées récurrentes, il faut d'abord les trier (avec la commande `sort`) puis les passer à la commande `uniq` :

```
# lastb -d -a --time-format notime | grep -v '[0-9]$' | sed -e 's/.* \(.*\)$/\1/' | sort |
uniq | head -n 10
116-25-254-113-on-nets.com
132.red-88-20-39.staticip.rima-tde.net
145-40-33-205.power-speed.at
tor.laquadrature.net
tor.momx.site
ua-83-226-233-154.bbcust.telenor.se
vmd38161.contaboserver.net
vmd60532.contaboserver.net
vmi488063.contaboserver.net
vmi515749.contaboserver.net
```

On voit ainsi comment il est possible de combiner une série de commandes pour obtenir le résultat voulu. La liste des noms d'hôtes pourra ensuite être utilisée pour rédiger des règles de pare-feu ou pour prendre d'autres mesures destinées à renforcer la sécurité du serveur.

Exercices guidés

1. La commande `last` affiche une liste des derniers utilisateurs connectés, y compris leurs adresses IP d'origine. Comment pourrait-on utiliser la commande `egrep` pour filtrer les résultats de `last`, en affichant uniquement les occurrences d'une adresse IPv4, et en supprimant toute information supplémentaire dans la ligne correspondante ?

2. Quelle option doit être utilisée avec `grep` pour filtrer correctement la sortie générée par la commande `find` exécutée avec l'option `-print0` ?

3. La commande `uptime -s` affiche la dernière date à laquelle le système a été démarré, comme dans *2019-08-05 20:13:22*. Quel sera le résultat de la commande `uptime -s | sed -e 's/(.*) (.*)/\1/'` ?

4. Quelle option doit être passée à `grep` pour qu'il compte les lignes correspondantes au lieu de les afficher ?

Exercices d'approfondissement

1. La structure de base d'un fichier HTML commence par les éléments `html`, `head` et `body`, par exemple :

```
<html>
<head>
  <title>Site d'actualités</title>
</head>
<body>
  <h1>À la une</h1>
  <p>Informations intéressantes.</p>
</body>
</html>
```

   Décrivez comment les adresses pourraient être utilisées dans `sed` pour afficher uniquement l'élément `body` et son contenu.

2. Quelle expression `sed` supprimera toutes les balises d'un document HTML, en conservant uniquement le texte rendu ?

3. Les fichiers avec l'extension `.ovpn` sont très utilisés pour configurer les clients VPN étant donné qu'ils contiennent non seulement les paramètres, mais aussi le contenu des clés et des certificats pour le client. Ces clés et certificats se trouvent à l'origine dans des fichiers séparés, et doivent donc être copiés dans le fichier `.ovpn`. Voici un extrait d'un gabarit `.ovpn` :

```
client
dev tun
remote 192.168.1.155 1194
<ca>
ca.crt
</ca>
<cert>
client.crt
</cert>
<key>
client.key
</key>
<tls-auth>
```

```
ta.key
</tls-auth>
```

En partant du fait que les fichiers `ca.crt`, `client.crt`, `client.key` et `ta.key` se trouvent dans le répertoire courant, comment la configuration du gabarit pourrait-elle être modifiée par `sed` de manière à remplacer chaque nom de fichier par son contenu ?

Résumé

Cette leçon présente les deux commandes Linux les plus importantes en rapport avec les expressions régulières : `grep` et `sed`. Les scripts et les commandes combinées s'appuient sur `grep` et `sed` pour effectuer toute une série de tâches de filtrage et d'analyse de texte. Voici les étapes de la leçon :

- Comment utiliser `grep` et ses déclinaisons comme `egrep` et `fgrep`.
- Comment utiliser `sed` et ses instructions internes pour manipuler du texte.
- Exemples pratiques d'expressions régulières qui utilisent `grep` et `sed`.

Réponses aux exercices guidés

1. La commande `last` affiche une liste des derniers utilisateurs connectés, y compris leurs adresses IP d'origine. Comment pourrait-on utiliser la commande `egrep` pour filtrer les résultats de `last`, en affichant uniquement les occurrences d'une adresse IPv4, et en supprimant toute information supplémentaire dans la ligne correspondante ?

   ```
   last -i | egrep -o '[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}'
   ```

2. Quelle option doit être utilisée avec `grep` pour filtrer correctement la sortie générée par la commande `find` exécutée avec l'option `-print0` ?

   L'option `-z` ou `--null-data`, comme dans `find . -print0 | grep -z expression`.

3. La commande `uptime -s` affiche la dernière date à laquelle le système a été démarré, comme dans *2019-08-05 20:13:22*. Quel sera le résultat de la commande `uptime -s | sed -e 's/(.*) (.*)/\1/'` ?

   Une erreur se produira. Par défaut, les parenthèses doivent être échappées pour utiliser les références inverses dans `sed`.

4. Quelle option doit être passée à `grep` pour qu'il compte les lignes correspondantes au lieu de les afficher ?

   L'option `-c`.

Réponses aux exercices d'approfondissement

1. La structure de base d'un fichier HTML commence par les éléments `html`, `head` et `body`, par exemple :

   ```
   <html>
   <head>
     <title>Site d'actualités</title>
   </head>
   <body>
     <h1>À la une</h1>
     <p>Informations intéressantes.</p>
   </body>
   </html>
   ```

   Décrivez comment les adresses pourraient être utilisées dans `sed` pour afficher uniquement l'élément `body` et son contenu.

   Pour n'afficher que le `body`, les adresses doivent être `/<body>/,/<\/body>/`, comme dans `sed -n -e '/<body>/,/<\/body>/p'`. L'option `-n` est fournie à `sed` pour qu'il n'affiche pas les lignes par défaut, d'où la commande `p` à la fin de l'expression `sed` pour afficher les lignes correspondantes.

2. Quelle expression `sed` supprimera toutes les balises d'un document HTML, en conservant uniquement le texte rendu ?

   L'expression `sed s/<[^>]*>//g` remplacera tout contenu inclus dans `<>` par une chaîne de caractères vide.

3. Les fichiers avec l'extension `.ovpn` sont très utilisés pour configurer les clients VPN étant donné qu'ils contiennent non seulement les paramètres, mais aussi le contenu des clés et des certificats pour le client. Ces clés et certificats se trouvent à l'origine dans des fichiers séparés, et doivent donc être copiés dans le fichier `.ovpn`. Voici un extrait d'un gabarit `.ovpn` :

   ```
   client
   dev tun
   remote 192.168.1.155 1194
   <ca>
   ca.crt
   </ca>
   <cert>
   ```

```
client.crt
</cert>
<key>
client.key
</key>
<tls-auth>
ta.key
</tls-auth>
```

En partant du fait que les fichiers `ca.crt`, `client.crt`, `client.key` et `ta.key` se trouvent dans le répertoire courant, comment la configuration du gabarit pourrait-elle être modifiée par `sed` de manière à remplacer chaque nom de fichier par son contenu ?

La commande

```
sed -r -e 's/(^[^.]*)\.(crt|key)$/cat \1.\2/e' < client.template > client.ovpn
```

remplace chaque ligne qui se termine par `.crt` ou `.key` par le contenu d'un fichier dont le nom est identique à la ligne. L'option `-r` indique à `sed` d'utiliser des expressions régulières étendues, tandis que `e` à la fin de l'expression indique à `sed` de remplacer les correspondances par la sortie de la commande `cat \1.\2`. Les références inverses `\1` et `\2` correspondent au nom du fichier et à l'extension trouvés dans la correspondance.

103.8 Édition de fichier simple

Reference to LPI objectives

LPIC-1 v5, Exam 101, Objective 103.8

Weight

3

Key knowledge areas

- Déplacement dans un document édité avec vi.
- Compréhension et utilisation des modes de base de vi.
- Insertion, modification, suppression, copie et recherche de texte dans vi.
- Connaissance de base de Emacs, nano et vim.
- Configuration de l'éditeur de texte par défaut.

Partial list of the used files, terms and utilities

- `vi`
- `/`, `?`
- `h`, `j`, `k`, `l`
- `i`, `o`, `a`
- `d`, `p`, `y`, `dd`, `yy`
- `ZZ`, `:w!`, `:q!`
- `EDITOR`

Linux Professional Institute

103.8 Leçon 1

Certification :	LPIC-1
Version :	5.0
Thème :	103 Commandes GNU et Unix
Objectif :	103.8 Édition élémentaire de fichiers textes
Leçon :	1 sur 1

Introduction

Dans la plupart des distributions Linux, `vi` — abréviation de “visual” — est pré-installé et c’est l’éditeur standard dans l’environnement shell. Vi est un éditeur de texte interactif, il affiche le contenu du fichier à l’écran pendant le processus d’édition. En tant que tel, il permet à l’utilisateur de se déplacer et d’apporter des modifications à n’importe quel endroit du document. Cependant, contrairement aux éditeurs visuels du bureau graphique, l’éditeur `vi` est une application shell avec des raccourcis clavier pour chaque tâche d’édition.

Une alternative à `vi`, appelée `vim` (*vi improved*), est parfois utilisée comme remplacement moderne de `vi`. Parmi d’autres améliorations, `vim` supporte la coloration syntaxique, l’annulation/rétablissement à plusieurs niveaux et l’édition multi-documents. Bien que plus performant, `vim` reste entièrement compatible avec `vi`, ce qui rend les deux éditeurs indiscernables pour la plupart des tâches.

La technique standard pour lancer `vi` consiste à lui fournir le chemin d’accès à un fichier comme paramètre. Pour aller directement à une ligne spécifique, son numéro doit être indiqué avec un signe plus, comme dans `vi +9 /etc/fstab` pour ouvrir `/etc/fstab/` et placer le curseur à la neuvième ligne. Sans numéro, le signe plus seul place le curseur à la dernière ligne.

L'interface de `vi` est très simple : tout l'espace disponible dans la fenêtre du terminal est occupé pour présenter un fichier, normalement renseigné comme argument de commande, à l'utilisateur. Les seuls indices visuels sont une ligne de bas de page montrant la position actuelle du curseur et un tilde `~` indiquant la fin du fichier. Il existe différents modes d'exécution pour `vi`, dans lesquels le comportement du programme change. Les plus courants sont : le *mode insertion* et le *mode commande*.

Le mode insertion

Le mode insertion est assez intuitif : le texte apparaît à l'écran au fur et à mesure que vous le tapez au clavier. C'est ce type d'interaction que la plupart des utilisateurs attendent d'un éditeur de texte, mais ce n'est pas comme ça que `vi` présente un document au départ. Pour activer le mode insertion, l'utilisateur doit exécuter la commande correspondante en mode commande. La touche `Esc` termine le mode insertion et retourne en mode commande, le mode par défaut de `vi`.

NOTE

Si vous souhaitez en savoir plus sur les autres modes d'exécution, lancez `vi` et tapez :

```
:help vim-modes-intro
```

Le mode commande

Le mode commande — également connu sous le nom de mode normal — est le mode de démarrage par défaut de `vi`. Dans ce mode, les touches du clavier sont associées à des commandes pour la navigation et la manipulation de texte. La plupart des commandes dans ce mode sont des touches uniques. Voici quelques-unes des touches et leurs fonctions en mode commande :

`0`, **`$`**
: Aller au début et à la fin de la ligne.

`1G`, **`G`**
: Aller au début et à la fin du document.

`(`, **`)`**
: Allez au début et à la fin de la phrase.

`{`, **`}`**
: Allez au début et à la fin du paragraphe.

`w`, `W`

Sauter un mot, et sauter un mot en incluant la ponctuation.

`h`, `j`, `k`, `l`

Gauche, bas, haut, droite.

`e` ou `E`

Aller à la fin du mot en cours.

`/`, `?`

Recherche en avant et recherche inverse.

`i`, `I`

Activer le mode insertion avant la position actuelle du curseur et au début de la ligne actuelle.

`a`, `A`

Activer le mode insertion après la position actuelle du curseur et à la fin de la ligne actuelle.

`o`, `O`

Ajouter une nouvelle ligne et passer en mode insertion à la ligne suivante ou à la ligne précédente.

`s`, `S`

Effacer le caractère sous le curseur ou la ligne entière et passer en mode insertion.

`c`

Modifier le(s) caractère(s) sous le curseur.

`r`

Remplacer le caractère sous le curseur.

`x`

Effacer les caractères sélectionnés ou le caractère sous le curseur.

`v`, `V`

Démarrer une nouvelle sélection avec le caractère en cours ou la ligne entière.

`y`, `yy`

Copier le(s) caractère(s) ou la ligne en entier.

`p`, `P`

Coller le contenu copié, après ou avant la position actuelle.

`u`

Annuler la dernière action.

`Ctrl-R`

Refaire la dernière action.

`ZZ`

Fermer et enregistrer.

`ZQ`

Fermer sans enregistrer.

Si elle est précédée d'un chiffre, la commande sera exécutée le même nombre de fois. Par exemple, appuyez sur `3yy` pour copier la ligne en cours et les deux lignes suivantes, appuyez sur `d5w` pour effacer le mot actuel et les 4 mots qui suivent, et ainsi de suite.

La plupart des tâches d'édition combinent plusieurs commandes. Par exemple, la séquence de touches `vey` est utilisée pour copier une sélection à partir de la position actuelle jusqu'à la fin du mot en cours. La répétition de commandes peut également être utilisée dans ces combinaisons, ainsi `v3ey` copierait une sélection en partant de la position actuelle jusqu'à la fin du troisième mot à partir de cette position.

`vi` permet d'organiser le texte copié dans des registres, ce qui permet de conserver simultanément des contenus distincts. Un registre est spécifié par un caractère précédé de `"` et une fois créé, il est conservé jusqu'à la fin de la session en cours. La séquence de touches `"ly` crée un registre contenant la sélection courante, et qui sera accessible par la touche `l`. Ensuite, le registre `l` peut être collé avec `"lp`.

On peut également définir des marques personnalisées à des positions arbitraires le long du texte, ce qui permet de naviguer rapidement d'une marque à l'autre. Les marques sont créées en appuyant sur la touche `m` puis sur une touche pour adresser la position actuelle. Une fois que c'est fait, le curseur reviendra à la position marquée lorsque `'` suivi de la touche choisie sont actionnés.

N'importe quelle séquence de touches pourra être enregistrée en tant que macro pour une exécution ultérieure. Une macro peut être enregistrée, par exemple, pour entourer un texte sélectionné de guillemets. Dans un premier temps, une chaîne de texte est sélectionnée et la touche `q` est pressée, suivie d'une touche de registre à laquelle on associera la macro, par exemple `d`. La ligne `recording @d` apparaît alors dans la ligne de bas de page, indiquant que

l'enregistrement est en cours. On part du principe qu'un texte est déjà sélectionné, la première commande est donc `x` pour supprimer (et copier automatiquement) le texte sélectionné. La touche `i` est pressée pour insérer deux guillemets à la position courante, puis `Esc` revient en mode commande. La dernière commande est `P`, pour réinsérer la sélection effacée juste avant le dernier guillemet. En appuyant à nouveau sur `q`, on termine l'enregistrement. A présent, une macro composée de la séquence de touches `x`, `i`, `""`, `Esc` et `P` s'exécutera chaque fois que les touches `@d` seront actionnées en mode commande, où `d` est la touche de registre associée à la macro.

En revanche, la macro ne sera disponible que pour la session en cours. Pour rendre les macros persistantes, elles devront être stockées dans le fichier de configuration. Comme la plupart des distributions modernes utilisent *vim* comme éditeur compatible *vi*, le fichier de configuration de l'utilisateur est `~/.vimrc`. Dans `~/.vimrc`, la ligne `let @d = 'xi""^[P'` va affecter le registre `d` à la séquence de touches entre guillemets simples. Le même registre précédemment assigné à une macro pourra être utilisé pour coller sa séquence de touches.

Les commandes à deux points

Le mode commande supporte également un autre ensemble de commandes `vi` : les commandes *à deux points*. Comme leur nom l'indique, elles sont exécutées après avoir appuyé sur la touche deux points `:` en mode commande. Les commandes à deux points permettent à l'utilisateur d'effectuer des recherches, de sauvegarder, de quitter, d'exécuter des commandes shell, de modifier les paramètres de `vi`, etc. Pour revenir en mode commande, il faut exécuter la commande `:visual` ou appuyer sur la touche Entrée sans aucune commande. Quelques-unes des commandes à deux points les plus courantes sont indiquées ici (l'initiale ne fait pas partie de la commande) :

`:s/REGEX/TEXT/g`

Remplace toutes les occurrences de l'expression régulière `REGEX` par `TEXT` dans la ligne courante. Elle accepte la même syntaxe que la commande `sed`, y compris les adresses.

`:!`

Exécute la commande shell qui suit.

`:quit` ou `:q`

Quitte le programme.

`:quit!` ou `:q!`

Quitte le programme sans enregistrer les modifications.

`:wq`

Enregistrer et quitter.

`:exit` ou `:x` ou `:e`

Enregistrer et quitter le cas échéant.

`:visual`

Revenir en mode navigation.

Le programme standard `vi` est capable d'effectuer la plupart des tâches d'édition de texte, mais tout autre éditeur en ligne de commande pourra être utilisé pour éditer des fichiers texte dans l'environnement shell.

TIP

Les utilisateurs novices auront peut-être du mal à mémoriser toutes les commandes de `vi` à la fois. Les distributions qui adoptent `vim` disposent également de la commande `vimtutor`, qui utilise `vim` lui-même pour ouvrir un tutoriel interactif des principales activités. Le fichier est une copie éditable qui pourra être utilisée pour pratiquer les commandes et s'y habituer progressivement.

Les éditeurs alternatifs

Les utilisateurs qui ne connaissent pas `vi` peuvent avoir du mal à s'y adapter, étant donné que son fonctionnement n'est pas intuitif. Une alternative plus simple est GNU `nano`, un petit éditeur de texte qui offre toutes les fonctionnalités de base de l'édition de texte comme annuler/refaire, la coloration syntaxique, la recherche interactive et le remplacement, l'indentation automatique, les numéros de ligne, la complétion de mots, le verrouillage des fichiers, les fichiers de sauvegarde, et le support de l'internationalisation. Contrairement à `vi`, toutes les touches actionnées sont insérées dans le document en cours d'édition. Les commandes dans `nano` sont fournies en utilisant la touche `Ctrl` ou la touche Meta (selon le système, Meta est `Alt` ou `commande`).

`Ctrl-6` ou `Meta-A`

Démarre une nouvelle sélection. On peut également créer une sélection en appuyant sur la touche `Maj` et en déplaçant le curseur.

`Meta-6`

Copier la sélection en cours.

`Ctrl-K`

Couper la sélection en cours.

`Ctrl-U`

Coller le contenu dans le presse-papiers.

`Meta-U`

Annuler.

`Meta-E`

Refaire.

`Ctrl-\`

Remplacer le texte dans la sélection.

`Ctrl-T`

Lancer une session de vérification orthographique pour le document ou la sélection en cours.

Emacs est un autre éditeur de texte très populaire pour l'environnement shell. Le texte est inséré en le tapant, comme dans `nano`, mais la navigation dans le document est assistée par des commandes au clavier, comme dans `vi`. Emacs comprend un grand nombre de fonctionnalités qui en font plus qu'un simple éditeur de texte. C'est aussi un EDI (*environnement de développement intégré*) capable de compiler, d'exécuter et de tester des programmes. Emacs peut être configuré comme un client de courrier électronique, un lecteur de nouvelles ou un client RSS, ce qui en fait une véritable suite logicielle dédiée à la productivité.

Le shell lui-même lancera un éditeur de texte par défaut, habituellement `vi`, chaque fois que cela sera nécessaire. C'est le cas, par exemple, lorsque `crontab -e` est exécuté pour éditer les tâches cron (*cronjobs*). Bash utilise les variables de session `VISUAL` ou `EDITOR` pour déterminer l'éditeur de texte par défaut de l'environnement shell. Par exemple, la commande `export EDITOR=nano` définit `nano` comme éditeur de texte par défaut pour la session en cours de l'interpréteur de commandes. Pour rendre ce changement persistant d'une session à l'autre, la commande devra être ajoutée à `~/.bash_profile`.

Exercices guidés

1. `vi` est utilisé principalement pour éditer des fichiers de configuration et du code source, où l'indentation permet d'identifier les sections d'un texte. Une sélection peut être indentée vers la gauche en appuyant sur `<` et vers la droite en appuyant sur `>`. Quelles touches faut-il utiliser en mode commande pour indenter la sélection en cours de trois crans vers la gauche ?

2. Une ligne entière peut être sélectionnée en appuyant sur `V` en mode commande `vi`. En revanche, le caractère de fin de ligne sera également pris en compte. Quelles touches doivent être actionnées en mode commande pour sélectionner à partir du caractère de départ jusqu'au dernier caractère mais sans le caractère de fin de ligne ?

3. Comment exécuter `vi` en ligne de commande pour ouvrir `~/.bash_profile` et passer directement à la dernière ligne ?

4. Quelles touches faut-il actionner en mode commande `vi` pour effacer les caractères à partir de la position actuelle du curseur jusqu'au prochain point ?

Exercices d'approfondissement

1. `vim` permet de sélectionner des blocs de texte de largeur arbitraire, et pas seulement des sections composées de lignes entières. En appuyant sur `Ctrl-V` en mode commande, une sélection est faite en déplaçant le curseur vers le haut, le bas, la gauche et la droite. En utilisant cette méthode, comment peut-on supprimer un bloc qui commence au premier caractère de la ligne en cours et qui contient les huit colonnes et les cinq lignes de texte subséquentes ?

2. Une session `vi` a été interrompue par une coupure de courant inopinée. Lors de la réouverture du fichier, `vi` demande à l'utilisateur s'il souhaite récupérer le fichier swap (une copie automatique effectuée par `vi`). Que doit faire l'utilisateur pour se débarrasser du fichier swap ?

3. Dans une session `vim`, une ligne a été copiée auparavant dans le registre `l`. Quelle combinaison de touches permet d'enregistrer une macro dans le registre `a` pour coller la ligne dans le registre `l` immédiatement avant la ligne actuelle ?

Résumé

Cette leçon présente l'éditeur de texte standard de l'environnement Linux : l'éditeur `vi`. Malgré son côté intimidant pour l'utilisateur non averti, `vi` possède des fonctionnalités qui en font un excellent choix pour l'édition de textes techniques et non techniques. La leçon aborde les points suivants :

- Utilisation basique de `vi` et fonctionnalités utiles.
- Présentation de `vim` — le `vi` amélioré — et autres éditeurs alternatifs.
- Comment définir l'éditeur de texte par défaut pour l'environnement shell.

Voici les procédures et les commandes abordées :

- L'éditeur `vi` et sa mouture améliorée `vim`.
- Édition de texte de base dans `vi`.
- Les éditeurs alternatifs `emacs` et `nano`.

Réponses aux exercices guidés

1. `vi` est utilisé principalement pour éditer des fichiers de configuration et du code source, où l'indentation permet d'identifier les sections d'un texte. Une sélection peut être indentée vers la gauche en appuyant sur `<` et vers la droite en appuyant sur `>`. Quelles touches faut-il utiliser en mode normal pour indenter la sélection en cours de trois crans vers la gauche ?

   Les touches `3<`, qui signifient trois crans vers la gauche.

2. Une ligne entière peut être sélectionnée en appuyant sur `V` en mode commande `vi`. En revanche, le caractère de fin de ligne sera également pris en compte. Quelles touches doivent être actionnées en mode commande pour sélectionner à partir du caractère de départ jusqu'au dernier caractère mais sans le caractère de fin de ligne ?

   Les touches `0v$h`, qui signifient `0` ("aller au début de la ligne"), `v` ("commencer la sélection des caractères"), `$` ("aller à la fin de la ligne") et `h` ("reculer d'un cran").

3. Comment exécuter `vi` en ligne de commande pour ouvrir `~/.bash_profile` et passer directement à la dernière ligne ?

   La commande `vi + ~/.bash_profile` va ouvrir le fichier et placer le curseur à la dernière ligne.

4. Quelles touches faut-il actionner en mode commande `vi` pour effacer les caractères à partir de la position actuelle du curseur jusqu'au prochain point ?

   Les touches `dt.`, c'est-à-dire `d` ("commencer la suppression"), `t` ("passer au caractère suivant") et `.` (point).

Réponses aux exercices d'approfondissement

1. `vim` permet de sélectionner des blocs de texte de largeur arbitraire, et pas seulement des sections composées de lignes entières. En appuyant sur `Ctrl-V` en mode commande, une sélection est faite en déplaçant le curseur vers le haut, le bas, la gauche et la droite. En utilisant cette méthode, comment peut-on supprimer un bloc qui commence au premier caractère de la ligne en cours et qui contient les huit colonnes et les cinq lignes de texte subséquentes ?

   La combinaison de touches `0`, `Ctrl-V` et `8l5jd` va sélectionner et supprimer le bloc correspondant.

2. Une session `vi` a été interrompue par une coupure de courant inopinée. Lors de la réouverture du fichier, `vi` demande à l'utilisateur s'il souhaite récupérer le fichier swap (une copie automatique effectuée par `vi`). Que doit faire l'utilisateur pour se débarrasser du fichier swap ?

   Appuyer sur `d` à l'invite de `vi`.

3. Dans une session `vim`, une ligne a été copiée auparavant dans le registre `l`. Quelle combinaison de touches permet d'enregistrer une macro dans le registre `a` pour coller la ligne dans le registre `l` immédiatement avant la ligne actuelle ?

   La combinaison de touches `qa"lPq`, c'est-à-dire `q` ("démarrer l'enregistrement de la macro"), `a` ("assigner le registre `a` à la macro"), `"l` ("sélectionner le texte dans le registre `l`\`"), \`P` ("coller avant la ligne courante") et `q` ("terminer l'enregistrement de la macro").

Linux Professional Institute

Thème 104 : Disques, systèmes de fichiers Linux , arborescence de fichiers standard (FHS)

104.1 Création des partitions et des systèmes de fichiers

Référence aux objectifs de LPI

LPIC-1 v5, Exam 101, Objective 104.1

Valeur

2

Liste partielle de termes, fichiers et utilitaires utilisés pour cet objectif

- Gestion des tables de partition MBR et GPT
- Utilisation des différentes commandes mkfs pour le paramétrage des partitions et la création des différents systèmes de fichiers comme :
  - ext2/ext3/ext4
  - XFS
  - VFAT
  - exFAT
- Connaissance de base de Btrfs, y compris les systèmes de fichiers sur plusieurs périphériques, la compression et les sous-volumes.

Partial list of the used files, terms and utilities

- `fdisk`
- `gdisk`
- `parted`
- `mkfs`
- `mkswap`

104.1 Leçon 1

Certification :	LPIC-1
Version :	5.0
Thème :	104 Disques, systèmes de fichiers Linux, arborescence de fichiers standard (FHS)
Objectif :	104.1 Créer des partitions et des systèmes de fichiers
Leçon :	1 sur 1

Introduction

Peu importe le système d'exploitation, un disque dur doit être partitionné avant de pouvoir être utilisé. Une partition est un sous-ensemble logique du disque physique et les informations relatives aux partitions sont stockées dans une table de partitionnement. Cette table comprend des informations sur le premier et le dernier secteur de la partition et son type, ainsi que d'autres détails sur chaque partition.

En règle générale, chaque partition est considérée par le système d'exploitation comme un "disque" à part entière, même si elles résident toutes sur le même support physique. Sur les systèmes Windows, on leur attribue des lettres comme `C:` (historiquement le disque principal), `D:` et ainsi de suite. Sous Linux, chaque partition est assignée à un fichier de périphérique sous `/dev`, comme `/dev/sda1` ou `/dev/sda2`.

Dans cette leçon, vous allez apprendre à créer, supprimer, restaurer et redimensionner des partitions en utilisant les trois outils les plus courants (`fdisk`, `gdisk` et `parted`), à créer un système de fichiers sur ces partitions et enfin à créer et configurer une partition d'échange (*swap*

partition) ou un fichier d'échange (*swap file*) pour l'utiliser comme mémoire virtuelle.

NOTE Pour des raisons historiques, nous appellerons les supports de stockage "disques" tout au long de cette leçon, même si les systèmes de stockage modernes, comme les disques SSD et les mémoires flash, ne contiennent pas de "disques" à proprement parler.

Comprendre le MBR et le GPT

Il existe principalement deux méthodes pour enregistrer les informations relatives aux partitions sur les disques durs. La première est le MBR (*Master Boot Record*), et la seconde est le GPT (*GUID Partition Table*).

MBR

Il s'agit là d'un vestige des débuts de MS-DOS (plus précisément de PC-DOS 2.0 de 1983). Pendant quelques décennies, c'était là le schéma de partitionnement standard sur les PC. La table de partitionnement est stockée sur le premier secteur d'un disque, appelé secteur d'amorçage (*boot sector*), avec un chargeur d'amorçage comme GRUB sur les systèmes Linux. En revanche, le MBR présente une série de limitations gênantes sur les systèmes modernes, comme l'impossibilité d'adresser des disques d'une taille supérieure à 2 To, et la limite maximale de 4 partitions primaires par disque.

GUID

Un système de partitionnement qui corrige la plupart des restrictions imposées par le MBR. Il n'y a pas de limite pratique pour la taille du disque, et le nombre maximum de partitions n'est restreint que par le système d'exploitation lui-même. On le trouve plus fréquemment sur les machines modernes qui utilisent l'UEFI au lieu de l'ancien BIOS des PC.

Dans le cadre des tâches d'administration système, il est fort possible que vous soyez confronté aux deux schémas. Il est donc important de savoir comment utiliser les outils associés à chacun d'entre eux pour créer, supprimer ou modifier des partitions.

Gérer les partitions MBR avec FDISK

L'outil standard pour gérer les partitions MBR sous Linux est `fdisk`. C'est un programme interactif, piloté par un menu. Pour l'utiliser, tapez `fdisk` suivi du nom du périphérique associé au disque que vous voulez modifier. Par exemple, la commande

```
# fdisk /dev/sda
```

va modifier la table de partitionnement du premier périphérique SATA (`sda`) du système. Gardez à l'esprit que vous devez spécifier le périphérique correspondant au disque physique, et non pas l'une de ses partitions (comme `/dev/sda1`).

NOTE Toutes les opérations sur les disques dans cette leçon devront être effectuées en tant qu'utilisateur `root` (l'administrateur du système), ou avec les privilèges de `root` en utilisant `sudo`.

Au lancement, `fdisk` affiche un message d'accueil ainsi qu'un avertissement, et il attend vos commandes.

```
# fdisk /dev/sda
Welcome to fdisk (util-linux 2.33.1).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.

Command (m for help):
```

L'avertissement est important. Vous pouvez créer, éditer ou supprimer des partitions à volonté, mais *rien* ne sera écrit sur le disque à moins que vous n'utilisiez la commande write (`w`). Vous pouvez donc vous "entraîner" sans risque de perdre des données, tant que vous évitez d'utiliser la touche `w`. Pour quitter `fdisk` sans sauvegarder les modifications, utilisez la commande `q`.

NOTE Ceci étant dit, ne vous entraînez jamais sur un disque contenant des données importantes, car il y a toujours des risques. Utilisez plutôt un disque externe disponible ou une clé USB.

Afficher la table de partitionnement en cours

La commande `p` est utilisée pour afficher la table de partitionnement en cours. Le résultat ressemble à ceci :

```
Command (m for help): p
Disk /dev/sda: 111.8 GiB, 120034123776 bytes, 234441648 sectors
Disk model: CT120BX500SSD1
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x97f8fef5

Device     Boot     Start       End   Sectors   Size Id Type
```

```
/dev/sda1            4096 226048942 226044847 107.8G 83 Linux
/dev/sda2       226048944 234437550   8388607     4G 82 Linux swap / Solaris
```

Voici la signification de chacune des colonnes :

`Device`
: Le périphérique affecté à la partition.

`Boot`
: Indique si la partition est amorçable (`bootable`) ou non.

`Start`
: Le secteur où commence la partition.

`End`
: Le secteur où se termine la partition.

`Sectors`
: Le nombre total de secteurs dans la partition. Multipliez-le par la taille d'un secteur pour obtenir la taille de la partition en octets.

`Size`
: La taille de la partition au format "humainement lisible". Dans l'exemple ci-dessus, les valeurs sont exprimées en gigaoctets.

`Id`
: La valeur numérique qui représente le type de la partition.

`Type`
: La description du type de partition.

Partitions primaires et étendues

Sur un disque MBR, vous pouvez avoir deux types de partitions, *primaire* et *étendue*. Comme nous l'avons dit plus haut, vous ne pouvez avoir que quatre partitions primaires sur le disque, et si vous voulez rendre le disque "amorçable", la première partition doit être une partition primaire.

Une manière de contourner cette limitation consiste à créer une partition étendue qui sert de conteneur pour les partitions *logiques*. Vous pourriez avoir, par exemple, une partition primaire, une partition étendue occupant le reste de l'espace disque et cinq partitions logiques à l'intérieur de celle-ci.

Pour un système d'exploitation comme Linux, les partitions primaires et étendues sont gérées exactement de la même manière, il n'y a donc pas le moindre "avantage" à utiliser l'une plutôt que l'autre.

Créer une partition

Pour créer une partition, utilisez la commande `n`. Par défaut, les partitions seront créées au début de l'espace non alloué sur le disque. Vous devrez indiquer le type de partition (primaire ou étendue), le premier secteur et le dernier secteur.

Pour le premier secteur, vous pouvez généralement accepter la valeur par défaut proposée par `fdisk`, à moins que vous n'ayez besoin d'une partition qui commence à un secteur donné. Au lieu de spécifier le dernier secteur, vous pouvez très bien préciser une taille suivie des lettres `K`, `M`, `G`, `T` ou `P` (Kilo, Méga, Giga, Téra ou Péta). Ainsi, si vous souhaitez créer une partition de 1 Go, vous pouvez spécifier `+1G` comme `Last sector`, et `fdisk` va dimensionner la partition en fonction. Voici un exemple de création d'une partition primaire :

```
Command (m for help): n
Partition type
   p   primary (0 primary, 0 extended, 4 free)
   e   extended (container for logical partitions)
Select (default p): p
Partition number (1-4, default 1): 1
First sector (2048-3903577, default 2048): 2048
Last sector, +/-sectors or +/-size{K,M,G,T,P} (2048-3903577, default 3903577): +1G
```

Vérifier l'espace non alloué

Si vous souhaitez connaître la quantité d'espace libre sur le disque, vous pouvez utiliser la commande `F` pour afficher l'espace non alloué, comme ceci :

```
Command (m for help): F
Unpartitioned space /dev/sdd: 881 MiB, 923841536 bytes, 1804378 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes

  Start     End Sectors  Size
2099200 3903577 1804378  881M
```

Supprimer des partitions

Pour supprimer une partition, utilisez la commande `d`. `fdisk` vous demandera le numéro de la partition que vous voulez effacer, *sauf* s'il n'y a *qu'une seule* partition sur le disque. Dans ce cas, cette partition sera *sélectionnée et supprimée immédiatement*.

Notez que si vous supprimez une partition étendue, toutes les partitions logiques qu'elle contient seront également supprimées.

Attention aux écarts !

Gardez à l'esprit que lorsque vous créez une nouvelle partition avec `fdisk`, la taille maximale sera limitée à la quantité maximale d'espace *contigu* non alloué sur le disque. Admettons, par exemple, que vous ayez la table de partitionnement suivante :

```
Device     Boot   Start     End Sectors  Size Id Type
/dev/sdd1          2048 1050623 1048576  512M 83 Linux
/dev/sdd2       1050624 2099199 1048576  512M 83 Linux
/dev/sdd3       2099200 3147775 1048576  512M 83 Linux
```

Vous supprimez alors la partition 2 et vérifiez l'espace libre :

```
Command (m for help): F
Unpartitioned space /dev/sdd: 881 MiB, 923841536 bytes, 1804378 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes

  Start     End Sectors  Size
1050624 2099199 1048576  512M
3147776 3903577  755802  369M
```

En additionnant la taille de l'espace non alloué, nous disposons théoriquement de 881 Mo. Or, voyez ce qui se passe lorsque nous essayons de créer une partition de 700 Mo :

```
Command (m for help): n
Partition type
   p   primary (2 primary, 0 extended, 2 free)
   e   extended (container for logical partitions)
Select (default p): p
Partition number (2,4, default 2): 2
First sector (1050624-3903577, default 1050624):
```

```
Last sector, +/-sectors or +/-size{K,M,G,T,P} (1050624-2099199, default 2099199): +700M
Value out of range.
```

Cela est dû au fait que le plus grand espace contigu non alloué sur le disque est le bloc de 512 Mo qui appartenait à la partition 2. Votre nouvelle partition ne peut pas "déborder" vers la partition 3 pour utiliser une partie de l'espace non alloué qui se trouve après.

Changer le type de partition

Il est parfois nécessaire de changer le type de partition, notamment lorsqu'il s'agit de disques qui seront utilisés avec d'autres systèmes d'exploitation et d'autres plateformes. Pour ce faire, utilisez la commande `t` suivie du numéro de la partition que vous souhaitez modifier.

Le type de partition doit être spécifié par son code hexadécimal correspondant, et vous pouvez afficher une liste de tous les codes disponibles en utilisant la commande `l`.

Ne confondez pas le type de partition avec le système de fichiers utilisé. Même si au début il y avait une corrélation entre les deux, vous ne pouvez plus supposer que c'est le cas aujourd'hui. Une partition Linux, par exemple, peut contenir n'importe quel système de fichiers natif de Linux, comme *ext4* ou *ReiserFS*.

TIP Les partitions Linux sont de type `83` (Linux). Les partitions d'échange sont de type `82` (Linux Swap).

Gérer les partitions GUID avec GDISK

L'outil `gdisk` est l'équivalent de `fdisk` pour les disques avec des partitions GPT. En fait, l'interface est inspirée de `fdisk`, avec une invite interactive et les mêmes commandes (ou des commandes très similaires).

Afficher la table de partitionnement en cours

La commande `p` est utilisée pour afficher la table de partitionnement en cours. Le résultat ressemble à ceci :

```
Command (? for help): p
Disk /dev/sdb: 3903578 sectors, 1.9 GiB
Model: DataTraveler 2.0
Sector size (logical/physical): 512/512 bytes
Disk identifier (GUID): AB41B5AA-A217-4D1E-8200-E062C54285BE
Partition table holds up to 128 entries
Main partition table begins at sector 2 and ends at sector 33
```

```
First usable sector is 34, last usable sector is 3903544
Partitions will be aligned on 2048-sector boundaries
Total free space is 1282071 sectors (626.0 MiB)

Number  Start (sector)    End (sector)  Size       Code  Name
   1            2048         2099199   1024.0 MiB  8300  Linux filesystem
   2         2623488         3147775   256.0 MiB   8300  Linux filesystem
```

Nous remarquons d'emblée plusieurs choses :

- Chaque disque possède un identifiant de disque unique (GUID). Il s'agit d'un nombre hexadécimal de 128 bits, attribué de manière aléatoire lors de la création de la table de partitionnement. Étant donné qu'il existe $3{,}4 \times 10^{38}$ valeurs possibles pour ce nombre, les chances que deux disques aléatoires disposent du même GUID sont très faibles. Le GUID peut être utilisé pour identifier les systèmes de fichiers à monter au démarrage (et à quel endroit), ce qui évite d'utiliser le chemin d'accès au périphérique (comme `/dev/sdb`).
- Vous voyez la phrase `Partition table holds up to 128 entries` (la table de partitionnement peut contenir jusqu'à 128 entrées) ? C'est exact, vous pouvez avoir jusqu'à 128 partitions sur un disque GPT. Pour cette raison, il n'est pas nécessaire d'avoir des partitions *primaires* et *étendues*.
- L'espace libre est affiché sur la dernière ligne, il n'est donc pas nécessaire d'avoir un équivalent de la commande `F` de fdisk.

Créer une partition

La commande pour créer une partition est `n`, comme dans `fdisk`. La principale différence est qu'en plus du numéro de partition et du premier et dernier secteur (ou de la taille), vous pouvez également spécifier le type de partition pendant la création. Les partitions GPT supportent beaucoup plus de types que les partitions MBR. Vous pouvez consulter la liste de tous les types supportés en utilisant la commande `l`.

Supprimer une partition

Pour supprimer une partition, tapez `d` et le numéro de la partition. Contrairement à `fdisk`, la première partition ne sera pas sélectionnée automatiquement si c'est la seule sur le disque.

Sur les disques GPT, les partitions peuvent être facilement réorganisées, ou "triées", pour éviter les écarts dans la séquence de numérotation. Pour ce faire, il suffit d'utiliser la commande `s`. Par exemple, prenons un disque avec la table de partitionnement suivante :

```
Number  Start (sector)    End (sector)  Size       Code  Name
```

```
   1            2048         2099199   1024.0 MiB  8300  Linux filesystem
   2         2099200         2361343   128.0 MiB   8300  Linux filesystem
   3         2361344         2623487   128.0 MiB   8300  Linux filesystem
```

Si vous supprimez la deuxième partition, la table deviendra :

```
Number  Start (sector)    End (sector)  Size       Code  Name
   1            2048         2099199   1024.0 MiB  8300  Linux filesystem
   3         2361344         2623487   128.0 MiB   8300  Linux filesystem
```

Si vous utilisez la commande `s`, ce sera :

```
Number  Start (sector)    End (sector)  Size       Code  Name
   1            2048         2099199   1024.0 MiB  8300  Linux filesystem
   2         2361344         2623487   128.0 MiB   8300  Linux filesystem
```

Notez que la troisième partition est devenue la seconde.

Les options de récupération

Les disques GPT enregistrent des copies de sauvegarde de l'en-tête GPT et de la table de partitionnement, ce qui facilite la récupération des disques au cas où ces données sont endommagées. `gdisk` fournit des fonctionnalités pour faciliter ces tâches de récupération, accessibles avec la commande `r`.

Vous pouvez reconstruire un en-tête GPT ou une table de partitionnement endommagés avec `b` et `c` respectivement, ou utiliser l'en-tête et la table pour reconstruire une sauvegarde avec `d` et `e`. Vous pouvez également convertir un MBR en GPT avec `f` et faire l'inverse avec `g`, parmi d'autres opérations. Tapez `?` dans le menu de récupération pour obtenir une liste de toutes les commandes de récupération disponibles ainsi que la description de ce qu'elles font.

Créer des systèmes de fichiers

Le partitionnement du disque n'est que la première étape vers l'utilisation d'un disque. Il faut ensuite formater la partition avec un système de fichiers avant de l'utiliser pour stocker des données.

Un système de fichiers contrôle la manière dont les données sont stockées et accessibles sur le disque. Linux prend en charge plusieurs systèmes de fichiers, certains natifs, comme la famille ext (*Extended Filesystem*), tandis que d'autres proviennent d'autres systèmes d'exploitation, comme

FAT de MS-DOS, NTFS de Windows NT, HFS et HFS+ de Mac OS, etc.

L'outil standard utilisé pour créer un système de fichiers sous Linux est `mkfs`, qui existe en plusieurs "variétés" en fonction du système de fichiers qu'il doit gérer.

Créer un système de fichiers ext2/ext3/ext4

Le système de fichiers ext (*Extended Filesystem*) a été le premier système de fichiers pour Linux. Au fil des années, il a été remplacé par de nouvelles versions appelées ext2, ext3 et ext4. Cette dernière version est actuellement le système de fichiers par défaut pour de nombreuses distributions Linux.

Les outils `mkfs.ext2`, `mkfs.ext3` et `mkfs.ext4` sont utilisés pour créer des systèmes de fichiers ext2, ext3 et ext4. En fait, tous ces "outils" n'existent qu'en tant que liens symboliques vers un autre programme appelé `mke2fs`. `mke2fs` modifie ses paramètres par défaut en fonction du nom par lequel il est invoqué. Ainsi, ils ont tous le même comportement et les mêmes paramètres de ligne de commande.

La forme d'utilisation la plus simple est :

```
# mkfs.ext2 TARGET
```

Où `TARGET` est le nom de la partition dans laquelle le système de fichiers doit être créé. Par exemple, pour créer un système de fichiers ext3 dans la partition `/dev/sdb1`, la commande serait :

```
# mkfs.ext3 /dev/sdb1
```

Au lieu d'utiliser la commande associée au système de fichiers que vous souhaitez créer, vous pouvez passer le paramètre `-t` à `mke2fs` suivi du nom du système de fichiers. Par exemple, les commandes suivantes sont équivalentes, et vont créer un système de fichiers ext4 sur `/dev/sdb1`.

```
# mkfs.ext4 /dev/sdb1
# mke2fs -t ext4 /dev/sdb1
```

Paramètres en ligne de commande

`mke2fs` supporte toute une panoplie de paramètres et d'options en ligne de commande. En voici quelques-uns parmi les plus importants. Tous s'appliquent également à `mkfs.ext2`, `mkfs.ext3` et

`mkfs.ext4` :

`-b SIZE`

Fixe la taille des blocs de données dans le périphérique à `SIZE` (taille), qui peut être de 1024, 2048 ou 4096 octets par bloc.

`-c`

Vérifie que le périphérique cible (`TARGET`) ne contient pas de blocs défectueux avant de créer le système de fichiers. Vous pouvez effectuer une vérification minutieuse, mais beaucoup plus lente, en passant ce paramètre deux fois, comme dans `mkfs.ext4 -c -c TARGET`.

`-d DIRECTORY`

Copie le contenu du répertoire (`DIRECTORY`) spécifié vers la racine du nouveau système de fichiers. Cette option est utile si vous avez besoin de "pré-remplir" le disque avec un ensemble de fichiers prédéfinis.

`-F`

Attention, danger ! Cette option va *forcer* `mke2fs` à créer un système de fichiers, même si les autres options qui lui sont passées ou la cible sont dangereuses ou n'ont aucun sens. Si elle est spécifiée deux fois (comme dans `-F -F`), elle peut même être utilisée pour créer un système de fichiers sur un périphérique monté ou en cours d'utilisation, ce qui est une très, *très* mauvaise idée.

`-L VOLUME_LABEL`

Fixe le nom du volume à celui spécifié dans `VOLUME_LABEL`. Cette étiquette doit comporter 16 caractères au maximum.

`-n`

C'est une option très pratique qui simule la création du système de fichiers et affiche ce qui se passerait si elle était exécutée sans l'option `n`. Considérez-la comme un mode "test" qui permet de bien vérifier les choses avant d'effectuer des changements sur le disque.

`-q`

Mode silencieux. `mke2fs` s'exécutera normalement, mais n'affichera rien dans le terminal. Utile pour lancer `mke2fs` à partir d'un script.

`-U ID`

Cette option définit l'UUID (*Universally Unique Identifier*) d'une partition à la valeur spécifiée par ID. Les UUID sont des nombres de 128 bits en notation hexadécimale qui servent à identifier de manière unique une partition pour le système d'exploitation. Ce numéro est

spécifié sous la forme d'une chaîne de 32 chiffres au format 8-4-4-4-12, c'est-à-dire 8 chiffres, un tiret, 4 chiffres, un tiret, 4 chiffres, un tiret, 4 chiffres, un tiret, 12 chiffres, comme dans `D249E380-7719-45A1-813C-35186883987E`. Au lieu d'un ID, vous pouvez également spécifier des paramètres comme `clear` pour supprimer l'UUID du système de fichiers, `random` pour utiliser un UUID généré de manière aléatoire, ou `time` pour générer un UUID basé sur l'heure.

`-V`

Le mode verbeux permet d'afficher beaucoup plus d'informations que d'habitude pendant l'opération. Utile pour le débogage.

Créer un système de fichiers XFS

XFS est un système de fichiers très performant développé à l'origine par Silicon Graphics en 1993 pour son système d'exploitation IRIX. En raison de ses performances et de sa fiabilité, il est couramment utilisé pour les serveurs et autres environnements qui nécessitent une bande passante élevée (ou garantie) pour le système de fichiers.

Les outils pour gérer les systèmes de fichiers XFS sont inclus dans le paquet `xfsprogs`. Ce paquet devra éventuellement être installé manuellement, étant donné qu'il n'est pas inclus par défaut dans certaines distributions Linux. D'autres, comme Red Hat Enterprise Linux 7, utilisent XFS comme système de fichiers par défaut.

Les systèmes de fichiers XFS sont divisés en deux parties au moins, une section de journalisation (*log section*) qui enregistre l'ensemble des opérations du système de fichiers (communément appelée *journal*), et la section de données (*data section*). La section de journalisation peut être située à l'intérieur de la section de données (comportement par défaut), ou même sur un disque séparé, pour gagner en performance et en fiabilité.

La commande la plus simple pour créer un système de fichiers XFS est `mkfs.xfs TARGET`, où `TARGET` représente la partition sur laquelle vous souhaitez créer le système de fichiers. Par exemple : `mkfs.xfs /dev/sda1`.

Tout comme `mke2fs`, `mkfs.xfs` accepte un certain nombre d'options en ligne de commande. Voici les options les plus courantes.

`-b size=VALUE`

Définit la taille des blocs sur le système de fichiers, exprimée en octets, à celle spécifiée dans `VALUE`. La valeur par défaut est de 4096 octets (4 KiB), la valeur minimale est de 512 et la valeur maximale est de 65536 (64 KiB).

`-m crc=VALUE`

Les paramètres qui commencent par `-m` sont des options de métadonnées. Celui-ci active (si `VALUE` est `1`) ou désactive (si `VALUE` est `0`) l'utilisation des contrôles CRC32c pour vérifier l'intégrité de toutes les métadonnées sur le disque. Cette vérification permet une meilleure détection des erreurs et une récupération en cas de panne liée à des problèmes matériels, c'est pourquoi elle est activée par défaut. L'impact de cette vérification sur les performances devrait être minime, il n'y a donc aucune raison de la désactiver en temps normal.

`-m uuid=VALUE`

Définit l'UUID de la partition à celui spécifié dans `VALUE`. Rappelez-vous que les UUID sont des nombres de 32 caractères (128 bits) en base hexadécimale, spécifiés en groupes de 8, 4, 4, 4 et 12 chiffres séparés par des tirets, comme `1E83E3A3-3AE9-4AAC-BF7E-29DFFECD36C0`.

`-f`

Force la création d'un système de fichiers sur le périphérique cible, même si un système de fichiers y est détecté.

`-l logdev=DEVICE`

Cette option transfère la section de journalisation du système de fichiers sur le périphérique (`DEVICE`) spécifié, au lieu de la ranger dans la section de données.

`-l size=VALUE`

Cette option va fixer la taille de la section de journalisation à celle spécifiée dans `VALUE`. La taille peut être spécifiée en octets, avec des suffixes comme `m` ou `g`. `-l size=10m`, par exemple, va limiter la section de journalisation à 10 mégaoctets.

`-q`

Mode silencieux. Dans ce cas, `mkfs.xfs` ne va pas afficher les paramètres du système de fichiers en cours de création.

`-L LABEL`

Définit l'étiquette du système de fichiers, qui peut comporter douze caractères au maximum.

`-N`

Tout comme l'option `-n` de `mke2fs`, `mkfs.xfs` va afficher tous les paramètres de création du système de fichiers, sans pour autant le créer.

Créer un système de fichiers FAT ou VFAT

Le système de fichiers FAT est apparu avec MS-DOS et a fait l'objet de nombreuses révisions au fil

des ans, pour aboutir finalement au format FAT32 publié en 1996 avec Windows 95 OSR2.

VFAT est une extension du format FAT16 qui prend en charge les noms de fichiers longs (jusqu'à 255 caractères). Les deux systèmes de fichiers sont gérés par le même outil, `mkfs.fat`. `mkfs.vfat` est un alias de cet outil.

Le système de fichiers FAT présente de sérieux inconvénients qui limitent son utilisation sur les disques de grande capacité. FAT16, par exemple, prend en charge des volumes de 4 Go au maximum et une taille de fichiers maximale de 2 Go. FAT32 augmente la taille des volumes jusqu'à 2 Po avec une taille de fichiers maximale de 4 Go. Pour cette raison, les systèmes de fichiers FAT sont aujourd'hui plus couramment utilisés sur les petits disques flash ou les cartes mémoire (d'une taille maximale de 2 Go), ainsi que sur les anciens appareils et les systèmes d'exploitation qui ne prennent pas en charge des systèmes de fichiers plus avancés.

La commande la plus simple pour créer un système de fichiers FAT est `mkfs.fat TARGET`, où `TARGET` représente la partition sur laquelle vous souhaitez créer le système de fichiers. Par exemple : `mkfs.fat /dev/sda1`.

Tout comme d'autres outils, `mkfs.fat` prend en charge un certain nombre d'options en ligne de commande. En voici les plus importantes. Une liste complète avec une description de chaque option peut être consultée dans le manuel de cet outil, à l'aide de la commande `man mkfs.fat`.

`-c`

Vérifie la présence de blocs défectueux sur le périphérique cible avant de créer le système de fichiers.

`-C FILENAME BLOCK_COUNT`

Crée le fichier spécifié par `FILENAME` avec un système de fichiers FAT à l'intérieur, réalisant ainsi une "image disque" vide qui pourra être écrite par la suite sur un périphérique en utilisant un outil tel que `dd` ou montée en tant que périphérique boucle (*loopback device*). Avec cette option, le nombre de blocs dans le système de fichiers (`BLOCK_COUNT`) devra être spécifié après le nom du périphérique.

`-F SIZE`

Sélectionne la taille de la FAT (table d'allocation de fichiers ou *File Allocation Table*), entre 12, 16 ou 32, c'est-à-dire entre FAT12, FAT16 ou FAT32. Si elle n'est pas spécifiée, `mkfs.fat` sélectionnera l'option appropriée en fonction de la taille du système de fichiers.

`-n NAME`

Définit l'étiquette ou le nom du volume pour le système de fichiers. Ce nom peut comporter jusqu'à 11 caractères et il n'y a pas de nom par défaut.

`-v`

Mode verbeux. Affiche beaucoup plus d'informations qu'en temps normal, ce qui peut être utile pour le débogage.

NOTE

`mkfs.fat` *ne peut pas* créer un système de fichiers "amorçable". Selon la page de manuel, "ce n'est pas aussi facile que vous ne le pensez" et cela ne sera pas implémenté.

Créer un système de fichiers exFAT

exFAT est un système de fichiers créé par Microsoft en 2006 pour pallier l'une des principales limitations de FAT32 : la taille des fichiers et des disques. Avec exFAT, la taille maximale des fichiers est de 16 exaoctets (contre 4 Go avec FAT32) et la taille maximale des disques est de 128 pétaoctets.

Comme il est bien pris en charge par les trois principaux systèmes d'exploitation (Windows, Linux et macOS), c'est un excellent choix lorsque l'interopérabilité est recherchée, par exemple sur les lecteurs flash, les cartes mémoire et les disques externes de grande capacité. D'ailleurs, c'est le système de fichiers par défaut, tel que défini par la *SD Association*, pour les cartes mémoire SDXC de plus de 32 Go.

La commande par défaut pour créer des systèmes de fichiers exFAT est `mkfs.exfat`, qui constitue un lien vers `mkexfatfs`. La commande la plus basique est `mkfs.exfat TARGET`, où `TARGET` représente la partition sur laquelle vous souhaitez créer le système de fichiers. Par exemple : `mkfs.exfat /dev/sdb2`.

Contrairement aux autres outils abordés dans cette leçon, `mkfs.exfat` a très peu d'options en ligne de commande. Les voici :

`-i VOL_ID`

Définit l'ID du volume à la valeur spécifiée dans `VOL_ID`. Il s'agit d'un nombre hexadécimal de 32 bits. S'il n'est pas défini, un ID basé sur l'heure actuelle est défini.

`-n NAME`

Définit l'étiquette ou le nom du volume pour le système de fichiers. Ce nom peut comporter jusqu'à 15 caractères et il n'y a pas de nom par défaut.

`-p SECTOR`

Spécifie le premier secteur de la première partition du disque. Ce paramètre est facultatif et la valeur par défaut est zéro.

`-s SECTORS`

Définit le nombre de secteurs physiques par grappe d'allocation. Il doit s'agir d'une puissance de deux, comme 1, 2, 4, 8, etc.

Découvrir le système de fichiers Btrfs

Btrfs (officiellement le *B-Tree Filesystem*, prononcé "Butter FS", "Better FS" ou même "Butterfuss", à vous de choisir) est un système de fichiers développé depuis 2007 pour Linux par la société Oracle et d'autres entreprises, notamment Fujitsu, Red Hat, Intel et SUSE.

Btrfs présente plusieurs caractéristiques qui le rendent particulièrement intéressant sur les systèmes modernes où le stockage massif est courant. Parmi celles-ci, citons la prise en charge de plusieurs périphériques y compris le *striping* (entrelacement de disques), le *mirroring* (disques en miroir) et le *striping+mirroring* (une combinaison des deux approches) comme dans une configuration RAID, la compression transparente, l'optimisation des disques SSD, les sauvegardes incrémentales, les *snapshots* (instantanés), la défragmentation en ligne, les vérifications hors ligne, la prise en charge des sous-volumes (avec quotas), la déduplication et bien d'autres choses encore.

Étant donné qu'il s'agit d'un système de fichiers *copy-on-write* (copie à l'écriture), il est très résilient face aux défaillances. De plus, Btrfs est simple à utiliser et bien supporté par la plupart des distributions Linux. Certaines d'entre elles, comme SUSE, l'utilisent même comme système de fichiers par défaut.

NOTE

Dans le cas d'un système de fichiers traditionnel, lorsque vous souhaitez écraser une partie d'un fichier, les nouvelles données sont écrites directement sur les anciennes données qu'elles vont remplacer. Dans un système de fichiers *copy-on-write*, les nouvelles données sont écrites dans l'espace libre du disque, puis les métadonnées d'origine du fichier sont mises à jour pour pointer vers ces nouvelles données et ce n'est qu'ensuite que les anciennes données sont dégagées, étant donné qu'elles ne sont plus nécessaires. Cette façon de procéder réduit les risques de perte de données en cas de panne, vu que les anciennes données ne sont éliminées que lorsque le système de fichiers est absolument certain qu'elles ne sont plus nécessaires et que les nouvelles données sont bien en place.

Créer un système de fichiers Btrfs

La commande `mkfs.btrfs` est utilisée pour créer un système de fichiers Btrfs. Invoquée sans option, elle crée un système de fichiers Btrfs sur un périphérique donné, comme ceci :

```
# mkfs.btrfs /dev/sdb1
```

TIP Si vous ne disposez pas de l'outil `mkfs.btrfs`, utilisez le gestionnaire de paquets de votre distribution pour chercher le paquet `btrfs-progs`.

Utilisez l'option `-L` pour définir un label (ou nom) pour votre système de fichiers. Les étiquettes Btrfs peuvent contenir jusqu'à 256 caractères, à l'exception des retours à la ligne :

```
# mkfs.btrfs /dev/sdb1 -L "New Disk"
```

TIP Utilisez des apostrophes (comme ci-dessus) si l'étiquette contient des espaces.

Notez cette particularité de Btrfs : vous pouvez passer plusieurs périphériques à la commande `mkfs.btrfs`. Ce faisant, le système de fichiers sera réparti sur tous les périphériques, ce qui ressemble à une configuration RAID ou LVM. Utilisez le paramètre `-m` pour spécifier la distribution des métadonnées dans la matrice de disques. Les paramètres supportés sont `raid0`, `raid1`, `raid5`, `raid6`, `raid10`, `single` et `dup`.

Par exemple, pour créer un système de fichiers qui englobe `/dev/sdb1` et `/dev/sdc1` en concaténant les deux partitions en une seule grande partition, utilisez :

```
# mkfs.btrfs -d single -m single /dev/sdb /dev/sdc
```

WARNING Les systèmes de fichiers répartis sur plusieurs partitions, tels que celui décrit ci-dessus, peuvent sembler avantageux à première vue. Or, ce n'est pas une bonne idée du point de vue de la sécurité des données, étant donné qu'une défaillance sur un seul disque de la matrice entraîne une perte de données inévitable. Le risque augmente avec le nombre de disques utilisés, vu qu'il y a plus de points de défaillance possibles.

Gérer les sous-volumes

Les sous-volumes sont comme des systèmes de fichiers à l'intérieur de systèmes de fichiers. Imaginez-les comme un répertoire qui peut être monté (et traité) comme un système de fichiers à part entière. Les sous-volumes facilitent l'organisation et l'administration du système, étant donné que chacun d'entre eux peut avoir des quotas ou des règles d'instantané distinctes.

NOTE Les sous-volumes ne sont pas des partitions. Une partition alloue un espace fixe sur un disque. Cela peut entraîner des problèmes par la suite, comme le fait qu'une

> partition manque d'espace alors qu'une autre en a encore beaucoup. Ce n'est pas le cas avec les sous-volumes, étant donné qu'ils "partagent" l'espace libre de leur système de fichiers racine et qu'ils s'agrandissent en fonction des besoins.

Admettons que vous ayez un système de fichiers Btrfs monté sur `/mnt/disk` et que vous souhaitiez créer un sous-volume à l'intérieur de celui-ci pour y stocker vos sauvegardes. Appelons-le `BKP` :

```
# btrfs subvolume create /mnt/disk/BKP
```

Ensuite, nous affichons le contenu du système de fichiers `/mnt/disk`. Vous verrez que nous disposons d'un nouveau répertoire, qui porte le nom de notre sous-volume.

```
$ ls -lh /mnt/disk/
total 0
drwxr-xr-x 1 root   root     0 jul 13 17:35 BKP
drwxrwxr-x 1 carol carol 988 jul 13 17:30 Images
```

NOTE Oui, les sous-volumes sont également accessibles comme n'importe quel autre répertoire.

Nous pouvons vérifier que le sous-volume est actif, avec cette commande :

```
# btrfs subvolume show /mnt/disk/BKP/
    Name:           BKP
    UUID:           e90a1afe-69fa-da4f-9764-3384f66fa32e
    Parent UUID:        -
    Received UUID:      -
    Creation time:      2019-07-13 17:35:40 -0300
    Subvolume ID:       260
    Generation:         23
    Gen at creation:    22
    Parent ID:      5
    Top level ID:       5
    Flags:          -
    Snapshot(s):
```

Vous pouvez monter le sous-volume sur `/mnt/BKP` en passant les paramètres `-t btrfs -o subvol=NAME` à la commande `mount` :

```
# mount -t btrfs -o subvol=BKP /dev/sdb1 /mnt/bkp
```

NOTE Le paramètre `-t` spécifie le type de système de fichiers à monter.

Travailler avec des snapshots

Les instantanés (*snapshots*) sont comme des sous-volumes, mais ils sont pré-remplis avec le contenu du volume à partir duquel l'instantané a été pris.

Au moment de la création, un *snapshot* et le volume d'origine ont exactement le même contenu. Mais à partir de là, ils vont évoluer différemment. Les modifications apportées au volume d'origine (comme les fichiers ajoutés, renommés ou supprimés) ne seront pas reflétées dans le *snapshot*, et vice-versa.

Gardez à l'esprit qu'un instantané ne duplique pas les fichiers et qu'il n'occupe pratiquement pas d'espace disque. Il duplique simplement l'arborescence du système de fichiers, tout en pointant vers les données d'origine.

La commande pour créer un *snapshot* est la même que celle utilisée pour créer un sous-volume, il suffit d'ajouter le paramètre `snapshot` après `btrfs subvolume`. La commande ci-dessous va créer un instantané du système de fichiers Btrfs monté sur `/mnt/disk` dans `/mnt/disk/snap` :

```
# btrfs subvolume snapshot /mnt/disk /mnt/disk/snap
```

Maintenant, imaginez que vous avez le contenu suivant dans `/mnt/disk` :

```
$ ls -lh
total 2,8M
-rw-rw-r-- 1 carol carol 109K jul 10 16:22 Galaxy_Note_10.png
-rw-rw-r-- 1 carol carol 484K jul  5 15:01 geminoid2.jpg
-rw-rw-r-- 1 carol carol 429K jul  5 14:52 geminoid.jpg
-rw-rw-r-- 1 carol carol 467K jul  2 11:48 LG-G8S-ThinQ-Mirror-White.jpg
-rw-rw-r-- 1 carol carol 654K jul  2 11:39 LG-G8S-ThinQ-Range.jpg
-rw-rw-r-- 1 carol carol  94K jul  2 15:43 Mimoji_Comparativo.jpg
-rw-rw-r-- 1 carol carol 112K jul 10 16:20 Note10Plus.jpg
drwx------ 1 carol carol  366 jul 13 17:56 snap
-rw-rw-r-- 1 carol carol 118K jul 11 16:36 Twitter_Down_20190711.jpg
-rw-rw-r-- 1 carol carol 324K jul  2 15:22 Xiaomi_Mimoji.png
```

Notez le répertoire `snap` qui contient l'instantané. Supprimons quelques fichiers et vérifions le

contenu du répertoire :

```
$ rm LG-G8S-ThinQ-*
$ ls -lh
total 1,7M
-rw-rw-r-- 1 carol carol 109K jul 10 16:22 Galaxy_Note_10.png
-rw-rw-r-- 1 carol carol 484K jul  5 15:01 geminoid2.jpg
-rw-rw-r-- 1 carol carol 429K jul  5 14:52 geminoid.jpg
-rw-rw-r-- 1 carol carol  94K jul  2 15:43 Mimoji_Comparativo.jpg
-rw-rw-r-- 1 carol carol 112K jul 10 16:20 Note10Plus.jpg
drwx------ 1 carol carol  366 jul 13 17:56 snap
-rw-rw-r-- 1 carol carol 118K jul 11 16:36 Twitter_Down_20190711.jpg
-rw-rw-r-- 1 carol carol 324K jul  2 15:22 Xiaomi_Mimoji.png
```

Cependant, si vous jetez un œil dans le répertoire `snap`, les fichiers que vous venez de supprimer sont toujours là et peuvent être restaurés le cas échéant.

```
$ ls -lh snap/
total 2,8M
-rw-rw-r-- 1 carol carol 109K jul 10 16:22 Galaxy_Note_10.png
-rw-rw-r-- 1 carol carol 484K jul  5 15:01 geminoid2.jpg
-rw-rw-r-- 1 carol carol 429K jul  5 14:52 geminoid.jpg
-rw-rw-r-- 1 carol carol 467K jul  2 11:48 LG-G8S-ThinQ-Mirror-White.jpg
-rw-rw-r-- 1 carol carol 654K jul  2 11:39 LG-G8S-ThinQ-Range.jpg
-rw-rw-r-- 1 carol carol  94K jul  2 15:43 Mimoji_Comparativo.jpg
-rw-rw-r-- 1 carol carol 112K jul 10 16:20 Note10Plus.jpg
-rw-rw-r-- 1 carol carol 118K jul 11 16:36 Twitter_Down_20190711.jpg
-rw-rw-r-- 1 carol carol 324K jul  2 15:22 Xiaomi_Mimoji.png
```

Vous pouvez également créer des *snapshots* en lecture seule. Ils fonctionnent tout comme les *snapshots* en écriture, à la différence que le contenu du *snapshot* ne pourra pas être modifié, il est "figé" dans le temps. Il suffit d'ajouter le paramètre `-r` lors de la création du *snapshot* :

```
# btrfs subvolume snapshot -r /mnt/disk /mnt/disk/snap
```

Quelques remarques sur la compression

Btrfs supporte la compression transparente des fichiers, avec trois algorithmes différents disponibles pour les utilisateurs. Cette opération s'effectue automatiquement pour chaque fichier, à partir du moment où le système de fichiers est monté avec l'option `-o compress`. Les

algorithmes sont suffisamment intelligents pour détecter les fichiers incompressibles et n'essaieront même pas de les compresser, ce qui permet d'économiser les ressources système. Ainsi, dans un même répertoire, vous pouvez avoir en même temps des fichiers compressés et non compressés. L'algorithme de compression par défaut est ZLIB, mais LZO (plus rapide, taux de compression moins performant) ou ZSTD (plus rapide que ZLIB, compression équivalente) sont disponibles, avec plusieurs niveaux de compression (voir l'objectif correspondant dans les options de montage).

Gérer les partitions avec GNU Parted

GNU Parted est un éditeur de partitions (*partition editor*, d'où le nom) très efficace qui peut être utilisé pour créer, supprimer, déplacer, redimensionner, restaurer et copier des partitions. Il peut fonctionner avec les disques GPT et MBR et répond pratiquement à tous vos besoins en matière de gestion de disque.

Il existe une série d'interfaces graphiques qui facilitent l'utilisation de `parted` comme *GParted* pour les environnements de bureau basés sur GNOME ou le *KDE Partition Manager* pour les environnements de bureau KDE. Cependant, vous devriez apprendre à utiliser `parted` en ligne de commande, étant donné que dans un environnement serveur, vous ne pourrez jamais compter sur la présence d'un environnement de bureau graphique.

WARNING

Contrairement à `fdisk` et `gdisk`, `parted` effectue des changements sur le disque *immédiatement* après l'exécution de la commande et sans attendre une autre commande pour écrire les changements sur le disque. Lorsque vous vous entraînez, il est prudent de le faire sur un disque ou une clé USB de rechange ou vide afin d'éviter tout risque de perte de données en cas d'erreur.

La façon la plus simple de commencer à utiliser `parted` consiste à taper `parted DEVICE`, où `DEVICE` représente le périphérique que vous souhaitez gérer (`parted /dev/sdb`). Le programme démarre une interface interactive en ligne de commande comme `fdisk` et `gdisk` avec une invite `(parted)` qui vous permet de saisir des commandes.

```
# parted /dev/sdb
GNU Parted 3.2
Using /dev/sdb
Welcome to GNU Parted! Type 'help' to view a list of commands.

(parted)
```

WARNING

Faites très attention ! Si vous ne spécifiez pas de périphérique, `parted` sélectionnera automatiquement le disque primaire (habituellement

`/dev/sda`) pour effectuer les opérations.

Sélectionner les disques

Pour basculer vers un autre disque que celui spécifié sur la ligne de commande, vous pouvez utiliser la commande `select` suivie du nom du périphérique :

```
(parted) select /dev/sdb
Using /dev/sdb
```

Afficher les informations

La commande `print` sert à obtenir plus d'informations sur une partition donnée voire même tous les périphériques bloc (disques) connectés au système.

Pour obtenir des informations sur la partition en cours, il suffit de taper `print` :

```
(parted) print
Model: ATA CT120BX500SSD1 (scsi)
Disk /dev/sda: 120GB
Sector size (logical/physical): 512B/512B
Partition Table: msdos
Disk Flags:

Number  Start   End    Size    Type     File system     Flags
 1      2097kB  116GB  116GB   primary  ext4
 2      116GB   120GB  4295MB  primary  linux-swap(v1)
```

Vous pouvez obtenir une liste de tous les périphériques connectés à votre système en invoquant `print devices` :

```
(parted) print devices
/dev/sdb (1999MB)
/dev/sda (120GB)
/dev/sdc (320GB)
/dev/mapper/cryptswap (4294MB)
```

Pour obtenir des informations sur tous les périphériques connectés en même temps, vous pouvez utiliser `print all`. Si vous souhaitez savoir combien d'espace libre il y a sur chaque périphérique, utilisez `print free` :

```
(parted) print free
Model: ATA CT120BX500SSD1 (scsi)
Disk /dev/sda: 120GB
Sector size (logical/physical): 512B/512B
Partition Table: msdos
Disk Flags:

Number  Start   End     Size    Type     File system     Flags
        32.3kB  2097kB  2065kB           Free Space
 1      2097kB  116GB   116GB   primary  ext4
        116GB   116GB   512B             Free Space
 2      116GB   120GB   4295MB  primary  linux-swap(v1)
        120GB   120GB   2098kB           Free Space
```

Créer une table de partitionnement sur un disque vide

Pour créer une table de partitionnement sur un disque vide, utilisez la commande `mklabel` suivie du type de table de partitions que vous souhaitez utiliser.

Il existe une multitude de types de tables de partitionnement supportées, mais les deux principaux types que vous devrez connaître sont `msdos` qui est utilisé ici pour se référer à une table de partitionnement MBR, et `gpt` pour se référer à une table de partitionnement GPT. Pour créer une table de partitions MBR, tapez :

```
(parted) mklabel msdos
```

Et pour créer une table de partitionnement GPT, la commande est :

```
(parted) mklabel gpt
```

Créer une partition

Pour créer une partition, la commande `mkpart` sera utilisée selon la syntaxe `mkpart PARTTYPE FSTYPE START END`, où :

`PARTTYPE`

Correspond au type de partition qui peut être `primary` (primaire), `logical` (logique) ou `extended` (étendu) lorsqu'on utilise une table de partitionnement MBR.

`FSTYPE`

Spécifie le système de fichiers qui sera utilisé sur cette partition. Notez bien que `parted` ne crée pas le système de fichiers. Il définit simplement un fanion sur la partition pour indiquer au système d'exploitation le type de données qu'il doit attendre sur la partition.

`START`

Spécifie le point exact du périphérique où la partition démarre. Vous pouvez utiliser différentes unités pour spécifier ce point. `2s` peut être utilisé pour se référer au deuxième secteur du disque, tandis que `1m` se réfère au début du premier mégaoctet du disque. D'autres unités courantes sont `B` (octets) et `%` (pourcentage du disque).

`END`

Spécifie la fin de la partition. Notez que ce n'est *pas* la taille de la partition, c'est *l'endroit du disque où elle se termine*. Par exemple, si vous spécifiez `100m`, la partition se terminera 100 Mo après le début du disque. Vous pouvez utiliser les mêmes unités que pour le paramètre `START`.

Ainsi, la commande :

```
(parted) mkpart primary ext4 1m 100m
```

Crée une partition primaire de type `ext4` qui commence au premier mégaoctet du disque et se termine après le centième mégaoctet.

Supprimer une partition

Pour supprimer une partition, utilisez la commande `rm` suivie du numéro de la partition, que vous pouvez afficher à l'aide de la commande `print`. Ainsi, `rm 2` supprimera la deuxième partition sur le disque sélectionné.

Restaurer des partitions

`parted` est capable de récupérer une partition supprimée. Admettons que vous ayez la structure de partitionnement suivante :

```
Number  Start   End     Size    File system  Name     Flags
 1      1049kB  99.6MB  98.6MB  ext4         primary
 2      99.6MB  200MB   100MB   ext4         primary
 3      200MB   300MB   99.6MB  ext4         primary
```

Vous avez supprimé la partition 2 par inadvertance en utilisant la commande `rm 2`. Pour la

récupérer, vous pouvez utiliser la commande `rescue`, avec la syntaxe `rescue START END`, où `START` est l'emplacement approximatif du début de la partition, et `END` l'emplacement approximatif de la fin de la partition.

`parted` va analyser le disque à la recherche de partitions et proposer de restaurer celles qu'il a trouvées. Dans l'exemple ci-dessus, la partition `2` a commencé à 99,6 Mo et s'est terminée à 200 Mo. Vous pouvez donc utiliser la commande suivante pour restaurer la partition :

```
(parted) rescue 90m 210m
Information: A ext4 primary partition was found at 99.6MB -> 200MB.
Do you want to add it to the partition table?

Yes/No/Cancel? y
```

Cette opération va restaurer la partition et son contenu. Notez que `rescue` ne peut récupérer que les partitions sur lesquelles un système de fichiers est installé. Les partitions vides ne sont pas détectées.

Redimensionner des partitions ext2/3/4

`parted` peut être utilisé pour redimensionner les partitions afin de les agrandir ou de les rétrécir. Ceci étant dit, il faut faire attention à certains détails :

- Pendant l'opération de redimensionnement, la partition doit être inutilisée et démontée.
- Vous aurez besoin de suffisamment d'espace libre *après* la partition pour l'agrandir jusqu'à la taille souhaitée.

La commande est `resizepart`, suivie du numéro de la partition et de l'endroit où elle doit se terminer. Par exemple, si vous avez la table de partitionnement suivante :

```
Number  Start   End     Size    File system  Name     Flags
 1      1049kB  99.6MB  98.6MB  ext4         primary
 2      99.6MB  200MB   100MB   ext4
 3      200MB   300MB   99.6MB  ext4         primary
```

Une tentative d'agrandissement de la partition `1` avec `resizepart` entraînerait un message d'erreur, étant donné qu'avec la nouvelle taille, la partition `1` empiéterait sur la partition `2`. En revanche, la partition `3` peut tout à fait être redimensionnée vu qu'il y a de l'espace libre après cette dernière, ce qui peut être vérifié à l'aide de la commande `print free` :

```
(parted) print free
Model: Kingston DataTraveler 2.0 (scsi)
Disk /dev/sdb: 1999MB
Sector size (logical/physical): 512B/512B
Partition Table: gpt
Disk Flags:

 Number  Start   End     Size    File system  Name     Flags
        17.4kB  1049kB  1031kB  Free Space
 1      1049kB  99.6MB  98.6MB  ext4         primary
 2      99.6MB  200MB   100MB   ext4
 3      200MB   300MB   99.6MB  ext4         primary
        300MB   1999MB  1699MB  Free Space
```

Vous pouvez donc utiliser la commande suivante pour redimensionner la partition 3 à 350 Mo :

```
(parted) resizepart 3 350m

(parted) print
Model: Kingston DataTraveler 2.0 (scsi)
Disk /dev/sdb: 1999MB
Sector size (logical/physical): 512B/512B
Partition Table: gpt
Disk Flags:

Number  Start   End     Size    File system  Name     Flags
 1      1049kB  99.6MB  98.6MB  ext4         primary
 2      99.6MB  200MB   100MB   ext4
 3      200MB   350MB   150MB   ext4         primary
```

Rappelez-vous que le nouveau point final est spécifié en comptant à partir du début du disque. Ainsi, puisque la partition `3` finissait à 300 Mo, elle doit maintenant se terminer à 350 Mo.

Or, le redimensionnement de la partition ne constitue qu'une partie du travail. Il vous faudra également redimensionner le système de fichiers qui y est installé. Pour les systèmes de fichiers ext2/3/4, cela se fait avec la commande `resize2fs`. Dans l'exemple ci-dessus, la partition 3 affiche toujours l'ancienne taille lorsqu'elle est montée :

```
$ df -h /dev/sdb3
Filesystem      Size  Used Avail Use% Mounted on
/dev/sdb3        88M  1.6M   80M   2% /media/carol/part3
```

Pour ajuster la taille, la commande `resize2fs DEVICE SIZE` peut être utilisée, où `DEVICE` correspond à la partition que vous voulez redimensionner, et `SIZE` à la nouvelle taille. Si vous ne précisez pas le paramètre de taille, l'espace disponible de la partition sera utilisé en totalité. Avant de redimensionner, il est préférable de démonter la partition.

Dans l'exemple ci-dessus :

```
$ sudo resize2fs /dev/sdb3
resize2fs 1.44.6 (5-Mar-2019)
Resizing the filesystem on /dev/sdb3 to 146212 (1k) blocks.
The filesystem on /dev/sdb3 is now 146212 (1k) blocks long.

$ df -h /dev/sdb3
Filesystem      Size  Used Avail Use% Mounted on
/dev/sdb3       135M  1.6M  123M   2% /media/carol/part3
```

Pour *réduire* la taille d'une partition, l'opération devra être effectuée dans l'ordre inverse. D'abord, vous réduisez le système de fichiers à la nouvelle taille, puis vous redimensionnez la partition elle-même à l'aide de `parted`.

WARNING Faites attention lorsque vous réduisez des partitions. Si vous vous trompez dans l'ordre, vous allez perdre des données !

Dans notre exemple :

```
# resize2fs /dev/sdb3 88m
resize2fs 1.44.6 (5-Mar-2019)
Resizing the filesystem on /dev/sdb3 to 90112 (1k) blocks.
The filesystem on /dev/sdb3 is now 90112 (1k) blocks long.

# parted /dev/sdb3
(parted) resizepart 3 300m
Warning: Shrinking a partition can cause data loss, are you sure
you want to continue?

Yes/No? y

(parted) print
Model: Kingston DataTraveler 2.0 (scsi)
Disk /dev/sdb: 1999MB
Sector size (logical/physical): 512B/512B
Partition Table: gpt
```

```
Disk Flags:

Number  Start   End     Size    File system  Name     Flags
 1      1049kB  99.6MB  98.6MB  ext4         primary
 2      99.6MB  200MB   100MB   ext4
 3      200MB   300MB   99.7MB  ext4         primary
```

TIP Au lieu de préciser une nouvelle taille, vous pouvez utiliser le paramètre `-M` de `resize2fs` pour ajuster la taille du système de fichiers de sorte qu'il soit juste assez grand pour les fichiers qu'il contient.

Créer des partitions d'échange (swap)

Sous Linux, le système peut délocaliser des pages de mémoire depuis la RAM vers le disque selon les besoins, en les stockant dans un espace séparé, généralement implémenté comme une partition distincte sur un disque, appelée *partition d'échange*, *partition swap* ou tout simplement *swap*. Cette partition doit être d'un type spécifique et configurée avec un outil approprié (`mkswap`) avant de pouvoir être utilisée.

Pour créer une partition swap à l'aide de `fdisk` ou `gdisk`, procédez comme pour une partition normale, comme nous l'avons vu plus haut. La seule différence, c'est que vous devrez changer le type de partition en *Linux swap*.

- Dans `fdisk`, utilisez la commande `t`. Sélectionnez la partition que vous voulez utiliser et modifiez son type en `82`. Écrivez les modifications sur le disque et quittez avec `w`.
- Dans `gdisk`, la commande pour changer le type de partition est également `t`, mais le code est `8200`. Écrivez les modifications sur le disque et quittez avec `w`.

Si vous utilisez `parted`, la partition doit être identifiée en tant que partition *swap* lors de sa création. Utilisez simplement `linux-swap` comme type de système de fichiers. Par exemple, la commande pour créer une partition d'échange de 500 Mo et qui commence à 300 Mo sur le disque est la suivante :

```
(parted) mkpart primary linux-swap 301m 800m
```

Une fois que la partition est créée et correctement identifiée, il suffit d'utiliser `mkswap` suivi du périphérique représentant la partition que vous voulez utiliser, comme ceci :

```
# mkswap /dev/sda2
```

Pour activer la *swap* sur cette partition, utilisez `swapon` suivi du nom du périphérique :

```
# swapon /dev/sda2
```

De même, `swapoff` suivi du nom du périphérique va désactiver la *swap* sur ce périphérique.

Linux supporte également l'utilisation de *fichiers swap* en dehors des partitions. Créez simplement un fichier vide de la taille que vous voulez en utilisant `dd` et utilisez ensuite `mkswap` et `swapon` avec ce fichier comme cible.

Les commandes ci-dessous vont créer un fichier de 1 Go appelé `myswap` et rempli de zéros dans le répertoire actuel, puis le configurer et l'activer en tant que fichier d'échange (*swap*).

Créez le fichier *swap* :

```
$ dd if=/dev/zero of=myswap bs=1M count=1024
1024+0 records in
1024+0 records out
1073741824 bytes (1.1 GB, 1.0 GiB) copied, 7.49254 s, 143 MB/s
```

`if=` est le fichier d'entrée, la source des données qui seront écrites dans le fichier. Dans ce cas, il s'agit du périphérique `/dev/zero` qui fournit autant de caractères `NULL` que nécessaire. `of=` est le fichier de sortie, le fichier qui sera créé. `bs=` correspond à la taille des blocs de données, spécifiée ici en mégaoctets, et `count=` correspond à la quantité de blocs à écrire sur la sortie. 1024 blocs de 1 Mo chacun correspondent à 1 Go.

```
# mkswap myswap
Setting up swapspace version 1, size = 1024 MiB (1073737728 bytes)
no label, UUID=49c53bc4-c4b1-4a8b-a613-8f42cb275b2b

# swapon myswap
```

Avec les commandes ci-dessus, ce fichier d'échange ne sera utilisé que durant la session système en cours. Si la machine est redémarrée, le fichier sera toujours disponible, mais il ne sera pas chargé automatiquement. Vous pouvez automatiser cette opération en ajoutant le nouveau fichier d'échange à `/etc/fstab`, ce que nous allons voir dans une leçon ultérieure.

TIP

`mkswap` et `swapon` vont protester si votre fichier *swap* ne possède pas les permissions adéquates. Les permissions saines pour ces fichiers sont `0600`, avec le propriétaire `root` et le groupe `root`.

Exercices guidés

1. Quel schéma de partitionnement doit-on utiliser pour partitionner un disque dur de 3 To en trois partitions de 1 Go ? Pourquoi ?

2. Avec `gdisk`, comment pouvons-nous savoir combien d'espace est disponible sur le disque ?

3. Quelle serait la commande pour créer un système de fichiers ext3, en vérifiant les blocs défectueux avant, avec le label `MyDisk` et un UUID aléatoire, sur le périphérique `/dev/sdc1` ?

4. En utilisant `parted`, quelle est la commande pour créer une partition ext4 de 300 Mo et qui commence à 500 Mo sur le disque ?

5. Imaginez que vous avez 2 partitions, l'une sur `/dev/sda1` et l'autre sur `/dev/sdb1`, toutes les deux d'une taille de 20 Go. Comment pouvez-vous les utiliser sur un seul système de fichiers Btrfs de manière à ce que le contenu d'une partition soit automatiquement dupliqué sur l'autre comme dans une configuration RAID1 ? Quelle sera alors la taille du système de fichiers ?

Exercices d'approfondissement

1. Prenez un disque de 2 Go avec une table de partition MBR et la configuration suivante :

```
Disk /dev/sdb: 1.9 GiB, 1998631936 bytes, 3903578 sectors
Disk model: DataTraveler 2.0
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x31a83a48

Device     Boot   Start     End Sectors  Size Id Type
/dev/sdb1          2048 1050623 1048576  512M 83 Linux
/dev/sdb3       2099200 3147775 1048576  512M 83 Linux
```

   Pouvez-vous créer une partition de 600 Mo sur ce disque ? Pourquoi ?

2. Sur un disque `/dev/sdc`, nous avons une première partition de 1 Go qui contient un système de fichiers ext4 avec 241 Mo de fichiers. En utilisant `parted`, comment pouvez-vous la réduire pour qu'elle ait juste assez d'espace pour les fichiers ?

3. Prenons l'exemple d'un disque `/dev/sdb`. Vous voulez créer une partition de 1 Go au début de ce disque. Donc, en utilisant `parted`, vous créez la partition avec `mkpart primary linux-swap 0 1024M`. Ensuite, vous activez la *swap* sur cette partition avec `swapon /dev/sdb1`, mais vous obtenez le message d'erreur suivant :

   `swapon: /dev/sdb1: read swap header failed`

   Que s'est-il passé ?

4. Au cours de cette leçon, vous avez testé quelques commandes dans `parted`, et vous avez supprimé par erreur la troisième partition de votre disque dur. Vous savez qu'elle venait après une partition EFI de 250 Mo et une partition *swap* de 4 Go, et qu'elle avait une taille de 10 Go. Quelle commande `parted` pouvez-vous utiliser pour la récupérer ?

5. Imaginez que vous avez une partition inutilisée de 4 Go sur `/dev/sda3`. En utilisant `fdisk`,

quelle serait la séquence d'opérations pour la transformer en une partition d'échange active ?

Résumé

Dans cette leçon, vous avez appris à :

- Créer une table de partitionnement MBR sur un disque avec `fdisk` et utiliser cet outil pour créer et supprimer des partitions.
- Créer une table de partitionnement GPT sur un disque avec `gdisk` et utiliser cet outil pour créer et supprimer des partitions.
- Créer des partitions ext2, ext3, ext4, XFS, VFAT et exFAT.
- Utiliser `parted` pour créer, supprimer et récupérer des partitions sur les disques MBR et GPT.
- Redimensionner des partitions ext2, ext3, ext4 et Btrfs.
- Créer, configurer et activer des partitions et des fichiers *swap*.

Les commandes suivantes ont été abordées dans cette leçon :

- `fdisk`
- `gdisk`
- `mkfs.ext2`, `mkfs.ext3`, `mkfs.ext4`, `mkfs.xfs`, `mkfs.vfat` et `mkfs.exfat`
- `parted`
- `btrfs`
- `mkswap`
- `swapon` et `swapoff`

Réponses aux exercices guidés

1. Quel schéma de partitionnement doit-on utiliser pour partitionner un disque dur de 3 To en trois partitions de 1 Go ? Pourquoi ?

   GPT, étant donné que le schéma MBR supporte au maximum les disques de 2 To.

2. Avec `gdisk`, comment pouvons-nous savoir combien d'espace est disponible sur le disque ?

   Utilisez `p` (print). L'espace libre total apparaîtra sur la dernière ligne d'infos avant la table de partitionnement elle-même.

3. Quelle serait la commande pour créer un système de fichiers ext3, en vérifiant les blocs défectueux avant, avec le label `MyDisk` et un UUID aléatoire, sur le périphérique `/dev/sdc1` ?

   La commande serait `mkfs.ext3 -c -L MyDisk -U random /dev/sdc1`. Alternativement, `mke2fs -t ext3` peut également être utilisé à la place de `mkfs.ext3`.

4. En utilisant `parted`, quelle est la commande pour créer une partition ext4 de 300 Mo et qui commence à 500 Mo sur le disque ?

   Utilisez `mkpart primary ext4 500m 800m`. Rappelez-vous que vous devrez créer le système de fichiers en utilisant `mkfs.ext4`, étant donné que `parted` ne le fait pas.

5. Imaginez que vous avez 2 partitions, l'une sur `/dev/sda1` et l'autre sur `/dev/sdb1`, toutes les deux d'une taille de 20 Go. Comment pouvez-vous les utiliser sur un seul système de fichiers Btrfs de manière à ce que le contenu d'une partition soit automatiquement dupliqué sur l'autre comme dans une configuration RAID1 ? Quelle sera alors la taille du système de fichiers ?

   Utilisez `mkfs.btrfs /dev/sda1 /dev/sdb1 -m raid1`. Le système de fichiers résultant aura une taille de 20 Go, étant donné qu'une partition ne fait que dupliquer l'autre.

Réponses aux exercices d'approfondissement

1. Prenez un disque de 2 Go avec une table de partition MBR et la configuration suivante :

```
Disk /dev/sdb: 1.9 GiB, 1998631936 bytes, 3903578 sectors
Disk model: DataTraveler 2.0
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x31a83a48

Device     Boot   Start     End Sectors  Size Id Type
/dev/sdb1          2048 1050623 1048576  512M 83 Linux
/dev/sdb3       2099200 3147775 1048576  512M 83 Linux
```

   Pouvez-vous créer une partition de 600 Mo sur ce disque ? Pourquoi ?

   Vous ne pouvez pas, étant donné qu'il n'y a pas assez d'espace contigu. Le premier indice que quelque chose ne va pas est la liste des périphériques : vous avez `/dev/sdb1` et `/dev/sdb3`, mais pas de `/dev/sdb2`. Il manque donc quelque chose.

   Ensuite, il faut regarder où se termine une partition et où commence l'autre. La partition 1 se termine au secteur `1050623`, et la partition 2 commence à `2099200`. Cela fait un "écart" de 1048577 secteurs. A 512 octets par secteur, cela fait 536.871.424 octets. Si vous divisez ce chiffre par 1024, vous obtenez 524,288 kilo-octets. Divisez à nouveau par 1024 et vous obtenez… 512 MO. C'est la taille de l'écart.

   Si le disque a une capacité de 2 Go, nous disposons au maximum de 512 Mo supplémentaires après la partition 3. Même si nous avons au total environ 1 Go non alloué, le plus grand bloc contigu est de 512 Mo. Il n'y a donc pas assez de place pour une partition de 600 Mo.

2. Sur un disque `/dev/sdc`, nous avons une première partition de 1 Go qui contient un système de fichiers ext4 avec 241 Mo de fichiers. En utilisant `parted`, comment pouvez-vous la réduire pour qu'elle ait juste assez d'espace pour les fichiers ?

   Il s'agit d'une opération en plusieurs étapes. Dans un premier temps, vous devez réduire le système de fichiers en utilisant `resize2fs`. Au lieu de spécifier directement la nouvelle taille, vous pouvez utiliser le paramètre `-M` pour qu'elle soit "juste assez grande". Voici ce que cela donne : `resize2fs -M /dev/sdc1`.

   Ensuite, vous redimensionnez la partition elle-même avec `parted` en utilisant `resizepart`. Vu

qu'il s'agit de la première partition, nous pouvons partir du principe qu'elle commence à zéro et qu'elle se termine à 241 Mo. La commande est donc `resizepart 1 241M`.

3. Prenons l'exemple d'un disque `/dev/sdb`. Vous voulez créer une partition de 1 Go au début de ce disque. Donc, en utilisant `parted`, vous créez la partition avec `mkpart primary linux-swap 0 1024M`. Ensuite, vous activez la *swap* sur cette partition avec `swapon /dev/sdb1`, mais vous obtenez le message d'erreur suivant :

   ```
   swapon: /dev/sdb1: read swap header failed
   ```

   Que s'est-il passé ?

   Vous avez créé une partition du type correct (`linux-swap`), mais rappelez-vous que `mkpart` *ne crée pas de système de fichiers*. Vous avez oublié de configurer la partition comme espace d'échange avec `mkswap` avant de l'utiliser.

4. Au cours de cette leçon, vous avez testé quelques commandes dans `parted`, et vous avez supprimé par erreur la troisième partition de votre disque dur. Vous savez qu'elle venait après une partition EFI de 250 Mo et une partition *swap* de 4 Go, et qu'elle avait une taille de 10 Go. Quelle commande `parted` pouvez-vous utiliser pour la récupérer ?

   Pas de panique, vous avez toutes les informations nécessaires pour récupérer la partition. Utilisez simplement `rescue` et faites le calcul. Vous aviez 250 Mo + 4.096 Mo (4*1024) avant, donc le point de départ devrait se situer autour de 4346 Mo. Plus 10.240 Mo (10*1024) en taille, cela devrait finir à 14.586 Mo. Donc, `rescue 4346m 14586m` devrait faire l'affaire. Il se peut que vous soyez obligé de donner un peu de "marge" à `rescue`, en commençant un peu plus tôt et en terminant un peu plus tard, en fonction de la géométrie de votre disque.

5. Imaginez que vous avez une partition inutilisée de 4 Go sur `/dev/sda3`. En utilisant `fdisk`, quelle serait la séquence d'opérations pour la transformer en une partition d'échange active ?

   Dans un premier temps, modifiez le type de partition en `Linux Swap` (82), écrivez les modifications sur le disque et quittez. Ensuite, utilisez `mkswap` pour configurer la partition comme une zone d'échange. Puis, utilisez `swapon` pour l'activer.

104.2 Maintenance de l'intégrité des systèmes de fichiers

Référence aux objectifs de LPI

LPIC-1 version 5.0, Exam 101, Objective 104.2

Valeur

2

Domaines de connaissance les plus importants

- Vérification de l'intégrité des systèmes de fichiers.
- Contrôle de l'espace et des inodes libres.
- Réparation de problèmes élémentaires sur les système de fichiers.

Liste partielle de termes, fichiers et utilitaires utilisés pour cet objectif

- `du`
- `df`
- `fsck`
- `e2fsck`
- `mke2fs`
- `tune2fs`
- `xfs_repair`
- `xfs_fsr`
- `xfs_db`

104.2 Leçon 1

Certification :	LPIC-1
Version :	5.0
Thème :	104 Disques, systèmes de fichiers Linux, arborescence de fichiers standard (FHS)
Objectif :	104.2 Maintenir l'intégrité des systèmes de fichiers
Leçon :	1 of 1

Introduction

Les systèmes de fichiers Linux modernes sont journalisés. Cela veut dire que chaque opération est enregistrée dans un journal interne avant d'être exécutée. Si l'opération est interrompue en raison d'une erreur du système (comme une panique du noyau -*kernel panic*-, une panne de courant, etc.), elle peut être reconstruite en inspectant le journal, ce qui permet d'éviter la dégradation du système de fichiers et la perte de données.

Cela réduit considérablement la nécessité d'effectuer des vérifications manuelles du système de fichiers, mais ces dernières peuvent toujours s'avérer nécessaires. La maîtrise des outils utilisés à cet effet (avec les paramètres correspondants) peut faire la différence entre un dîner en famille à la maison et une nuit blanche au travail dans la salle des serveurs.

Dans cette leçon, nous allons voir de plus près les outils disponibles pour surveiller l'utilisation du système de fichiers, optimiser le fonctionnement de ce dernier et voir comment détecter et réparer les éventuelles dégradations.

Vérifier l'utilisation du disque

Il existe deux commandes qui peuvent être utilisées pour vérifier l'espace utilisé et l'espace restant sur un système de fichiers. La première est `du`, qui signifie "*disk usage*" (utilisation du disque).

La commande `du` est récursive en soi. Dans sa forme la plus basique, la commande montre simplement combien de blocs d'un kilo-octet sont utilisés par le répertoire en cours et tous ses sous-répertoires :

```
$ du
4816    .
```

Cela ne nous aide pas beaucoup, et nous pouvons demander une sortie plus "humainement lisible" (*human readable*) en ajoutant le paramètre `-h` :

```
$ du -h
4.8M    .
```

Par défaut, `du` n'affiche que le décompte de l'utilisation des répertoires (en prenant en compte tous les fichiers et sous-répertoires qu'ils contiennent). Pour afficher un décompte individuel pour tous les fichiers du répertoire, vous pouvez utiliser le paramètre `-a` :

```
$ du -ah
432K    ./geminoid.jpg
508K    ./Linear_B_Hero.jpg
468K    ./LG-G8S-ThinQ-Mirror-White.jpg
656K    ./LG-G8S-ThinQ-Range.jpg
60K ./Stranger3_Titulo.png
108K    ./Baidu_Banho.jpg
324K    ./Xiaomi_Mimoji.png
284K    ./Mi_CC_9e.jpg
96K ./Mimoji_Comparativo.jpg
32K ./Xiaomi FCC.jpg
484K    ./geminoid2.jpg
108K    ./Mimoji_Abre.jpg
88K ./Mi8_Hero.jpg
832K    ./Tablet_Linear_B.jpg
332K    ./Mimoji_Comparativo.png
4.8M    .
```

Le comportement par défaut consiste à afficher l'utilisation de chaque sous-répertoire, puis l'utilisation totale du répertoire actuel, *y compris* les sous-répertoires :

```
$ du -h
4.8M    ./Temp
6.0M    .
```

Dans l'exemple ci-dessus, nous constatons que le sous-répertoire `Temp` occupe 4,8 Mo et que le répertoire courant, *y compris* `Temp`, occupe 6,0 Mo. Mais combien d'espace occupent les *fichiers* du répertoire courant, sans compter les sous-répertoires ? Pour cela, nous disposons du paramètre `-S` :

```
$ du -Sh
4.8M    ./Temp
1.3M    .
```

TIP Gardez à l'esprit que les paramètres en ligne de commande sont sensibles à la casse : `-s` n'est pas la même chose que `-S`.

Si vous voulez conserver cette distinction entre l'espace utilisé par les fichiers du répertoire courant et l'espace utilisé par les sous-répertoires, tout en gardant un total général à la fin, vous pouvez ajouter le paramètre `-c` :

```
$ du -Shc
4.8M    ./Temp
1.3M    .
6.0M    total
```

Vous pouvez régler la "profondeur" des résultats de `du` avec le paramètre `-d N`, où `N` indique les niveaux. Par exemple, si vous utilisez le paramètre `-d 1`, il affichera le répertoire courant et ses sous-répertoires, mais pas les sous-répertoires de ces derniers.

Voyez la différence ci-dessous. Sans `-d` :

```
$ du -h
216K    ./somedir/anotherdir
224K    ./somedir
232K    .
```

Et en limitant la profondeur à un seul niveau avec `-d 1` :

```
$ du -h -d1
224K    ./somedir
232K    .
```

Notez que même si `anotherdir` n'est pas affiché, sa taille sera toujours prise en compte.

Il peut être souhaitable d'exclure certains types de fichiers du décompte, ce que vous pouvez effectuer avec `--exclude="PATTERN"`, où `PATTERN` correspond au motif que vous voulez faire correspondre. Prenons l'exemple de ce répertoire :

```
$ du -ah
124K    ./ASM68K.EXE
2.0M    ./Contra.bin
36K ./fixheadr.exe
4.0K    ./README.txt
2.1M    ./Contra_NEW.bin
4.0K    ./Built.bat
8.0K    ./Contra_Main.asm
4.2M    .
```

Maintenant, nous allons utiliser `--exclude` pour éliminer tous les fichiers avec l'extension `.bin` :

```
$ du -ah --exclude="*.bin"
124K    ./ASM68K.EXE
36K ./fixheadr.exe
4.0K    ./README.txt
4.0K    ./Built.bat
8.0K    ./Contra_Main.asm
180K    .
```

Notez que le total ne prend plus en compte la taille des fichiers exclus.

Vérifier l'espace disponible

`du` fonctionne au niveau des fichiers. Il existe une autre commande qui peut vous renseigner sur l'utilisation du disque et sur l'espace disponible au niveau du système de fichiers. C'est la commande `df`.

La commande `df` fournit une liste de tous les systèmes de fichiers disponibles (déjà montés) sur votre système, avec leur taille totale, l'espace utilisé, l'espace disponible, le pourcentage d'utilisation et le point de montage :

```
$ df
Filesystem     1K-blocks      Used Available Use% Mounted on
udev             2943068         0   2943068   0% /dev
tmpfs             595892      2496    593396   1% /run
/dev/sda1      110722904  25600600  79454800  25% /
tmpfs            2979440    951208   2028232  32% /dev/shm
tmpfs               5120         0      5120   0% /run/lock
tmpfs            2979440         0   2979440   0% /sys/fs/cgroup
tmpfs             595888        24    595864   1% /run/user/119
tmpfs             595888       116    595772   1% /run/user/1000
/dev/sdb1          89111      1550     80824   2% /media/carol/part1
/dev/sdb3          83187      1550     75330   3% /media/carol/part3
/dev/sdb2          90827      1921     82045   3% /media/carol/part2
/dev/sdc1      312570036 233740356  78829680  75% /media/carol/Samsung Externo
```

En revanche, l'affichage de la taille par blocs de 1 ko n'est pas très convivial. Tout comme pour `du`, vous pouvez ajouter le paramètre `-h` pour obtenir une sortie plus "humainement lisible" :

```
$ df -h
Filesystem      Size  Used Avail Use% Mounted on
udev            2.9G     0  2.9G   0% /dev
tmpfs           582M  2.5M  580M   1% /run
/dev/sda1       106G   25G   76G  25% /
tmpfs           2.9G  930M  2.0G  32% /dev/shm
tmpfs           5.0M     0  5.0M   0% /run/lock
tmpfs           2.9G     0  2.9G   0% /sys/fs/cgroup
tmpfs           582M   24K  582M   1% /run/user/119
tmpfs           582M  116K  582M   1% /run/user/1000
/dev/sdb1        88M  1.6M   79M   2% /media/carol/part1
/dev/sdb3        82M  1.6M   74M   3% /media/carol/part3
/dev/sdb2        89M  1.9M   81M   3% /media/carol/part2
/dev/sdc1       299G  223G   76G  75% /media/carol/Samsung Externo
```

Vous pouvez également utiliser le paramètre `-i` pour afficher les inodes utilisés/disponibles plutôt que les blocs :

```
$ df -i
```

```
Filesystem      Inodes  IUsed   IFree IUse% Mounted on
udev            737142    547  736595    1% /dev
tmpfs           745218    908  744310    1% /run
/dev/sda6      6766592 307153 6459439    5% /
tmpfs           745218    215  745003    1% /dev/shm
tmpfs           745218      4  745214    1% /run/lock
tmpfs           745218     18  745200    1% /sys/fs/cgroup
/dev/sda1        62464    355   62109    1% /boot
tmpfs           745218     43  745175    1% /run/user/1000
```

Un paramètre intéressant est `-T`, qui affichera en outre le type de chacun des systèmes de fichiers :

```
$ df -hT
Filesystem     Type      Size  Used Avail Use% Mounted on
udev           devtmpfs  2.9G     0  2.9G   0% /dev
tmpfs          tmpfs     582M  2.5M  580M   1% /run
/dev/sda1      ext4      106G   25G   76G  25% /
tmpfs          tmpfs     2.9G  930M  2.0G  32% /dev/shm
tmpfs          tmpfs     5.0M     0  5.0M   0% /run/lock
tmpfs          tmpfs     2.9G     0  2.9G   0% /sys/fs/cgroup
tmpfs          tmpfs     582M   24K  582M   1% /run/user/119
tmpfs          tmpfs     582M  116K  582M   1% /run/user/1000
/dev/sdb1      ext4       88M  1.6M   79M   2% /media/carol/part1
/dev/sdb3      ext4       82M  1.6M   74M   3% /media/carol/part3
/dev/sdb2      ext4       89M  1.9M   81M   3% /media/carol/part2
/dev/sdc1      fuseblk   299G  223G   76G  75% /media/carol/Samsung Externo
```

À partir du moment où vous connaissez le type du système de fichiers, vous pouvez filtrer les résultats. Vous pouvez afficher uniquement les systèmes de fichiers d'un type donné avec `-t TYPE` ou exclure les systèmes de fichiers d'un type donné avec `-x TYPE`, comme dans les exemples ci-dessous.

Exclure les systèmes de fichiers `tmpfs` :

```
$ df -hx tmpfs
Filesystem      Size  Used Avail Use% Mounted on
udev            2.9G     0  2.9G   0% /dev
/dev/sda1       106G   25G   76G  25% /
/dev/sdb1        88M  1.6M   79M   2% /media/carol/part1
/dev/sdb3        82M  1.6M   74M   3% /media/carol/part3
/dev/sdb2        89M  1.9M   81M   3% /media/carol/part2
```

```
/dev/sdc1       299G  223G   76G  75% /media/carol/Samsung Externo
```

Afficher uniquement les systèmes de fichiers `ext4` :

```
$ df -ht ext4
Filesystem      Size  Used Avail Use% Mounted on
/dev/sda1       106G   25G   76G  25% /
/dev/sdb1        88M  1.6M   79M   2% /media/carol/part1
/dev/sdb3        82M  1.6M   74M   3% /media/carol/part3
/dev/sdb2        89M  1.9M   81M   3% /media/carol/part2
```

Vous pouvez également personnaliser l'affichage de `df` en sélectionnant les éléments à afficher et dans quel ordre, grâce au paramètre `--output=` suivi d'une liste de champs à afficher, séparés par des virgules. Voici quelques champs disponibles :

`source`
: Le périphérique qui correspond au système de fichiers.

`fstype`
: Le type de système de fichiers.

`size`
: La taille totale du système de fichiers.

`used`
: L'espace utilisé.

`avail`
: L'espace disponible.

`pcent`
: Le pourcentage d'utilisation.

`target`
: L'endroit où le système de fichiers est monté (point de montage).

Si vous souhaitez obtenir un résultat avec la cible, la source, le type et l'utilisation, vous pouvez utiliser :

```
$ df -h --output=target,source,fstype,pcent
```

```
Mounted on                    Filesystem     Type     Use%
/dev                          udev           devtmpfs   0%
/run                          tmpfs          tmpfs      1%
/                             /dev/sda1      ext4      25%
/dev/shm                      tmpfs          tmpfs     32%
/run/lock                     tmpfs          tmpfs      0%
/sys/fs/cgroup                tmpfs          tmpfs      0%
/run/user/119                 tmpfs          tmpfs      1%
/run/user/1000                tmpfs          tmpfs      1%
/media/carol/part1            /dev/sdb1      ext4       2%
/media/carol/part3            /dev/sdb3      ext4       3%
/media/carol/part2            /dev/sdb2      ext4       3%
/media/carol/Samsung Externo  /dev/sdc1      fuseblk   75%
```

`df` peut également être utilisé pour vérifier les informations sur les inodes, en passant les champs suivants à `--output=` :

`itotal`
: Nombre total d'inodes du système de fichiers.

`iused`
: Nombre d'inodes utilisés dans le système de fichiers.

`iavail`
: Nombre d'inodes disponibles dans le système de fichiers.

`ipcent`
: Pourcentage d'inodes utilisés dans le système de fichiers.

Par exemple :

```
$ df --output=source,fstype,itotal,iused,ipcent
Filesystem     Type      Inodes  IUsed IUse%
udev           devtmpfs  735764    593    1%
tmpfs          tmpfs     744858   1048    1%
/dev/sda1      ext4     7069696 318651    5%
tmpfs          tmpfs     744858    222    1%
tmpfs          tmpfs     744858      3    1%
tmpfs          tmpfs     744858     18    1%
tmpfs          tmpfs     744858     22    1%
tmpfs          tmpfs     744858     40    1%
```

Maintenir les systèmes de fichiers ext2, ext3 et ext4

Pour analyser l'intégrité d'un système de fichiers (et corriger les erreurs éventuelles si tout se passe bien), Linux fournit l'utilitaire `fsck` (pensez à "*filesystem check*", ce qui vous aidera à retenir le nom). Dans sa forme la plus sommaire, vous pouvez l'invoquer avec `fsck` suivi de l'emplacement du système de fichiers que vous voulez vérifier :

```
# fsck /dev/sdb1
fsck from util-linux 2.33.1
e2fsck 1.44.6 (5-Mar-2019)
DT_2GB: clean, 20/121920 files, 369880/487680 blocks
```

WARNING Ne lancez JAMAIS `fsck` (ou les outils équivalents) sur un système de fichiers monté. Si vous le faites quand même, vous risquez de perdre vos données.

Ce n'est pas `fsck` en soi qui va vérifier l'intégrité du système de fichiers. Il va plutôt faire appel à l'outil approprié en fonction du type de système de fichiers pour le faire. Dans l'exemple ci-dessus, aucun type de système de fichiers n'a été spécifié. `fsck` a donc présumé un système de fichiers ext2/3/4 par défaut et a fait appel à `e2fsck`.

Pour spécifier un système de fichiers, utilisez l'option `-t` suivie du nom du système de fichiers, comme dans `fsck -t vfat /dev/sdc`. Alternativement, vous pouvez invoquer directement l'outil spécifique au système de fichiers, comme `fsck.msdos` pour les systèmes de fichiers FAT.

TIP Tapez `fsck` et deux fois la touche `Tab` pour afficher la liste de toutes les commandes associées sur votre système.

`fsck` accepte une série d'arguments en ligne de commande. Voici les plus courants :

`-A`
: Vérifie l'intégrité de tous les systèmes de fichiers répertoriés dans `/etc/fstab`.

`-C`
: Affiche une barre de progression lors de la vérification d'un système de fichiers. Fonctionne uniquement sur les systèmes de fichiers ext2/3/4 pour l'instant.

`-N`
: Affiche ce qui serait fait et quitte le programme sans réellement vérifier le système de fichiers.

`-R`
: Utilisé conjointement avec `-A`, ce paramètre fait l'impasse sur la vérification du système de

fichiers racine.

`-V`

Mode verbeux : affiche plus d'informations que d'habitude pendant l'opération. Ce mode est utile pour le débogage.

L'outil spécifique pour les systèmes de fichiers ext2, ext3 et ext4 est `e2fsck`, également appelé `fsck.ext2`, `fsck.ext3` et `fsck.ext4` (ces trois derniers ne sont que des liens vers `e2fsck`). Par défaut, il fonctionne en mode interactif : dès qu'une erreur est détectée dans le système de fichiers, le programme s'arrête et demande à l'utilisateur ce qu'il doit faire. L'utilisateur doit taper `y` pour corriger le problème, `n` pour ne pas le corriger ou `a` pour corriger le problème actuel et tous les problèmes subséquents.

Évidemment, rester devant un terminal en attendant que `e2fsck` vous demande ce qu'il faut faire n'est pas vraiment une utilisation productive de votre temps, surtout si vous avez affaire à un système de fichiers conséquent. Il existe donc une série d'options qui permettent à `e2fsck` de s'exécuter en mode non-interactif :

`-p`

Cette option va tenter de corriger automatiquement toutes les erreurs trouvées. Si une erreur nécessitant l'intervention de l'administrateur système est détectée, `e2fsck` va fournir un rapport d'analyse du problème et terminer l'exécution.

`-y`

Ceci va répondre `y` (oui) à toutes les questions.

`-n`

Le contraire de `-y`. En dehors de la réponse `n` (non) à toutes les questions, ceci aura pour effet de monter le système de fichiers en lecture seule, de manière à ce qu'il ne puisse pas être modifié.

`-f`

Force `e2fsck` à vérifier un système de fichiers même s'il est identifié comme "propre", c'est-à-dire s'il a été correctement démonté.

Peaufinage d'un système de fichiers ext

Les systèmes de fichiers ext2, ext3 et ext4 ont un certain nombre de paramètres qui peuvent être peaufinés - ou faire l'objet d'un "*tuning*" - par l'administrateur du système pour mieux répondre aux besoins du système. L'outil qui sert à afficher ou modifier ces paramètres s'appelle `tune2fs`.

Pour afficher les paramètres en cours pour un système de fichiers donné, utilisez le paramètre `-l` suivi du périphérique correspondant à la partition. L'exemple ci-dessous montre la sortie de cette commande sur la première partition du premier disque (`/dev/sda1`) d'une machine :

```
# tune2fs -l /dev/sda1
tune2fs 1.44.6 (5-Mar-2019)
Filesystem volume name:   <none>
Last mounted on:          /
Filesystem UUID:          6e2c12e3-472d-4bac-a257-c49ac07f3761
Filesystem magic number:  0xEF53
Filesystem revision #:    1 (dynamic)
Filesystem features:      has_journal ext_attr resize_inode dir_index filetype
needs_recovery extent 64bit flex_bg sparse_super large_file huge_file dir_nlink extra_isize
metadata_csum
Filesystem flags:         signed_directory_hash
Default mount options:    user_xattr acl
Filesystem state:         clean
Errors behavior:          Continue
Filesystem OS type:       Linux
Inode count:              7069696
Block count:              28255605
Reserved block count:     1412780
Free blocks:              23007462
Free inodes:              6801648
First block:              0
Block size:               4096
Fragment size:            4096
Group descriptor size:    64
Reserved GDT blocks:      1024
Blocks per group:         32768
Fragments per group:      32768
Inodes per group:         8192
Inode blocks per group:   512
Flex block group size:    16
Filesystem created:       Mon Jun 17 13:49:59 2019
Last mount time:          Fri Jun 28 21:14:38 2019
Last write time:          Mon Jun 17 13:53:39 2019
Mount count:              8
Maximum mount count:      -1
Last checked:             Mon Jun 17 13:49:59 2019
Check interval:           0 (<none>)
Lifetime writes:          20 GB
Reserved blocks uid:      0 (user root)
```

```
Reserved blocks gid:      0 (group root)
First inode:              11
Inode size:           256
Required extra isize:     32
Desired extra isize:      32
Journal inode:            8
First orphan inode:       5117383
Default directory hash:   half_md4
Directory Hash Seed:      fa95a22a-a119-4667-a73e-78f77af6172f
Journal backup:           inode blocks
Checksum type:            crc32c
Checksum:                 0xe084fe23
```

Les systèmes de fichiers ext disposent d'un *compteur de montages* (*mount counts*). Ce compteur est incrémenté à chaque montage du système de fichiers. Lorsqu'une valeur seuil (le *maximum mount count*) est atteinte, l'intégrité du système sera automatiquement vérifiée avec `e2fsck` au prochain démarrage.

Le maximum de montages peut être défini avec le paramètre `-c N`, où `N` correspond au nombre de fois que le système de fichiers peut être monté sans être inspecté. Le paramètre `-C N` fixe le nombre de fois où le système a été monté à la valeur de `N`. Notez bien que les paramètres en ligne de commande sont sensibles à la casse et que `-c` n'est pas la même chose que `-C`.

Vous pouvez également définir un laps de temps entre les vérifications avec le paramètre `-i` suivi d'un nombre et les lettres `d` (*days*) pour les jours, `m` (*months*) pour les mois et `y` (*years*) pour les années. Par exemple, `-i 10d` vérifierait l'intégrité du système de fichiers au prochain redémarrage tous les dix jours. La valeur zéro permet de désactiver cette fonctionnalité.

`-L` peut être utilisé pour étiqueter le système de fichiers. Cette étiquette peut comporter jusqu'à seize caractères. Le paramètre `-U` définit l'UUID du système de fichiers, un nombre hexadécimal de 128 bits. Dans l'exemple ci-dessus, l'UUID est `6e2c12e3-472d-4bac-a257-c49ac07f3761`. L'étiquette et l'UUID peuvent être utilisés à la place du nom du périphérique (comme `/dev/sda1`) pour monter le système de fichiers.

L'option `-e BEHAVIOUR` définit le comportement (*behaviour*) du noyau lorsqu'une erreur est détectée dans le système de fichiers. On distingue trois comportements possibles :

`continue`

Continue l'exécution normalement.

`remount-ro`

Remonte le système de fichiers en lecture seule.

`panic`

Déclenche une panique du noyau (*kernel panic*).

Le comportement par défaut est `continue`. `remount-ro` (remontage en lecture seule) peut s'avérer utile pour les applications critiques, étant donné que les écritures sur le disque sont immédiatement interrompues, ce qui permet d'éviter d'autres erreurs par la suite.

Les systèmes de fichiers ext3 sont essentiellement des systèmes de fichiers ext2 dotés d'un journal. En utilisant `tune2fs` vous pouvez ajouter un journal à un système de fichiers ext2 et le convertir ainsi en ext3. La procédure est simple, il suffit de passer le paramètre `-j` à `tune2fs`, suivi du périphérique qui contient le système de fichiers :

```
# tune2fs -j /dev/sda1
```

Par la suite, lorsque vous montez le système de fichiers converti, n'oubliez pas de spécifier le type `ext3` pour utiliser le journal.

Lorsque vous avez affaire à des systèmes de fichiers journalisés, le paramètre `-J` vous permet d'utiliser des paramètres supplémentaires pour définir une série d'options, comme `-J size=` pour fixer la taille du journal (en mégaoctets), `-J location=` pour spécifier l'endroit où le journal doit être stocké (soit un bloc spécifique, soit une position spécifique sur le disque avec des suffixes comme `M` ou `G`) et même installer le journal sur un périphérique externe avec `-J device=`.

Vous pouvez spécifier plusieurs paramètres à la fois en les séparant par une virgule. Par exemple : `-J size=10,location=100M,device=/dev/sdb1` va créer un journal de 10 Mo à la position 100 Mo sur le périphérique `/dev/sdb1`.

WARNING

`tune2fs` dispose d'une option "force brute", `-f`, qui le forcera à exécuter une opération même en cas d'erreur. Il va sans dire que cette option ne doit être utilisée qu'avec une précaution extrême.

Maintenir les systèmes de fichiers XFS

L'équivalent de `fsck` pour les systèmes de fichiers XFS est `xfs_repair`. Si vous soupçonnez que quelque chose ne tourne pas rond avec le système de fichiers, la première chose à faire est de vérifier s'il n'est pas dégradé.

Pour ce faire, passez le paramètre `-n` à `xfs_repair`, suivi du périphérique qui contient le système de fichiers. Ce paramètre `-n` signifie "pas de modification" (*no modify*) : le système de fichiers sera analysé, les erreurs seront signalées mais aucune correction ne sera effectuée :

```
# xfs_repair -n /dev/sdb1
Phase 1 - find and verify superblock...
Phase 2 - using internal log
        - zero log...
        - scan filesystem freespace and inode maps...
        - found root inode chunk
Phase 3 - for each AG...
        - scan (but do not clear) agi unlinked lists...
        - process known inodes and perform inode discovery...
        - agno = 0
        - agno = 1
        - agno = 2
        - agno = 3
        - process newly discovered inodes...
Phase 4 - check for duplicate blocks...
        - setting up duplicate extent list...
        - check for inodes claiming duplicate blocks...
        - agno = 1
        - agno = 3
        - agno = 0
        - agno = 2
No modify flag set, skipping phase 5
Phase 6 - check inode connectivity...
        - traversing filesystem ...
        - traversal finished ...
        - moving disconnected inodes to lost+found ...
Phase 7 - verify link counts...
No modify flag set, skipping filesystem flush and exiting.
```

En cas d'erreur, vous pouvez procéder à la réparation en omettant le paramètre `-n`, comme ceci : `xfs_repair /dev/sdb1`.

`xfs_repair` accepte un certain nombre d'options en ligne de commande. En voici quelques-unes :

`-l LOGDEV` et **`-r RTDEV`**

Nécessaires si le système de fichiers est doté de sections *log* (journalisation) et *realtime* (temps réel) externes. Dans ce cas, remplacez `LOGDEV` et `RTDEV` par les périphériques correspondants.

`-m N`

Sert à limiter l'utilisation de la mémoire de `xfs_repair` à `N` mégaoctets, ce qui peut être utile sur les serveurs. Selon la page de manuel, par défaut, `xfs_repair` adaptera son utilisation de la mémoire en fonction des besoins, jusqu'à 75% de la RAM physique du système.

`-d`

Le mode "dangereux" (*dangerous*) permet de réparer les systèmes de fichiers montés en lecture seule.

`-v`

Vous l'aurez probablement deviné : il s'agit du mode verbeux (*verbose*). La "verbosité" est augmentée à chaque utilisation. (Par exemple, `-v -v` va afficher plus d'informations qu'un simple `-v`).

Notez que `xfs_repair` est incapable de réparer les systèmes de fichiers avec un journal "corrompu". Vous pouvez "éliminer" un journal dégradé avec le paramètre `-L`, mais gardez à l'esprit qu'il s'agit d'un *dernier recours* dans la mesure où cette opération peut entraîner la dégradation du système de fichiers et la perte des données.

Pour déboguer un système de fichiers XFS, vous pouvez utiliser l'outil `xfs_db`, comme dans `xfs_db /dev/sdb1`. Il est utilisé principalement pour analyser les différents éléments et paramètres du système de fichiers.

L'outil dispose d'une invite interactive, un peu comme `parted`, avec toute une série de commandes internes. Un système d'aide est également disponible : tapez `help` pour afficher la liste de toutes les commandes, et `help` suivi du nom de la commande pour obtenir plus d'informations sur la commande en question.

Un autre outil utile est `xfs_fsr`, qui peut être utilisé pour réorganiser (ou "défragmenter") un système de fichiers XFS. Lorsqu'il est exécuté sans arguments supplémentaires, il va tourner pendant deux heures et tenter de défragmenter tous les systèmes de fichiers XFS montés et accessibles en écriture qui sont listés dans le fichier `/etc/mtab/`. Vous devrez éventuellement installer cet outil à l'aide du gestionnaire de paquets de votre distribution Linux, étant donné qu'il ne fait peut-être pas partie de l'installation par défaut. Pour plus d'informations, consultez la page de manuel correspondante.

Exercices guidés

1. En utilisant `du`, comment pouvons-nous vérifier l'espace utilisé par les fichiers du répertoire courant ?

2. En utilisant `df`, affichez les informations pour chaque système de fichiers ext4, avec les résultats qui incluent les informations suivantes, dans l'ordre : périphérique, point de montage, nombre total d'inodes, nombre d'inodes disponibles, pourcentage d'espace libre.

3. Quelle est la commande pour exécuter `e2fsck` sur `/dev/sdc1` en mode non-interactif, tout en essayant de corriger automatiquement la plupart des erreurs ?

4. Admettons que `/dev/sdb1` est un système de fichiers ext2. Comment pouvez-vous le convertir en ext3 et en même temps réinitialiser son décompte de montage et modifier son étiquette en `UserData` ?

5. Comment pouvez-vous vérifier l'intégrité d'un système de fichiers XFS *sans* pour autant réparer les éventuelles erreurs qu'il contient ?

Exercices d'approfondissement

1. Prenons l'exemple d'un système de fichiers ext4 sur `/dev/sda1` avec les paramètres suivants, obtenus à l'aide de `tune2fs` :

   ```
   Mount count:              8
   Maximum mount count:      -1
   ```

   Que se passera-t-il au prochain démarrage si la commande `tune2fs -c 9 /dev/sda1` est invoquée ?

2. Considérez le résultat suivant de `du -h`:

   ```
   $ du -h
   216K    ./somedir/anotherdir
   224K    ./somedir
   232K    .
   ```

   Quel est l'espace occupé par les seuls fichiers du répertoire courant ? Comment pourrions-nous reformuler la commande pour faire apparaître cette information plus clairement ?

3. Que se passera-t-il avec le système de fichiers ext2 `/dev/sdb1` si la commande ci-dessous est invoquée ?

   ```
   # tune2fs -j /dev/sdb1 -J device=/dev/sdc1 -i 30d
   ```

4. Comment pouvons-nous vérifier les erreurs sur un système de fichiers XFS sur `/dev/sda1` doté d'une section de journalisation sur `/dev/sdc1`, mais sans effectuer de réparations ?

5. Quelle est la différence entre les paramètres `-T` et `-t` pour `df` ?

Résumé

Dans cette leçon, vous avez appris à :

- Vérifier l'espace utilisé et l'espace disponible dans un système de fichiers.
- Adapter l'affichage de `df` à vos besoins.
- Vérifier l'intégrité d'un système de fichiers et réparer les erreurs avec `fsck` et `e2fsck`.
- Peaufiner un système de fichiers avec `tune2fs`.
- Vérifier l'intégrité d'un système de fichiers XFS et réparer les erreurs avec `xfs_repair`.

Les commandes suivantes ont été abordées dans cette leçon :

`du`
: Afficher la quantité d'espace disque utilisée dans un système de fichiers.

`df`
: Afficher la quantité d'espace disque disponible (*free*) dans un système de fichiers.

`fsck`
: L'outil d'analyse et de réparation des systèmes de fichiers.

`e2fsck`
: L'outil d'analyse et de réparation spécifique aux systèmes de fichiers ext2/3/4.

`tune2fs`
: Modifie les paramètres des systèmes de fichiers ext2/3/4.

`xfs_repair`
: L'équivalent de `fsck` pour les systèmes de fichiers XFS.

`xfs_db`
: Cet outil sert à afficher les paramètres divers et variés d'un système de fichiers XFS.

Réponses aux exercices guidés

1. En utilisant `du`, comment pouvons-nous vérifier l'espace utilisé par les fichiers du répertoire courant ?

   Dans un premier temps, utilisez le paramètre `-S` pour séparer l'affichage du répertoire courant des sous-répertoires. Ensuite, utilisez `-d 0` pour limiter la profondeur de l'affichage à zéro, ce qui signifie "pas de sous-répertoires". N'oubliez pas `-h` pour obtenir une sortie dans un format "humainement lisible" :

   ```
   $ du -S -h -d 0
   ```

   ou

   ```
   $ du -Shd 0
   ```

2. En utilisant `df`, affichez les informations pour chaque système de fichiers ext4, avec les résultats qui incluent les informations suivantes, dans l'ordre : périphérique, point de montage, nombre total d'inodes, nombre d'inodes disponibles, pourcentage d'espace libre.

   Vous pouvez filtrer les systèmes de fichiers avec l'option `-t` suivie du nom du système de fichiers. Pour obtenir les infos demandées, utilisez `--output=source,target,itotal,iavail,pcent`. La réponse est donc :

   ```
   $ df -t ext4 --output=source,target,itotal,iavail,pcent
   ```

3. Quelle est la commande pour exécuter `e2fsck` sur `/dev/sdc1` en mode non-interactif, tout en essayant de corriger automatiquement la plupart des erreurs ?

   Le paramètre qui permet d'essayer de corriger automatiquement la plupart des erreurs est `-p`. La réponse est donc :

   ```
   # e2fsck -p /dev/sdc1
   ```

4. Admettons que `/dev/sdb1` est un système de fichiers ext2. Comment pouvez-vous le convertir en ext3 et en même temps réinitialiser son décompte de montage et modifier son étiquette en `UserData` ?

   Rappelez-vous que la conversion d'un système de fichiers ext2 en ext3 revient à ajouter un

journal, ce qui peut être fait avec le paramètre `-j`. Pour réinitialiser le décompte de montages, utilisez `-C 0`. Pour changer l'étiquette, utilisez `-L UserData`. La bonne réponse est donc :

```
# tune2fs -j -C 0 -L UserData /dev/sdb1
```

5. Comment pouvez-vous vérifier l'intégrité d'un système de fichiers XFS *sans* pour autant réparer les éventuelles erreurs qu'il contient ?

   Utilisez le paramètre `-n`, comme dans `xfs -n`, suivi du périphérique correspondant.

Réponses aux exercices d'approfondissement

1. Prenons l'exemple d'un système de fichiers ext4 sur `/dev/sda1` avec les paramètres suivants, obtenus à l'aide de `tune2fs` :

   ```
   Mount count:              8
   Maximum mount count:      -1
   ```

   Que se passera-t-il au prochain démarrage si la commande `tune2fs -c 9 /dev/sda1` est invoquée ?

   La commande va fixer le nombre maximum de montages pour le système de fichiers à 9. Puisque ce nombre est actuellement égal à 8, le prochain démarrage du système va entraîner une vérification de l'intégrité du système de fichiers.

2. Considérez le résultat suivant de `du -h`:

   ```
   $ du -h
   216K    ./somedir/anotherdir
   224K    ./somedir
   232K    .
   ```

   Quel est l'espace occupé par les seuls fichiers du répertoire courant ? Comment pourrions-nous reformuler la commande pour faire apparaître cette information plus clairement ?

   Sur un total de 232 Ko utilisés, 224 Ko sont utilisés par le sous-répertoire `somedir` et ses sous-répertoires. Donc, en excluant ces derniers, nous avons 8 Ko occupés par les fichiers du répertoire courant. Cette information peut être affichée plus clairement en utilisant le paramètre `-S`, qui va dissocier les répertoires dans le décompte.

3. Que se passera-t-il avec le système de fichiers ext2 `/dev/sdb1` si la commande ci-dessous est invoquée ?

   ```
   # tune2fs -j /dev/sdb1 -J device=/dev/sdc1 -i 30d
   ```

   Un journal sera ajouté à `/dev/sdb1`, ce qui le convertira en ext3. Le journal sera stocké sur le périphérique `/dev/sdc1` et le système de fichiers sera vérifié tous les 30 jours.

4. Comment pouvons-nous vérifier les erreurs sur un système de fichiers XFS sur `/dev/sda1` doté d'une section de journalisation sur `/dev/sdc1`, mais sans effectuer de réparations ?

Utilisez `xfs_repair` suivi de `-l /dev/sdc1` pour indiquer le périphérique contenant la section de journalisation et `-n` pour éviter les modifications.

```
# xfs_repair -l /dev/sdc1 -n
```

5. Quelle est la différence entre les paramètres `-T` et `-t` pour `df` ?

   Le paramètre `-T` va inclure le type de chaque système de fichiers dans l'affichage de `df`. `-t` est un filtre, et ne montrera que les systèmes de fichiers du type donné dans l'affichage, en excluant tous les autres.

104.3 Montage et démontage des systèmes de fichiers

Référence aux objectifs de LPI

LPIC-1 version 5.0, Exam 101, Objective 104.3

Valeur

3

Domaines de connaissance les plus importants

- Montage et démontage manuel des systèmes de fichiers.
- Configuration du montage des systèmes de fichiers au démarrage du système.
- Configuration des options de montage des systèmes de fichiers.
- Utilisation des étiquettes et UUID pour l'identification et le montage des systèmes de fichier
- Connaissance de base des unités de montage systemd mount units (systemd mount units).

Liste partielle de termes, fichiers et utilitaires utilisés pour cet objectif

- `/etc/fstab`
- `/media/`
- `mount`
- `umount`
- `blkid`
- `lsblk`

Linux Professional Institute

104.3 Leçon 1

Certification :	LPIC-1
Version :	5.0
Thème :	104 Disques, systèmes de fichiers Linux, arborescence de fichiers standard (FHS)
Objectif :	104.3 Gérer le montage et le démontage des systèmes de fichiers
Leçon :	1 sur 1

Introduction

À ce stade, vous avez appris à partitionner les disques et à créer et maintenir des systèmes de fichiers sur ces derniers. En revanche, avant de pouvoir accéder à un système de fichiers sous Linux, il faut que celui-ci soit *monté*.

Cette opération consiste à rattacher le système de fichiers à un point spécifique de l'arborescence de votre système, que l'on appelle "point de montage". Les systèmes de fichiers peuvent être montés à la main ou de manière automatique, et il existe une multitude de façons de le faire. Nous allons en découvrir quelques-unes dans cette leçon.

Monter et démonter des systèmes de fichiers

La commande pour monter manuellement un système de fichiers s'appelle `mount` et sa syntaxe est :

```
mount -t TYPE DEVICE MOUNTPOINT
```

Avec :

`TYPE`
: Le type de système de fichiers en cours de montage (par exemple ext4, btrfs, exfat, etc.).

`DEVICE`
: Le nom de la partition qui contient le système de fichiers (par exemple `/dev/sdb1`).

`MOUNTPOINT`
: L'endroit où le système de fichiers sera monté. Le répertoire monté ne doit pas nécessairement être vide, mais il doit exister. En revanche, tous les fichiers qu'il contient ne seront pas accessibles par leur nom tant que le système de fichiers sera monté.

Par exemple, pour monter une clé USB avec un système de fichiers exFAT située sur `/dev/sdb1` vers un répertoire appelé `flash` dans votre répertoire utilisateur, vous pouvez utiliser :

```
# mount -t exfat /dev/sdb1 ~/flash/
```

TIP

La plupart des systèmes Linux utilisent l'interpréteur de commandes Bash, et sur ces systèmes, le tilde `~` dans le chemin vers le point de montage est une abréviation pour le répertoire personnel de l'utilisateur actuel. Si l'utilisateur courant s'appelle `john`, par exemple, il sera remplacé par `/home/john`.

Après le montage, le contenu du système de fichiers sera accessible dans le répertoire `~/flash` :

```
$  ls -lh ~/flash/
total 469M
-rwxrwxrwx 1 root root 454M jul 19 09:49 lineage-16.0-20190711-MOD-quark.zip
-rwxrwxrwx 1 root root  16M jul 19 09:44 twrp-3.2.3-mod_4-quark.img
```

Afficher la liste des systèmes de fichiers montés

Si vous tapez simplement `mount`, vous obtenez une liste de tous les systèmes de fichiers en cours de montage sur votre système. Cette liste peut être assez conséquente, étant donné qu'en plus des disques attachés à votre système, ce dernier contient également un certain nombre de systèmes de fichiers exécutés en mémoire et destinés à des usages divers. Pour filtrer la sortie, vous pouvez utiliser le paramètre `-t` pour afficher uniquement les systèmes de fichiers du type correspondant,

comme ceci :

```
# mount -t ext4
/dev/sda1 on / type ext4 (rw,noatime,errors=remount-ro)
```

Vous pouvez spécifier plusieurs systèmes de fichiers à la fois en les séparant par une virgule :

```
# mount -t ext4,fuseblk
/dev/sda1 on / type ext4 (rw,noatime,errors=remount-ro)
/dev/sdb1 on /home/carol/flash type fuseblk
(rw,nosuid,nodev,relatime,user_id=0,group_id=0,default_permissions,allow_other,blksize=4096)
[DT_8GB]
```

Le résultat des exemples ci-dessus peut être décrit sous la forme :

```
SOURCE on TARGET type TYPE OPTIONS
```

Où `SOURCE` est la partition contenant le système de fichiers, `TARGET` est le répertoire dans lequel il est monté, `TYPE` est le type de système de fichiers et `OPTIONS` sont les options passées à la commande `mount` au moment du montage.

Les options de la commande

La commande `mount` peut être utilisée avec un certain nombre de paramètres. Voici les plus courants :

`-a`

Monter tous les systèmes de fichiers répertoriés dans le fichier `/etc/fstab` (voir un peu plus loin pour plus d'informations).

`-o` ou `--options`

Passer une liste *d'options de montage* séparées par des virgules à la commande `mount`, ce qui modifie la façon dont le système de fichiers sera monté. Ces options seront également discutées lorsque nous aborderons `/etc/fstab`.

`-r` ou `-ro`

Monter le système de fichiers en lecture seule.

`-w` ou `-rw`

Monter le système de fichiers en lecture/écriture.

Pour démonter un système de fichiers, utilisez la commande `umount` suivie du nom du périphérique ou du point de montage. Si l'on considère l'exemple précédent, les deux commandes ci-dessous sont interchangeables :

```
# umount /dev/sdb1
# umount ~/flash
```

Voici quelques options de la commande `umount` :

`-a`

Démonter tous les systèmes de fichiers répertoriés dans `/etc/fstab`.

`-f`

Force le démontage d'un système de fichiers. Cette option peut s'avérer utile si vous avez monté un système de fichiers distant et que celui-ci est devenu inaccessible.

`-r`

Si le système de fichiers ne peut pas être démonté, il sera mis en lecture seule.

Gérer les fichiers ouverts

Lors du démontage d'un système de fichiers, vous pouvez rencontrer un message d'erreur indiquant que la "cible est occupée" (`target is busy`). Cette erreur se produit lorsque des fichiers sont encore ouverts sur le système de fichiers. Cependant, il n'est pas toujours évident de savoir où se trouvent ces fichiers ouverts, ni ce qui est en train d'accéder au système de fichiers.

Dans ce cas, vous pouvez utiliser la commande `lsof` suivie du nom du périphérique contenant le système de fichiers pour obtenir une liste des processus qui y accèdent ainsi que des fichiers ouverts. Par exemple :

```
# umount /dev/sdb1
umount: /media/carol/External_Drive: target is busy.

# lsof /dev/sdb1
COMMAND  PID   USER   FD   TYPE DEVICE SIZE/OFF NODE NAME
evince  3135 carol   16r   REG   8,17 21881768 5195 /media/carol/External_Drive/Documents/E-
Books/MagPi40.pdf
```

`COMMAND` est le nom de l'exécutable qui a ouvert le fichier, et `PID` le numéro du processus. `NAME` est le nom du fichier ouvert. Dans l'exemple ci-dessus, le fichier `MagPi40.pdf` est ouvert par le programme `evince` (un visualiseur PDF). Si nous fermons le programme, nous pourrons alors démonter le système de fichiers.

Avant que le résultat de `lsof` n'apparaisse, les utilisateurs de `GNOME` risquent de voir un avertissement dans la fenêtre du terminal.

```
lsof: WARNING: can't stat() fuse.gvfsd-fuse file system /run/user/1000/gvfs
Output information may be incomplete.
```

NOTE

`lsof` essaie de traiter tous les systèmes de fichiers montés. Ce message d'avertissement est affiché parce que `lsof` a rencontré un système de fichiers virtuel `GNOME` (GVFS). Il s'agit là d'un cas particulier de système de fichiers dans l'espace utilisateur (`FUSE`). Il agit comme un pont entre `GNOME`, ses APIs et le noyau. Personne - pas même `root` - ne peut accéder à l'un de ces systèmes de fichiers, à part le propriétaire qui l'a monté (en l'occurrence, `GNOME`). Vous pouvez sereinement ignorer cet avertissement.

Où monter ?

Vous pouvez monter un système de fichiers à peu près n'importe où. En revanche, certaines bonnes pratiques doivent être respectées pour faciliter l'administration du système.

Traditionnellement, `/mnt` était le répertoire à l'intérieur duquel tous les périphériques externes étaient montés. Un certain nombre de "points d'ancrage" préconfigurés pour les périphériques courants comme les lecteurs de CD-ROM (`/mnt/cdrom`) et les disquettes (`/mnt/floppy`) existaient à l'intérieur de ce répertoire.

Il a été remplacé par `/media`, qui constitue désormais le point de montage par défaut pour tous les médias amovibles (disques externes, clés USB, lecteurs de cartes mémoire, etc.) connectés au système.

Sur la plupart des distributions Linux et des environnements de bureau modernes, les périphériques amovibles sont automatiquement montés sous `/media/USER/LABEL` lorsqu'ils sont connectés au système, où `USER` est le nom d'utilisateur et `LABEL` l'étiquette du périphérique. Par exemple, une clé USB étiquetée `FlashDrive` et insérée par l'utilisateur `john` sera montée sous `/media/john/FlashDrive/`. La manière dont cela est géré peut varier en fonction de l'environnement de bureau.

Ceci étant dit, lorsque vous avez besoin de monter *manuellement* un système de fichiers, c'est une

bonne pratique de le monter en dessous de `/mnt`.

Monter les systèmes de fichiers au démarrage

Le fichier `/etc/fstab` contient les spécifications des systèmes de fichiers qui peuvent être montés. Il s'agit là d'un fichier texte dans lequel chaque ligne décrit un système de fichiers destiné à être monté, avec six champs par ligne dans l'ordre suivant :

```
FILESYSTEM MOUNTPOINT TYPE OPTIONS DUMP PASS
```

Avec :

`FILESYSTEM`
: Le périphérique qui contient le système de fichiers à monter. Au lieu du périphérique, vous pouvez spécifier l'UUID ou l'étiquette de la partition, ce que nous allons voir un peu plus loin.

`MOUNTPOINT`
: L'endroit où le système de fichiers sera monté.

`TYPE`
: Le type de système de fichiers.

`OPTIONS`
: Les options de montage qui seront passées à la commande `mount`.

`DUMP`
: Indique si les systèmes de fichiers ext2, ext3 ou ext4 doivent être pris en compte pour la sauvegarde par la commande `dump`. En règle générale, ce paramètre est égal à zéro, ce qui signifie qu'ils doivent être ignorés.

`PASS`
: Lorsqu'il a une valeur non nulle, il définit l'ordre dans lequel les systèmes de fichiers seront vérifiés au démarrage. En général, il est égal à zéro.

Par exemple, la première partition du premier disque d'une machine peut être décrite comme ceci :

```
/dev/sda1  /  ext4  noatime,errors
```

Les options de montage spécifiées dans `OPTIONS` sont une liste de paramètres séparés par des virgules, qui peuvent être génériques ou spécifiques à un système de fichiers. Parmi les paramètres génériques, nous avons :

`atime` et `noatime`

Par défaut, à chaque fois qu'un fichier est lu, les informations de temps d'accès sont mises à jour. Désactiver ceci (avec `noatime`) peut accélérer les entrées-sorties sur le disque. Ne pas confondre avec le temps de modification, qui est mis à jour à chaque fois qu'un fichier est modifié.

`auto` et `noauto`

Indique si le système de fichiers peut (ou non) être monté automatiquement avec `mount -a`.

`defaults`

Cette option va passer les options `rw`, `suid`, `dev`, `exec`, `auto`, `nouser` et `async` à `mount`.

`dev` et `nodev`

Indique si les périphériques de type caractère ou de type bloc seront interprétés dans le système de fichiers monté.

`exec` et `noexec`

Permet ou empêche l'exécution de binaires sur le système de fichiers.

`user` et `nouser`

Permet (ou non) à un utilisateur normal de monter le système de fichiers.

`group`

Permet à un utilisateur de monter le système de fichiers s'il appartient au même groupe que celui qui est propriétaire du périphérique correspondant.

`owner`

Permet à un utilisateur de monter le système de fichiers s'il est propriétaire du périphérique concerné.

`suid` et `nosuid`

Autorise ou non les bits SETUID et SETGID à prendre effet.

`ro` et `rw`

Monte un système de fichiers en lecture seule ou en écriture.

`remount`

Cette option va essayer de remonter un système de fichiers déjà monté. Elle n'est pas utilisée dans `/etc/fstab`, mais comme paramètre de `mount -o`. Par exemple, pour remonter la partition montée `/dev/sdb1` en lecture seule, vous pouvez utiliser la commande `mount -o remount,ro /dev/sdb1`. Lors du remontage, vous n'avez pas besoin de spécifier le type de système de fichiers, simplement le nom du périphérique *ou* le point de montage.

`sync` et `async`

Indique si toutes les opérations d'entrées-sorties vers le système de fichiers doivent être effectuées de manière synchrone ou asynchrone. `async` est habituellement la valeur par défaut. La page de manuel de `mount` prévient que l'utilisation de `sync` sur un média avec un nombre limité de cycles d'écriture (comme les lecteurs flash ou les cartes mémoire) peut raccourcir la durée de vie du périphérique.

Utiliser les UUID et les étiquettes

Le fait de spécifier le nom du périphérique qui contient le système de fichiers à monter peut poser quelques problèmes. En effet, le même nom de périphérique peut parfois être attribué à un autre périphérique en fonction du moment ou de l'endroit où il a été connecté au système. Par exemple, une clé USB sur `/dev/sdb1` peut être assignée à `/dev/sdc1` si elle est branchée sur un autre port ou après une autre clé USB.

Une solution consiste à spécifier l'étiquette ou l'UUID (*Universally Unique Identifier*) du volume. Ces deux éléments sont définis lors de la création du système de fichiers et ne changeront pas, à moins que le système de fichiers ne soit détruit ou qu'une nouvelle étiquette ou un nouvel UUID ne lui soient attribués manuellement.

La commande `lsblk` peut être utilisée pour obtenir des informations sur un système de fichiers et trouver l'étiquette et l'UUID qui lui sont associés. Pour ce faire, utilisez l'option `-f` suivie du nom du périphérique :

```
$ lsblk -f /dev/sda1
NAME FSTYPE LABEL UUID                                 FSAVAIL FSUSE% MOUNTPOINT
sda1 ext4         6e2c12e3-472d-4bac-a257-c49ac07f3761   64,9G    33% /
```

Voici la signification de chacune des colonnes :

`NAME`

Nom du périphérique qui contient le système de fichiers.

`FSTYPE`
: Type du système de fichiers.

`LABEL`
: Étiquette du système de fichiers.

`UUID`
: Identifiant universel unique (UUID) attribué au système de fichiers.

`FSAVAIL`
: L'espace disponible dans le système de fichiers.

`FSUSE%`
: Pourcentage d'utilisation du système de fichiers.

`MOUNTPOINT`
: L'endroit où le système de fichiers est monté.

Dans `/etc/fstab`, un périphérique peut être spécifié par son UUID avec l'option `UUID=` suivie de l'UUID, ou avec `LABEL=` suivi du label. Ainsi, au lieu de :

```
/dev/sda1  /  ext4  noatime,errors
```

On pourra utiliser :

```
UUID=6e2c12e3-472d-4bac-a257-c49ac07f3761  /  ext4  noatime,errors
```

Ou, si vous avez un disque étiqueté `homedisk` :

```
LABEL=homedisk  /home ext4  defaults
```

La même syntaxe pourra être utilisée avec la commande `mount`. Au lieu du nom du périphérique, passez l'UUID ou l'étiquette. Par exemple, pour monter un disque NTFS externe avec l'UUID `56C11DCC5D2E1334` sur `/mnt/external`, la commande serait :

```
# mount -t ntfs UUID=56C11DCC5D2E1334 /mnt/external
```

Monter des disques avec Systemd

Systemd est le processus d'initialisation, le premier processus à s'exécuter sur la plupart des distributions Linux. Il se charge de lancer d'autres processus, de démarrer des services et d'amorcer le système. En dehors de ces tâches, systemd peut également être utilisé pour gérer le montage (et l'automontage) des systèmes de fichiers.

Pour utiliser cette fonctionnalité de systemd, vous devez créer un fichier de configuration appelé "unité de montage" (*mount unit*). Chaque volume à monter dispose de sa propre unité de montage qui devra être rangée dans `/etc/systemd/system/`.

Les unités de montage sont de simples fichiers texte avec l'extension `.mount`. Voici le format de base :

```
[Unit]
Description=

[Mount]
What=
Where=
Type=
Options=

[Install]
WantedBy=
```

`Description=`

Description sommaire de l'unité de montage, quelque chose comme `Monte le disque de sauvegarde`.

`What=`

Ce qui doit être monté. Le volume doit être spécifié sous la forme `/dev/disk/by-uuid/VOL_UUID` où `VOL_UUID` correspond à l'UUID du volume.

`Where=`

Indique le chemin complet vers l'endroit où le volume doit être monté.

`Type=`

Le type de système de fichiers.

`Options=`

Les options de montage que vous pouvez souhaiter passer, les mêmes que celles utilisées avec la commande `mount` ou dans `/etc/fstab`.

`WantedBy=`

Utilisé pour la gestion des dépendances. Dans ce cas, nous utiliserons `multi-user.target`, ce qui signifie qu'à chaque fois que le système démarre dans un environnement multi-utilisateur (un démarrage normal), l'unité sera montée.

L'exemple précédent du disque externe pourrait s'écrire comme ceci :

```
[Unit]
Description=External data disk

[Mount]
What=/dev/disk/by-uuid/56C11DCC5D2E1334
Where=/mnt/external
Type=ntfs
Options=defaults

[Install]
WantedBy=multi-user.target
```

Mais ce n'est pas tout. Pour fonctionner correctement, l'unité de montage doit *impérativement* avoir le même nom que le point de montage. Dans notre cas, le point de montage est `/mnt/external`. Le fichier devra donc être nommé `mnt-external.mount`.

Ensuite, vous devez recharger la configuration du démon systemd avec la commande `systemctl` et démarrer l'unité de montage :

```
# systemctl daemon-reload
# systemctl start mnt-external.mount
```

À présent, le contenu du disque externe devrait être disponible dans `/mnt/external`. Vous pouvez vérifier l'état du montage avec la commande `systemctl status mnt-external.mount`, comme ceci :

```
# systemctl status mnt-external.mount
● mnt-external.mount - External data disk
   Loaded: loaded (/etc/systemd/system/mnt-external.mount; disabled; vendor pres
```

```
   Active: active (mounted) since Mon 2019-08-19 22:27:02 -03; 14s ago
    Where: /mnt/external
     What: /dev/sdb1
    Tasks: 0 (limit: 4915)
   Memory: 128.0K
   CGroup: /system.slice/mnt-external.mount

ago 19 22:27:02 pop-os systemd[1]: Mounting External data disk...
ago 19 22:27:02 pop-os systemd[1]: Mounted External data disk.
```

La commande `systemctl start mnt-external.mount` n'activera l'unité que pour la session en cours. Si vous voulez l'activer à chaque démarrage, remplacez `start` par `enable` :

```
# systemctl enable mnt-external.mount
```

Montage automatique d'une unité de montage

Les unités de montage peuvent être montées automatiquement chaque fois que le point de montage est accédé. Pour ce faire, il vous faut un fichier `.automount` en plus du fichier `.mount` correspondant à l'unité. Voici le format de base :

```
[Unit]
Description=

[Automount]
Where=

[Install]
WantedBy=multi-user.target
```

Comme précédemment, `Description=` est un descriptif sommaire du fichier, et `Where=` correspond au point de montage. Par exemple, un fichier `.automount` pour l'exemple ci-dessus serait :

```
[Unit]
Description=Automount for the external data disk

[Automount]
Where=/mnt/external
```

```
[Install]
WantedBy=multi-user.target
```

Enregistrez le fichier avec le même nom que le point de montage (dans ce cas, `mnt-external.automount`), rechargez systemd et démarrez l'unité :

```
# systemctl daemon-reload
# systemctl start mnt-external.automount
```

Maintenant, chaque fois que le répertoire `/mnt/external` est accédé, le disque sera monté. Comme auparavant, pour activer le montage automatique à chaque démarrage, vous devez utiliser :

```
# systemctl enable mnt-external.automount
```

Exercices guidés

1. En utilisant `mount`, comment pouvez-vous monter un système de fichiers `ext4` sur `/dev/sdc1` vers `/mnt/external` en lecture seule et avec les options `noatime` et `async` ?

2. Lors du démontage d'un système de fichiers sur `/dev/sdd2`, vous obtenez le message d'erreur `target is busy`. Comment savoir quels fichiers du système de fichiers sont ouverts, et quels sont les processus qui les ont ouverts ?

3. Prenons l'entrée suivante dans `/etc/fstab` : `/dev/sdb1 /data ext4 noatime,noauto,async`. Est-ce que ce système de fichiers sera monté si la commande `mount -a` est invoquée ? Pourquoi ?

4. Comment trouver l'UUID d'un système de fichiers sous `/dev/sdb1` ?

5. Comment utiliser `mount` pour remonter en lecture seule un système de fichiers exFAT avec l'UUID `6e2c12e3-472d-4bac-a257-c49ac07f3761` monté sur `/mnt/data` ?

6. Comment peut-on afficher la liste de tous les systèmes de fichiers `ext3` et `ntfs` actuellement montés sur un système ?

Exercices d'approfondissement

1. Prenons l'entrée suivante dans `/etc/fstab` : `/dev/sdc1 /backup ext4 noatime,nouser,async`. Est-ce qu'un utilisateur peut monter ce système de fichiers avec la commande `mount /backup` ? Pourquoi ?

2. Imaginons un système de fichiers distant monté sur `/mnt/server`, et qui est devenu inaccessible à cause d'une perte de connectivité réseau. Comment pouvez-vous le forcer à être démonté ou remonté en lecture seule si cela n'est pas possible ?

3. Écrivez une entrée `/etc/fstab` pour monter un volume `btrfs` avec l'étiquette `Backup` sur `/mnt/backup`, en utilisant les options par défaut mais sans autoriser l'exécution de fichiers binaires depuis ce volume.

4. Prenons l'unité de montage systemd suivante :

```
[Unit]
Description=External data disk

[Mount]
What=/dev/disk/by-uuid/56C11DCC5D2E1334
Where=/mnt/external
Type=ntfs
Options=defaults

[Install]
WantedBy=multi-user.target
```

   - Quelle serait l'entrée `/etc/fstab` équivalente pour ce système de fichiers ?

5. Quel doit être le nom du fichier pour l'unité ci-dessus pour qu'il puisse être utilisé par systemd ? Où faut-il le placer ?

Résumé

Dans cette leçon, vous avez appris à monter et démonter des systèmes de fichiers, manuellement ou de manière automatique. Voici les commandes et les concepts abordés :

- `mount` (monte un périphérique vers un emplacement)
- `umount` (démonte un périphérique)
- `lsof` (liste des processus qui accèdent à un système de fichiers)
- les répertoires `/mnt` et `/media`
- `/etc/fstab`
- `lsblk` (affiche le type et l'UUID d'un système de fichiers)
- Monter un système de fichiers en utilisant son UUID ou son étiquette.
- Monter un système de fichiers en utilisant les unités de montage systemd.
- Monter automatiquement un système de fichiers avec les unités de montage systemd.

Réponses aux exercices guidés

1. En utilisant `mount`, comment pouvez-vous monter un système de fichiers `ext4` sur `/dev/sdc1` vers `/mnt/external` en lecture seule et avec les options `noatime` et `async` ?

   ```
   # mount -t ext4 -o noatime,async,ro /dev/sdc1 /mnt/external
   ```

2. Lors du démontage d'un système de fichiers sur `/dev/sdd2`, vous obtenez le message d'erreur `target is busy`. Comment savoir quels fichiers du système de fichiers sont ouverts, et quels sont les processus qui les ont ouverts ?

   Utilisez `lsof` suivi du nom du périphérique :

   ```
   $ lsof /dev/sdd2
   ```

3. Prenons l'entrée suivante dans `/etc/fstab` : `/dev/sdb1 /data ext4 noatime,noauto,async`. Est-ce que ce système de fichiers sera monté si la commande `mount -a` est invoquée ? Pourquoi ?

   Il ne sera pas monté. L'élément clé est le paramètre `noauto`, qui signifie que cette entrée sera ignorée par `mount -a`.

4. Comment trouver l'UUID d'un système de fichiers sous `/dev/sdb1` ?

   Utilisez `lsblk -f` suivi du nom du système de fichiers :

   ```
   $ lsblk -f /dev/sdb1
   ```

5. Comment utiliser `mount` pour remonter en lecture seule un système de fichiers exFAT avec l'UUID `6e2c12e3-472d-4bac-a257-c49ac07f3761` monté sur `/mnt/data` ?

   Puisque le système de fichiers est monté, ce n'est pas la peine de vous inquiéter du type de système de fichiers ou de l'ID, il suffit d'utiliser l'option `remount` avec le paramètre `ro` (lecture seule) et le point de montage :

   ```
   # mount -o remount,ro /mnt/data
   ```

6. Comment peut-on afficher la liste de tous les systèmes de fichiers `ext3` et `ntfs` actuellement montés sur un système ?

Utilisez `mount -t` suivi de la liste des systèmes de fichiers séparés par des virgules :

```
# mount -t ext3,ntfs
```

Réponses aux exercices d'approfondissement

1. Prenons l'entrée suivante dans `/etc/fstab` : `/dev/sdc1 /backup ext4 noatime,nouser,async`. Est-ce qu'un utilisateur peut monter ce système de fichiers avec la commande `mount /backup` ? Pourquoi ?

   Non, le paramètre `nouser` ne permettra pas aux utilisateurs normaux de monter ce système de fichiers.

2. Imaginons un système de fichiers distant monté sur `/mnt/server`, et qui est devenu inaccessible à cause d'une perte de connectivité réseau. Comment pouvez-vous le forcer à être démonté ou remonté en lecture seule si cela n'est pas possible ?

   Passez les options `-f` et `-r` au démontage. La commande serait `umount -f -r /mnt/serveur`. Rappelez-vous que vous pouvez combiner les options, donc `umount -fr /mnt/serveur` fonctionnerait également.

3. Écrivez une entrée `/etc/fstab` pour monter un volume `btrfs` avec l'étiquette `Backup` sur `/mnt/backup`, en utilisant les options par défaut mais sans autoriser l'exécution de fichiers binaires depuis ce volume.

   La ligne doit s'écrire `LABEL=Backup /mnt/backup btrfs defaults,noexec`

4. Prenons l'unité de montage systemd suivante :

   ```
   [Unit]
   Description=External data disk

   [Mount]
   What=/dev/disk/by-uuid/56C11DCC5D2E1334
   Where=/mnt/external
   Type=ntfs
   Options=defaults

   [Install]
   WantedBy=multi-user.target
   ```

   - Quelle serait l'entrée `/etc/fstab` équivalente pour ce système de fichiers ?

     L'entrée serait : `UUID=56C11DCC5D2E1334 /mnt/external ntfs defaults`

5. Quel doit être le nom du fichier pour l'unité ci-dessus pour qu'il puisse être utilisé par

systemd ? Où faut-il le placer ?

Le nom du fichier doit correspondre au point de montage, donc `mnt-external.mount`, placé dans `/etc/systemd/system`.

104.5 Gestion des permissions et de la propriété sur les fichiers

Référence aux objectifs de LPI

LPIC-1 version 5.0, Exam 101, Objective 104.5

Valeur

3

Liste partielle de termes, fichiers et utilitaires utilisés pour cet objectif

- Gestion des permissions d'accès sur les fichiers standards et les fichiers spéciaux, ainsi que sur les répertoires.
- Utilisation des modes d'accès comme suid, sgid et sticky bit pour maintenir la sécurité.
- Savoir changer le masque de création des fichiers par défaut.
- Utilisation du champ groupe pour attribuer les permissions aux membres d'un groupe.

Partial list of the used files, terms and utilities

- `chmod`
- `umask`
- `chown`
- `chgrp`

104.5 Leçon 1

Certification :	LPIC-1
Version :	5.0
Thème :	104 Disques, systèmes de fichiers Linux, arborescence de fichiers standard (FHS)
Objectif :	104.5 Gérer les droits d'accès et les propriétaires des fichiers
Leçon :	1 sur 1

Introduction

En tant que système multi-utilisateurs, Linux doit disposer d'un mécanisme permettant de savoir à qui appartient chaque fichier et si tel ou tel utilisateur est autorisé à effectuer des actions sur un fichier. Cela permet de garantir la confidentialité des utilisateurs qui voudraient protéger le contenu de leurs fichiers, et aussi d'assurer la collaboration en rendant certains fichiers accessibles à plusieurs utilisateurs.

Pour ce faire, un système de permissions à trois niveaux est mis en place. Chaque fichier sur le disque appartient à un utilisateur et à un groupe d'utilisateurs et possède trois jeux de permissions : un pour son propriétaire, un pour le groupe qui détient le fichier et un pour tous les autres. Dans cette leçon, vous apprendrez à consulter les permissions d'un fichier, à en connaître la signification et à les manipuler.

Consulter les informations sur les fichiers et les répertoires

La commande `ls` est utilisée pour afficher une liste du contenu de n'importe quel répertoire. Dans

sa forme basique, tout ce que vous obtenez, ce sont les noms des fichiers :

```
$ ls
Another_Directory  picture.jpg  text.txt
```

Mais il y a beaucoup plus d'informations disponibles pour chaque fichier, y compris son type, sa taille, son propriétaire et plus encore. Pour voir ces informations, vous devez demander à `ls` un affichage détaillé en utilisant l'option `-l` :

```
$ ls -l
total 536
drwxrwxr-x 2 carol carol   4096 Dec 10 15:57 Another_Directory
-rw------- 1 carol carol 539663 Dec 10 10:43 picture.jpg
-rw-rw-r-- 1 carol carol   1881 Dec 10 15:57 text.txt
```

Chaque colonne du résultat ci-dessus a une signification. Examinons les colonnes pertinentes pour cette leçon.

- La première colonne de la liste indique le type de fichier et les permissions. Par exemple, pour `drwxrwxr-x` :
  - Le premier caractère, `d`, indique le type de fichier.
  - Les trois caractères suivants, `rwx`, indiquent les permissions du *propriétaire* du fichier, également appelé *user* ou `u`.
  - Les trois caractères suivants, `rwx`, indiquent les permissions du *groupe* propriétaire du fichier, également appelé `g`.
  - Les trois derniers caractères, `r-x`, indiquent les permissions pour toute autre personne, autrement dit *others* ou `o`.

TIP Dans la langue anglaise, le jeu de permissions *others* est également appelé *world* dans le sens de "Everyone else in the world has these permissions" (Tout le monde a ces droits).

- La *troisième* et la *quatrième* colonne indiquent respectivement l'utilisateur et le groupe qui détiennent le fichier.
- La *septième* et dernière colonne indique le nom du fichier.

La *deuxième* colonne indique le nombre de liens physiques qui pointent vers le fichier. La *cinquième* colonne indique la taille du fichier. La *sixième* colonne indique la date et l'heure de la

dernière modification du fichier. Ces colonnes ne sont pas pertinentes pour le sujet qui nous intéresse.

Et les répertoires ?

Si vous essayez d'obtenir des informations détaillées sur un répertoire en utilisant `ls -l`, vous obtiendrez une liste du contenu du répertoire, ce qui n'est pas ce que nous recherchons :

```
$ ls -l Another_Directory/
total 0
-rw-r--r-- 1 carol carol 0 Dec 10 17:59 another_file.txt
```

Pour éviter cela et obtenir des informations sur le répertoire lui-même, ajoutez l'option `-d` à `ls` :

```
$ ls -l -d Another_Directory/
drwxrwxr-x 2 carol carol 4096 Dec 10 17:59 Another_Directory/
```

Afficher les fichiers cachés

Le listing du répertoire que nous avons effectué auparavant en utilisant `ls -l` est incomplet :

```
$ ls -l
total 544
drwxrwxr-x 2 carol carol   4096 Dec 10 17:59 Another_Directory
-rw------- 1 carol carol 539663 Dec 10 10:43 picture.jpg
-rw-rw-r-- 1 carol carol   1881 Dec 10 15:57 text.txt
```

Il y a trois autres fichiers dans ce répertoire, mais ils sont cachés. Sous Linux, les fichiers dont le nom commence par un point (`.`) sont automatiquement masqués. Pour les afficher, il faut ajouter l'option `-a` à `ls` :

```
$ ls -l -a
total 544
drwxrwxr-x 3 carol carol   4096 Dec 10 16:01 .
drwxrwxr-x 4 carol carol   4096 Dec 10 15:56 ..
drwxrwxr-x 2 carol carol   4096 Dec 10 17:59 Another_Directory
-rw------- 1 carol carol 539663 Dec 10 10:43 picture.jpg
-rw-rw-r-- 1 carol carol   1881 Dec 10 15:57 text.txt
-rw-r--r-- 1 carol carol      0 Dec 10 16:01 .thisIsHidden
```

Le fichier `.thisIsHidden` est caché tout simplement parce que son nom commence par `.`.

En revanche, les répertoires `.` et `..` sont particuliers. `.` est un pointeur vers le répertoire courant. Quant à `..`, c'est un pointeur vers le répertoire parent, celui qui contient le répertoire courant. Sous Linux, chaque répertoire contient au moins ces deux répertoires.

TIP Vous pouvez combiner plusieurs options pour `ls` (et beaucoup d'autres commandes Linux). Par exemple, `ls -l -a` pourra s'écrire `ls -la`.

Comprendre les types de fichiers

Nous avons dit que la première lettre de chaque ligne de résultat de `ls -l` décrivait le type de fichier. Les trois types de fichiers les plus courants sont :

`-` (fichier normal)

Un fichier peut contenir toutes sortes de données et permet de les gérer. Les fichiers peuvent être modifiés, déplacés, copiés et supprimés.

`d` (répertoire)

Un répertoire (*directory*) contient d'autres fichiers ou répertoires et permet d'organiser le système de fichiers. Techniquement, les répertoires sont un type de fichier particulier.

`l` (lien symbolique)

Ce "fichier" est un pointeur (ou raccourci) vers un autre fichier ou répertoire situé ailleurs dans le système de fichiers.

En plus de ces types de fichiers, il en existe trois autres dont vous devriez au moins connaître l'existence, mais qui n'entrent pas dans le cadre de cette leçon :

`b` (périphérique de type bloc)

Ce fichier représente un périphérique virtuel ou physique, généralement des disques ou d'autres types de périphériques de stockage, comme le premier disque dur qui pourrait être représenté par `/dev/sda`.

`c` (périphérique de type caractère)

Ce fichier représente un périphérique virtuel ou physique. Les terminaux (comme le terminal principal sur `/dev/ttyS0`) et les ports série sont des exemples courants de périphériques de type caractère.

`s` (socket)

Les sockets servent de "vases communicants" pour transmettre des informations entre deux

programmes.

WARNING Ne modifiez en aucun cas les permissions sur les périphériques de type bloc, caractère ou socket, à moins que vous ne sachiez ce que vous faites. Vous risqueriez d'empêcher votre système de fonctionner !

Comprendre les droits d'accès

Dans l'affichage de `ls -l`, les droits d'accès aux fichiers sont indiqués juste après le type de fichier, sous la forme de trois groupes de trois caractères chacun, dans l'ordre `r`, `w` et `x`. Voici ce qu'ils signifient. Gardez à l'esprit qu'un tiret `-` représente l'absence d'une permission.

Droits d'accès aux fichiers

`r`

Signifie *read* (lecture) avec une valeur octale de `4` (ne vous inquiétez pas, nous en parlerons prochainement). Cela correspond à la permission d'ouvrir un fichier et d'en lire le contenu.

`w`

Signifie *write* (écriture) avec une valeur octale de `2`. Cela équivaut au droit de modifier le contenu d'un fichier.

`x`

Signifie *execute* (exécution) avec une valeur octale de `1`. Cela signifie que le fichier peut être lancé comme un exécutable ou un script.

Par exemple, un fichier avec les permissions `rw-` peut être lu et écrit, mais il ne pourra pas être exécuté.

Droits d'accès aux répertoires

`r`

Signifie *read* (lecture) avec une valeur octale de `4`. Cela représente le droit de lire le contenu du répertoire, comme les noms des fichiers. En revanche, cela n'implique pas forcément la permission de lire le contenu des fichiers eux-mêmes.

`w`

Signifie *write* (écriture) avec une valeur octale de `2`. Cela signifie que l'on peut créer ou supprimer des fichiers dans le répertoire en question.

Notez bien que vous ne pouvez pas effectuer ces changements avec les seules permissions

write, mais qu'il vous faut également la permission *execute* (`x`) pour modifier le répertoire.

`x`

Signifie *execute* (exécution) avec une valeur octale de `1`. Cela signifie que l'on a la permission d'entrer dans un répertoire, mais pas d'en afficher les fichiers (pour cela, il faut `r`).

Cette dernière partie sur les répertoires peut prêter à confusion. Imaginons, par exemple, que vous ayez un répertoire nommé `Another_Directory` avec les permissions suivantes :

```
$ ls -ld Another_Directory/
d--x--x--x 2 carol carol 4,0K Dec 20 18:46 Another_Directory
```

Supposons que dans ce répertoire vous ayez un script shell nommé `hello.sh` :

```
-rwxr-xr-x 1 carol carol 33 Dec 20 18:46 hello.sh
```

Si vous êtes l'utilisatrice `carol` et que vous essayez d'afficher le contenu de `Another_Directory`, vous obtiendrez un message d'erreur, étant donné que votre utilisatrice n'a pas les droits de lecture pour ce répertoire :

```
$ ls -l Another_Directory/
ls: cannot open directory 'Another_Directory/': Permission denied
```

En revanche, l'utilisatrice `carol` dispose bien des permissions d'exécution, ce qui signifie qu'elle peut entrer dans le répertoire. Par conséquent, l'utilisatrice `carol` peut accéder aux fichiers à l'intérieur du répertoire, tant qu'elle a les permissions correctes *pour le fichier en question*. Supposons que l'utilisatrice ait les permissions complètes (`rwx`) pour le script `hello.sh`. Alors elle *peut* exécuter le script, même si elle *ne peut pas* lire le contenu du répertoire qui le contient si elle connaît le nom complet du fichier :

```
$ sh Another_Directory/hello.sh
Hello LPI World!
```

Comme nous l'avons dit plus haut, les autorisations sont spécifiées dans l'ordre : d'abord pour le propriétaire du fichier, puis pour le groupe propriétaire, et enfin pour les autres utilisateurs. Lorsque quelqu'un tente d'effectuer une action sur le fichier, les autorisations sont vérifiées de la même manière.

Le système vérifie d'abord si l'utilisateur en cours est propriétaire du fichier et, si c'est le cas, il

n'applique que la première série d'autorisations. Dans le cas contraire, il vérifie si l'utilisateur appartient au groupe propriétaire du fichier. Dans ce cas, il n'applique que la deuxième série d'autorisations. Autrement, le système appliquera le troisième jeu d'autorisations.

Cela signifie que si l'utilisateur en cours est le propriétaire du fichier, seules les autorisations du propriétaire sont effectives, même si les autorisations du groupe ou d'autres autorisations sont plus permissives que celles du propriétaire.

Modifier les droits d'accès des fichiers

La commande `chmod` est utilisée pour modifier les droits d'accès d'un fichier. Elle prend au moins deux paramètres : le premier décrit les droits à modifier, le second désigne le fichier ou le répertoire dans lequel la modification sera effectuée. Gardez à l'esprit que seul le propriétaire du fichier ou l'administrateur du système (`root`) peut modifier les droits d'un fichier.

Les droits à modifier peuvent être décrits de deux manières ou "modes" différents.

Le premier, appelé *mode symbolique*, offre un contrôle très précis et vous permet d'ajouter ou de révoquer une seule autorisation sans pour autant modifier les autres autorisations de la série. L'autre mode, appelé *mode octal*, est plus facile à mémoriser et plus rapide à utiliser si vous souhaitez définir d'un coup l'ensemble des droits d'accès.

Les deux modes aboutissent au même résultat. Par exemple, les commandes :

```
$ chmod ug+rw-x,o-rwx text.txt
```

et

```
$ chmod 660 text.txt
```

vont produire exactement le même résultat, un fichier avec les droits :

```
-rw-rw---- 1 carol carol  765 Dec 20 21:25 text.txt
```

Voyons maintenant le fonctionnement de chaque mode.

Le mode symbolique

Lorsque vous décrivez les droits à modifier en *mode symbolique*, le(s) premier(s) caractère(s)

indique(nt) les droits que vous allez modifier : ceux de l'utilisateur (`u`), du groupe (`g`), des autres (`o`) et/ou de tout le monde (`a`).

Ensuite, il vous faut indiquer à la commande ce qu'elle doit faire : vous pouvez accorder un droit (`+`), révoquer un droit (`-`) ou lui donner une valeur spécifique (`=`).

Enfin, vous indiquez le droit sur lequel vous souhaitez agir : lecture (`r`), écriture (`w`) ou exécution (`x`).

Par exemple, prenons un fichier nommé `text.txt` avec les droits d'accès suivants :

```
$ ls -l text.txt
-rw-r--r-- 1 carol carol 765 Dec 20 21:25 text.txt
```

Si vous souhaitez accorder les droits en écriture aux membres du groupe propriétaire du fichier, vous devez utiliser le paramètre `g+w`. C'est plus facile si vous y pensez de cette façon : "Pour le groupe (`g`), accorder (`+`) les droits d'écriture (`w`)". Ainsi, la commande serait :

```
$ chmod g+w text.txt
```

Vérifions le résultat avec `ls` :

```
$ ls -l text.txt
-rw-rw-r-- 1 carol carol 765 Dec 20 21:25 text.txt
```

Vous souhaitez supprimer les droits en lecture du propriétaire de ce même fichier ? Pensez-y de cette manière : "Pour l'utilisateur (`u`), révoquer (`-`) les droits en lecture (`r`)". Le paramètre est donc `u-r`, comme ceci :

```
$ chmod u-r text.txt
$ ls -l text.txt
--w-rw-r-- 1 carol carol 765 Dec 20 21:25 text.txt
```

Et si nous voulions définir les droits exactement comme `rw-` pour tout le monde ? Dans ce cas, il faut voir les choses comme suit : "Pour tous (`a`), définir exactement (`=`) lecture (`r`), écriture (`w`), et pas d'exécution (`-`)". Donc :

```
$ chmod a=rw- text.txt
$ ls -l text.txt
```

```
-rw-rw-rw- 1 carol carol 765 Dec 20 21:25 text.txt
```

Bien entendu, il est possible de modifier plusieurs droits en même temps. Dans ce cas, il faut les séparer par une virgule (`,`) :

```
$ chmod u+rwx,g-x text.txt
$ ls -lh text.txt
-rwxrw-rw- 1 carol carol 765 Dec 20 21:25 text.txt
```

L'exemple ci-dessus peut être lu comme suit : "Pour l'utilisateur (`u`), accorder (`+`) les droits en lecture, écriture et exécution (`rwx`), et pour le groupe (`g`), révoquer (`-`) les droits d'exécution (`x`)".

Lorsqu'on l'applique à un répertoire, `chmod` ne modifie que les droits de ce dernier. La commande `chmod` a également un mode récursif, qui est utile lorsque vous voulez changer les droits pour tous les fichiers d'un répertoire et de ses sous-répertoires. Pour l'utiliser, ajoutez l'option `-R` juste après la commande et avant les droits à modifier :

```
$ chmod -R u+rwx Another_Directory/
```

Cette commande peut être lue comme suit : "Récursivement (`-R`), pour l'utilisateur (`u`), accorder (`+`) les droits en lecture, écriture et exécution (`rwx`)".

WARNING

Soyez prudent et réfléchissez à deux fois avant d'utiliser l'option `-R`, étant donné qu'il est facile de modifier les droits sur des fichiers et des répertoires que vous ne souhaitez pas forcément modifier, en particulier avec des répertoires contenant un grand nombre de fichiers et de sous-répertoires.

Le mode octal

En mode *octal*, les permissions sont spécifiées différemment : sous forme d'une valeur à trois chiffres en notation octale, un système numérique en base 8.

Chaque permission a une valeur correspondante, et elles sont spécifiées dans l'ordre suivant : d'abord la lecture (`r`) qui vaut `4`, puis l'écriture (`w`) qui vaut `2`, et enfin l'exécution (`x`) représentée par `1`. En l'absence de permission, on utilise la valeur zéro (`0`). Ainsi, une permission de `rwx` correspondrait à `7` (`4+2+1`) et `r-x` à `5` (`4+0+1`).

Le premier des trois chiffres du jeu de permissions représente les droits de l'utilisateur (`u`), le deuxième ceux du groupe (`g`) et le troisième ceux de tous les autres (`o`). Si nous voulons fixer les permissions d'un fichier à `rw-rw----`, la valeur octale correspondante sera `660` :

```
$ chmod 660 text.txt
$ ls -l text.txt
-rw-rw---- 1 carol carol 765 Dec 20 21:25 text.txt
```

En dehors de cela, la syntaxe en *mode octal* est la même qu'en *mode symbolique*. Le premier paramètre représente les permissions que vous souhaitez modifier, et le second paramètre pointe vers le fichier ou le répertoire pour lequel la modification sera effectuée.

TIP Lorsqu'une valeur de permission est *impaire*, le fichier est forcément exécutable !

Quelle syntaxe utiliser ? Le *mode octal* est recommandé si vous voulez fixer les permissions à une valeur spécifique, par exemple `640` (`rw- r-- ---`).

Le *mode symbolique* est plus adapté si vous voulez basculer une valeur spécifique sans tenir compte des permissions en vigueur pour le fichier. Par exemple, vous pouvez ajouter des permissions d'exécution pour l'utilisateur en utilisant simplement `chmod u+x script.sh` sans tenir compte, ou même toucher, aux permissions actuelles pour le groupe et les autres.

Modifier le propriétaire et le groupe d'un fichier

La commande `chown` est utilisée pour modifier le propriétaire et le groupe d'un fichier ou d'un répertoire. La syntaxe est très simple :

```
chown USERNAME:GROUPNAME FILENAME
```

Par exemple, prenons un fichier `text.txt` :

```
$ ls -l text.txt
-rw-rw---- 1 carol carol 1881 Dec 10 15:57 text.txt
```

L'utilisatrice qui possède le fichier est `carol`, et le groupe est également `carol`. Maintenant, nous allons changer le groupe propriétaire du fichier en un autre groupe, comme `students` :

```
$ chown carol:students text.txt
$ ls -l text.txt
-rw-rw---- 1 carol students 1881 Dec 10 15:57 text.txt
```

Gardez à l'esprit que l'utilisateur qui possède un fichier n'a pas forcément besoin d'appartenir au groupe qui possède un fichier. Dans l'exemple ci-dessus, l'utilisatrice `carol` n'a pas besoin d'être

membre du groupe `students`.

Les autorisations de l'utilisateur ou du groupe peuvent être omises si vous ne souhaitez pas les modifier. Ainsi, pour changer uniquement le groupe propriétaire d'un fichier, vous utiliserez `chown :students text.txt`. Pour changer uniquement l'utilisateur, la commande serait `chown carol: text.txt` ou simplement `chown carol text.txt`. Alternativement, vous pouvez utiliser la commande `chgrp students text.txt`.

À moins d'être l'administrateur du système (root), vous ne pouvez pas changer le propriétaire ou le groupe d'un fichier au profit d'un autre utilisateur ou d'un groupe auquel vous n'appartenez pas. Si vous essayez de le faire, vous obtiendrez le message d'erreur `Operation not permitted`.

Effectuer des requêtes sur les groupes

Avant de modifier le propriétaire et le groupe d'un fichier, il peut être utile de savoir quels groupes existent sur le système, quels utilisateurs sont membres d'un groupe et à quels groupes un utilisateur appartient.

Pour voir quels groupes existent sur votre système, tapez `getent group`. Le résultat ressemblera à ceci (l'affichage a été abrégé) :

```
$ getent group
root:x:0:
daemon:x:1:
bin:x:2:
sys:x:3:
adm:x:4:syslog,rigues
tty:x:5:rigues
disk:x:6:
lp:x:7:
mail:x:8:
news:x:9:
uucp:x:10:rigues
```

Si vous voulez savoir à quels groupes un utilisateur appartient, ajoutez le nom d'utilisateur comme paramètre à `groups` :

```
$ groups carol
carol : carol students cdrom sudo dip plugdev lpadmin sambashare
```

Pour faire l'inverse (voir quels utilisateurs appartiennent à un groupe), utilisez `groupmems`.

L'option `-g` spécifie le groupe, et `-l` affichera la liste de tous ses membres :

```
# groupmems -g cdrom -l
carol
```

TIP `groupmems` doit être exécuté en tant que root, l'administrateur du système. Si vous n'êtes pas connecté en tant que root, ajoutez `sudo` avant la commande.

Les droits par défaut

Faisons une petite expérience. Ouvrez un terminal et créez un fichier vide à l'aide de la commande suivante :

```
$ touch testfile
```

Jetons un coup d'œil sur les permissions de ce fichier. Elles *peuvent* être différentes sur votre système, mais admettons qu'elles ressemblent à ceci :

```
$ ls -lh testfile
-rw-r--r-- 1 carol carol 0 jul 13 21:55 testfile
```

Les permissions sont `rw-r—r--` : *lecture* et *écriture* pour l'utilisateur, et *lecture* pour le groupe et les autres, ou `644` en mode octal. Maintenant, essayez de créer un répertoire :

```
$ mkdir testdir
$ ls -lhd testdir
drwxr-xr-x 2 carol carol 4,0K jul 13 22:01 testdir
```

Les permissions sont maintenant `rwxr-xr-x` : *lecture*, *écriture* et *exécution* pour l'utilisateur, *lecture* et *exécution* pour le groupe et les autres, ou `755` en mode octal.

Quel que soit l'endroit où vous vous trouvez dans le système de fichiers, chaque fichier ou répertoire que vous créez recevra les mêmes permissions. Est-ce que vous vous êtes déjà demandé d'où viennent ces droits ?

Ils proviennent du *user mask* ou `umask`, qui définit les permissions par défaut pour chaque fichier créé. Vous pouvez vérifier les valeurs en vigueur avec la commande `umask` :

```
$ umask
0022
```

Cela ne ressemble pas vraiment à `rw-r—r--`, ni même à `644`. Peut-être devrions-nous essayer avec le paramètre `-S`, pour obtenir un affichage en mode symbolique :

```
$ umask -S
u=rwx,g=rx,o=rx
```

Ce sont là les mêmes droits que ceux accordés à notre répertoire de test dans l'un des exemples ci-dessus. Or, comment se fait-il que lorsque nous créons un fichier, les droits d'accès ne sont pas les mêmes ?

Eh bien, cela n'a pas de sens de définir par défaut des droits d'exécution globaux pour tout le monde sur n'importe quel fichier, n'est-ce pas ? Les répertoires en revanche ont besoin de droits d'exécution (sinon vous ne pouvez pas y entrer), mais les fichiers n'en ont pas, donc ils ne les obtiennent pas. D'où le `rw-r—r--`.

En plus d'afficher les droits par défaut, `umask` peut également être utilisé pour les modifier dans la session en cours de l'interpréteur de commandes. Par exemple, si nous utilisons la commande :

```
$ umask u=rwx,g=rwx,o=
```

Chaque nouveau répertoire va hériter des droits `rwxrwx---`, et chaque fichier `rw-rw----` (car ils n'ont pas les droits d'exécution). Si vous reprenez les exemples ci-dessus pour créer un fichier `testfile` et un répertoire `testdir` et que vous vérifiez les permissions, vous devriez obtenir :

```
$ ls -lhd test*
drwxrwx--- 2 carol carol 4,0K jul 13 22:25 testdir
-rw-rw---- 1 carol carol    0 jul 13 22:25 testfile
```

Et si vous vérifiez le `umask` sans l'option `-S` (mode symbolique), vous obtenez :

```
$ umask
0007
```

Le résultat n'est pas évident étant donné que les valeurs utilisées sont différentes. Voici un tableau avec chaque valeur et sa signification respective :

Valeurs	Droits pour les fichiers	Droits pour les répertoires
`0`	`rw-`	`rwx`
`1`	`rw-`	`rw-`
`2`	`r--`	`r-x`
`3`	`r--`	`r--`
`4`	`-w-`	`-wx`
`5`	`-w-`	`-w-`
`6`	`---`	`--x`
`7`	`---`	`---`

Comme vous pouvez le voir, `007` correspond à `rwxrwx---`, exactement comme nous l'avons souhaité. Vous pouvez ignorer le zéro initial.

Droits d'accès étendus

En dehors des droits de lecture, d'écriture et d'exécution pour l'utilisateur, le groupe et les autres, chaque fichier peut comporter trois autres *droits d'accès étendus* qui peuvent modifier le fonctionnement d'un répertoire ou l'exécution d'un programme. Ces droits peuvent être définis en mode symbolique ou octal, à savoir :

Sticky Bit

Le *sticky bit*, également appelé *fanion de suppression restreinte*, a la valeur octale `1`. En mode symbolique, il est représenté par un `t` dans les permissions des autres. Il s'applique aux seuls répertoires et n'a aucun effet sur les fichiers normaux. Sous Linux, il empêche les utilisateurs de supprimer ou de renommer un fichier dans un répertoire à moins qu'ils ne soient propriétaires de ce fichier ou de ce répertoire.

Les répertoires avec le *sticky bit* activé présentent un `t` à la place du `x` au niveau des droits pour les autres (*others*) dans l'affichage de la commande `ls -l` :

```
$ ls -ld Sample_Directory/
drwxr-xr-t 2 carol carol 4096 Dec 20 18:46 Sample_Directory/
```

En mode octal, les droits d'accès étendus sont spécifiés à l'aide d'une notation à quatre chiffres, le premier chiffre représentant le droit spécial sur lequel on souhaite agir. Par exemple, pour activer le *sticky bit* (valeur `1`) pour le répertoire `Another_Directory` en mode octal avec les permissions

`755`, la commande serait la suivante :

```
$ chmod 1755 Another_Directory
$ ls -ld Another_Directory
drwxr-xr-t 2 carol carol 4,0K Dec 20 18:46 Another_Directory
```

SGID

SGID, également connu sous le nom de Set GID ou Set Group ID bit, a la valeur octale `2`. En mode symbolique, il est représenté par un `s` sur les permissions du groupe (*group*). Ce bit s'applique aux fichiers exécutables ou aux répertoires. Dans le cas des fichiers, le processus s'exécutera avec les privilèges du groupe propriétaire du fichier. Lorsqu'il est appliqué à un répertoire, il fait en sorte que chaque fichier ou répertoire créé à l'intérieur hérite du groupe du répertoire parent.

Les fichiers et répertoires avec un bit SGID présentent un `s` à la place du `x` au niveau des droits pour le groupe (*group*) dans l'affichage de `ls -l` :

```
$ ls -l test.sh
-rwxr-sr-x 1 carol root 33 Dec 11 10:36 test.sh
```

Pour ajouter des droits SGID à un fichier en mode symbolique, la commande est la suivante :

```
$ chmod g+s test.sh
$ ls -l test.sh
-rwxr-sr-x 1 carol root     33 Dec 11 10:36 test.sh
```

L'exemple suivant vous aidera à mieux comprendre les effets du SGID sur un répertoire. Admettons que nous ayons un répertoire appelé `Sample_Directory` appartenant à l'utilisatrice `carol` et au groupe `users`, avec la structure de droits suivante :

```
$ ls -ldh Sample_Directory/
drwxr-xr-x 2 carol users 4,0K Jan 18 17:06 Sample_Directory/
```

Maintenant, changeons de répertoire et créons un fichier vide à l'intérieur de celui-ci en utilisant la commande `touch`. Le résultat de cette opération serait le suivant :

```
$ cd Sample_Directory/
$ touch newfile
$ ls -lh newfile
```

```
-rw-r--r-- 1 carol carol 0 Jan 18 17:11 newfile
```

Comme nous pouvons le voir, le fichier appartient à l'utilisatrice `carol` et au groupe `carol`. Cependant, si le répertoire avait le droit SGID, le résultat serait différent. Pour commencer, ajoutons le bit SGID au répertoire `Sample_Directory` et vérifions le résultat :

```
$ sudo chmod g+s Sample_Directory/
$ ls -ldh Sample_Directory/
drwxr-sr-x 2 carol users 4,0K Jan 18 17:17 Sample_Directory/
```

Le `s` sur les droits du groupe indique que le bit SGID est activé. Maintenant, changeons de répertoire et créons un autre fichier vide avec la commande `touch` :

```
$ cd Sample_Directory/
$ touch emptyfile
$ ls -lh emptyfile
 -rw-r--r-- 1 carol users 0 Jan 18 17:20 emptyfile
```

Le groupe propriétaire du fichier est `users`. C'est parce que le bit SGID a fait en sorte que le fichier hérite du groupe propriétaire du répertoire parent, qui est `users`.

SUID

SUID, également connu sous le nom de Set User ID, a la valeur octale `4`. Il est représenté par un `s` sur les droits de l'utilisateur (*user*) en mode symbolique. Il s'applique aux seuls fichiers et n'a aucun effet sur les répertoires. Son comportement ressemble à celui du bit SGID, mais le processus s'exécutera avec les privilèges de l'*utilisateur* propriétaire du fichier. Les fichiers avec le bit SUID présentent un `s` à la place du `x` au niveau des droits de l'utilisateur dans l'affichage de `ls -l` :

```
$ ls -ld test.sh
-rwsr-xr-x 1 carol carol 33 Dec 11 10:36 test.sh
```

Vous pouvez combiner plusieurs droits d'accès étendus en un seul paramètre. Ainsi, pour fixer le SGID (valeur `2`) et le SUID (valeur `4`) en mode octal pour le script `test.sh` avec les permissions `755`, vous devez taper :

```
$ chmod 6755 test.sh
```

Et le résultat serait :

```
$ ls -lh test.sh
-rwsr-sr-x 1 carol carol 66 Jan 18 17:29 test.sh
```

TIP

Si votre terminal supporte les couleurs, ce qui est le cas de la plupart des terminaux de nos jours, vous pouvez voir rapidement si les droits d'accès étendus ont été définis en regardant l'affichage de `ls -l`. Pour le *sticky bit*, le nom du répertoire s'affiche en noir sur fond bleu. Le même principe s'applique aux fichiers avec les bits SGID (fond jaune) et SUID (fond rouge). Les couleurs peuvent varier en fonction de la distribution Linux et des paramètres du terminal que vous utilisez.

Exercices guidés

1. Créez un répertoire nommé `emptydir` en utilisant la commande `mkdir emptydir`. Maintenant, en utilisant `ls`, affichez les droits d'accès au répertoire `emptydir`.

2. Créez un fichier vide nommé `emptyfile` avec la commande `touch emptyfile`. Maintenant, en utilisant `chmod` en mode symbolique, ajoutez les droits d'exécution pour le propriétaire du fichier `emptyfile`, et enlevez les droits d'écriture et d'exécution pour tous les autres. Faites tout cela en n'utilisant qu'une seule commande `chmod`.

3. Quels seront les droits d'accès par défaut pour un fichier si la valeur `umask` est fixée à `027` ?

4. Prenons un script shell nommé `test.sh` avec les droits d'accès et les propriétaires suivants :

```
-rwxr-sr-x 1 carol root     33 Dec 11 10:36 test.sh
```

   - Quels sont les droits du propriétaire du fichier ?
   - En utilisant la notation octale, quelle devrait être la syntaxe de `chmod` pour "défaire" les droits d'accès étendus accordés à ce fichier ?

5. Prenons ce fichier :

```
$ ls -l /dev/sdb1
brw-rw---- 1 root disk 8, 17 Dec 21 18:51 /dev/sdb1
```

   Quel type de fichier est `sdb1` ? Qui peut y écrire ?

6. Prenons les quatre fichiers suivants :

```
drwxr-xr-t 2 carol carol 4,0K Dec 20 18:46 Another_Directory
----r--r-- 1 carol carol    0 Dec 11 10:55 foo.bar
-rw-rw-r-- 1 carol carol 1,2G Dec 20 18:22 HugeFile.zip
```

```
drwxr-sr-x 2 carol users  4,0K Jan 18 17:26 Sample_Directory
```

Écrivez les droits correspondants pour chaque fichier et répertoire en utilisant le mode octal avec la notation à 4 chiffres.

`Another_Directory`	
`foo.bar`	
`HugeFile.zip`	
`Sample_Directory`	

Exercices d'approfondissement

1. Essayez ceci dans un terminal : créez un fichier vide appelé `emptyfile` avec la commande `touch emptyfile`. Maintenant, "mettez à zéro" les droits du fichier avec `chmod 000 emptyfile`. Que va-t-il se passer si vous modifiez les droits de `emptyfile` en passant seulement *une* valeur pour `chmod` en mode octal, comme `chmod 4 emptyfile` ? Et si vous en passez deux, comme dans `chmod 44 emptyfile` ? Que pouvons-nous apprendre sur la façon dont `chmod` lit les valeurs numériques ?

2. Prenons les droits d'accès du répertoire temporaire d'un système Linux, `/tmp` :

   ```
   $ ls -l /tmp
   drwxrwxrwt  19 root root  16K Dec 21 18:58 tmp
   ```

   L'utilisateur, le groupe et les autres ont tous les droits. Est-ce qu'un utilisateur normal pourra supprimer *n'importe quel* fichier à l'intérieur de ce répertoire ? Pourquoi en est-il ainsi ?

3. Un fichier nommé `test.sh` a les droits suivants : `-rwsr-xr-x`, ce qui signifie que le bit SUID est activé. Maintenant, exécutez les commandes suivantes :

   ```
   $ chmod u-x test.sh
   $ ls -l test.sh
   -rwSr-xr-x 1 carol carol 33 Dec 11 10:36 test.sh
   ```

   Qu'avons-nous fait ? Que signifie le `S` majuscule ?

4. Comment peut-on créer un répertoire nommé `Box` où tous les fichiers appartiennent automatiquement au groupe `users`, et qui ne peuvent être supprimés que par l'utilisateur qui les a créés ?

Résumé

Dans cette leçon, vous avez appris à utiliser `ls` pour obtenir (et comprendre) des informations sur les droits d'accès aux fichiers, à contrôler ou modifier qui peut créer, supprimer ou modifier un fichier avec `chmod`, à la fois en mode *octal* et *symbolique*, à changer le propriétaire et le groupe des fichiers avec `chown` et `chgrp` et à interroger et modifier le masque de droits d'accès par défaut pour les fichiers et les répertoires avec `umask`.

Les commandes suivantes ont été abordées dans cette leçon :

`ls`
: Afficher la liste des fichiers en indiquant éventuellement les droits d'accès.

`chmod`
: Modifier les droits d'accès d'un fichier ou d'un répertoire.

`chown`
: Modifier le propriétaire et/ou le groupe d'un fichier ou d'un répertoire.

`chgrp`
: Modifier le groupe propriétaire d'un fichier ou d'un répertoire.

`umask`
: Afficher ou définir le masque de permissions par défaut pour les fichiers et les répertoires.

Réponses aux exercices guidés

1. Créez un répertoire nommé `emptydir` en utilisant la commande `mkdir emptydir`. Maintenant, en utilisant `ls`, affichez les droits d'accès au répertoire `emptydir`.

   Ajoutez l'option `-d` à `ls` pour voir les attributs d'un répertoire au lieu d'en afficher le contenu. La réponse est donc :

   ```
   ls -l -d emptydir
   ```

   Un bon point si vous avez songé à fusionner les deux options en une seule, comme dans `ls -ld emptydir`.

2. Créez un fichier vide nommé `emptyfile` avec la commande `touch emptyfile`. Maintenant, en utilisant `chmod` en mode symbolique, ajoutez les droits d'exécution pour le propriétaire du fichier `emptyfile`, et enlevez les droits d'écriture et d'exécution pour tous les autres. Faites tout cela en n'utilisant qu'une seule commande `chmod`.

   Pensez-y de cette manière :

   - "Pour l'utilisateur qui possède le fichier (`u`) ajouter (`+`) les droits d'exécution (`x`)", donc `u+x`.
   - "Pour le groupe (`g`) et les autres utilisateurs (`o`), supprimer (`-`) les droits d'écriture (`w`) et d'exécution (`x`)", donc `go-wx`.

     Pour combiner ces deux jeux de droits, nous ajoutons une virgule entre eux. Le résultat final est donc :

     ```
     chmod u+x,go-wx emptyfile
     ```

3. Quels seront les droits d'accès par défaut pour un fichier si la valeur `umask` est fixée à `027` ?

   Les droits d'accès seraient `rw-r-----`

4. Prenons un script shell nommé `test.sh` avec les droits d'accès et les propriétaires suivants :

   ```
   -rwxr-sr-x 1 carol root     33 Dec 11 10:36 test.sh
   ```

   - Quels sont les droits du propriétaire du fichier ?

     Les droits du propriétaire (2ème au 4ème caractère dans l'affichage de `ls -l`) sont `rwx`,

donc la réponse est : "lire, écrire et exécuter le fichier".

- En utilisant la notation octale, quelle devrait être la syntaxe de `chmod` pour "défaire" les droits d'accès étendus accordés à ce fichier ?

  Nous pouvons "désactiver" les droits d'accès étendus en passant un quatrième chiffre, `0`, à `chmod`. Les droits actuels sont `755`, donc la commande devrait être `chmod 0755`.

5. Prenons ce fichier :

```
$ ls -l /dev/sdb1
brw-rw---- 1 root disk 8, 17 Dec 21 18:51 /dev/sdb1
```

   Quel type de fichier est `sdb1` ? Qui peut y écrire ?

   Le premier caractère de l'affichage de `ls -l` indique le type de fichier. `b` est un *périphérique bloc*, généralement un disque (interne ou externe) connecté à la machine. Le propriétaire (`root`) et tous les membres du groupe `disk` peuvent écrire dessus.

6. Prenons les quatre fichiers suivants :

```
drwxr-xr-t 2 carol carol 4,0K Dec 20 18:46 Another_Directory
----r--r-- 1 carol carol    0 Dec 11 10:55 foo.bar
-rw-rw-r-- 1 carol carol 1,2G Dec 20 18:22 HugeFile.zip
drwxr-sr-x 2 carol users 4,0K Jan 18 17:26 Sample_Directory
```

   Écrivez les droits correspondants pour chaque fichier et répertoire en utilisant le mode octal avec la notation à 4 chiffres.

   Voici les droits correspondants, en mode octal :

`Another_Directory`	`1755`. `1` pour le sticky bit, `755` pour les droits normaux (`rwx` pour l'utilisateur, `r-x` pour le groupe et les autres).
`foo.bar`	`0044`. Pas de droits d'accès étendus (le premier chiffre est donc `0`), pas de droits pour l'utilisateur (`---`) et juste la lecture (`r—r--`) pour le groupe et les autres.

`HugeFile.zip`	`0664`. Pas de droits d'accès étendus, donc le premier chiffre est `0`. `6` (`rw-`) pour l'utilisateur et le groupe, `4` (`r--`) pour les autres.
`Sample_Directory`	`2755`. `2` pour le bit SGID, `7` (`rwx`) pour l'utilisateur, `5` (`r-x`) pour le groupe et les autres.

Réponses aux exercices d'approfondissement

1. Essayez ceci dans un terminal : créez un fichier vide appelé `emptyfile` avec la commande `touch emptyfile`. Maintenant, "mettez à zéro" les droits du fichier avec `chmod 000 emptyfile`. Que va-t-il se passer si vous modifiez les droits de `emptyfile` en passant seulement *une* valeur pour `chmod` en mode octal, comme `chmod 4 emptyfile` ? Et si vous en passez deux, comme dans `chmod 44 emptyfile` ? Que pouvons-nous apprendre sur la façon dont `chmod` lit les valeurs numériques ?

   Rappelez-vous que nous avons "mis à zéro" les permissions de `emptyfile`. Son état initial serait donc :

   ```
   ---------- 1 carol carol    0 Dec 11 10:55 emptyfile
   ```

   Maintenant, essayons la première commande, `chmod 4 emptyfile` :

   ```
   $ chmod 4 emptyfile
   $ ls -l emptyfile
   -------r-- 1 carol carol 0 Dec 11 10:55 emptyfile
   ```

   Vous voyez ? Les droits des autres (*others*) ont été modifiés. Et si nous essayons deux chiffres, comme dans `chmod 44 emptyfile` ?

   ```
   $ chmod 44 emptyfile
   $ ls -l emptyfile
   ----r--r-- 1 carol carol 0 Dec 11 10:55 emptyfile
   ```

   À présent, les droits du groupe (*group*) et des autres (*others*) sont affectés. Nous pouvons en conclure qu'en mode octal, `chmod` lit la valeur "à l'envers", du chiffre le moins significatif (*others*) au plus significatif (*user*). Si vous passez un chiffre, vous modifiez les droits pour les autres (*others*). Avec deux chiffres, vous modifiez le groupe (*group*) et les autres (*others*). Avec trois, vous modifiez l'utilisateurs (*user*), le groupe (*group*) et les autres (*others*). Et avec quatre chiffres, vous modifiez l'utilisateur (*user*), le groupe (*group*), les autres (*others*) et les droits d'accès étendus.

2. Prenons les droits d'accès du répertoire temporaire d'un système Linux, `/tmp` :

   ```
   $ ls -l /tmp
   ```

```
drwxrwxrwt  19 root root  16K Dec 21 18:58 tmp
```

L'utilisateur, le groupe et les autres ont tous les droits. Est-ce qu'un utilisateur normal pourra supprimer *n'importe quel* fichier à l'intérieur de ce répertoire ? Pourquoi en est-il ainsi ?

`/tmp` est ce que nous appelons un répertoire aux droits d'écriture universels (*world writeable*), ce qui signifie que n'importe quel utilisateur peut y écrire. Or, nous ne voulons pas qu'un utilisateur s'amuse avec des fichiers créés par d'autres, c'est pourquoi le *sticky bit* est activé (comme indiqué par le `t` sur les permissions pour *others*). Cela signifie qu'un utilisateur peut supprimer des fichiers sur `/tmp`, mais seulement ceux qu'il a créés lui-même.

3. Un fichier nommé `test.sh` a les droits suivants : `-rwsr-xr-x`, ce qui signifie que le bit SUID est activé. Maintenant, exécutez les commandes suivantes :

```
$ chmod u-x test.sh
$ ls -l test.sh
-rwSr-xr-x 1 carol carol 33 Dec 11 10:36 test.sh
```

Qu'avons-nous fait ? Que signifie le `S` majuscule ?

Nous avons supprimé les droits d'exécution pour l'utilisateur propriétaire du fichier. Le `s` (ou `t`) prend la place du `x` dans le résultat de `ls -l`, donc le système a besoin d'un moyen pour montrer si l'utilisateur a les permissions d'exécution ou non. Il le fait en changeant la casse du caractère spécial.

Un `s` minuscule sur le premier lot de permissions signifie que l'utilisateur qui possède le fichier a les droits d'exécution et que le bit SUID est activé. Un `S` majuscule signifie que l'utilisateur qui possède le fichier n'a pas les droits d'exécution (`-`) et que le bit SUID est activé.

Il en va de même pour le SGID. Un `s` minuscule sur le deuxième lot de permissions signifie que le groupe propriétaire du fichier a des droits d'exécution et que le bit SGID est activé. Un `S` majuscule signifie que le groupe propriétaire du fichier n'a pas les droits d'exécution (`-`) et que le bit SGID est activé.

Ce principe s'applique également au *sticky bit*, symbolisé par le `t` dans le troisième lot de permissions. Le `t` minuscule signifie que le *sticky bit* est activé et que les autres ont les droits d'exécution. Le `T` majuscule signifie que le *sticky bit* est activé et que les autres n'ont pas les droits d'exécution.

4. Comment peut-on créer un répertoire nommé `Box` où tous les fichiers appartiennent automatiquement au groupe `users`, et qui ne peuvent être supprimés que par l'utilisateur qui

les a créés ?

C'est un processus en plusieurs étapes. La première étape consiste à créer le répertoire :

```
$ mkdir Box
```

Nous voulons que chaque fichier créé dans ce répertoire soit automatiquement assigné au groupe `users`. Nous pouvons le faire en définissant ce groupe comme propriétaire du répertoire, puis en fixant le bit SGID sur ce répertoire. Nous devons également nous assurer que tous les membres du groupe peuvent écrire dans ce répertoire.

Puisque nous ne nous préoccupons pas des autres droits, et que nous voulons "basculer" uniquement les droits d'accès spéciaux, il est logique d'utiliser le mode symbolique :

```
$ chown :users Box/
$ chmod g+wxs Box/
```

Notez que si votre utilisateur actuel n'appartient pas au groupe `users`, vous devrez utiliser la commande `sudo` avant les commandes ci-dessus pour effectuer le changement en tant que root.

Passons maintenant à la dernière partie, qui consiste à s'assurer que seul l'utilisateur qui a créé un fichier est autorisé à l'effacer. Cela se fait en paramétrant le *sticky bit* (représenté par un `t`) sur le répertoire. Rappelez-vous qu'il est défini dans les permissions pour les autres (`o`).

```
$ chmod o+t Box/
```

Les droits d'accès au répertoire `Box` doivent être les suivants :

```
drwxrwsr-t 2 carol users  4,0K Jan 18 19:09 Box
```

Bien sûr, vous pouvez spécifier le SGID et le sticky bit en utilisant une seule commande `chmod` :

```
$ chmod g+wxs,o+t Box/
```

Un bon point pour vous si vous y avez pensé.

104.6 Création et modification des liens physiques et symboliques sur les fichiers

Référence aux objectifs de LPI

LPIC-1 version 5.0, Exam 101, Objective 104.6

Valeur

2

Domaines de connaissance les plus importants

- Création des liens.
- Identification des liens physiques et/ou symboliques.
- Copie versus liens vers les fichiers.
- Utilisation des liens pour les tâches d'administration système.

Liste partielle de termes, fichiers et utilitaires utilisés pour cet objectif

- `ln`
- `ls`

Linux Professional Institute

104.6 Leçon 1

Certification :	LPIC-1
Version :	5.0
Thème :	104 Disques, systèmes de fichiers Linux, arborescence de fichiers standard (FHS)
Objectif :	104.6 Créer et modifier des liens physiques et symboliques
Leçon :	1 sur 1

Introduction

Sous Linux, certains fichiers sont gérés différemment, soit du fait de l'endroit où ils sont stockés, comme les fichiers temporaires, soit de la manière dont ils interagissent avec le système de fichiers, comme les liens. Dans cette leçon, vous apprendrez ce que sont les liens et comment les gérer.

Comprendre les liens

Comme nous l'avons déjà dit, tout est traité comme un fichier sous Linux. Mais il existe un type spécial de fichier, appelé *lien*, et il y a deux types de liens sur un système Linux :

Liens symboliques

Également appelés *soft links*, ils pointent vers le chemin d'accès d'un autre fichier. Si vous supprimez le fichier vers lequel pointe le lien (appelé *cible*), ce dernier existera toujours, mais il ne "fonctionnera plus", car il pointera désormais vers "rien".

Liens physiques

Un lien physique est un deuxième nom pour le fichier d'origine. Il ne s'agit pas d'un doublon, mais d'une entrée supplémentaire dans le système de fichiers qui pointe vers le même emplacement (*inode*) sur le disque.

TIP

Un *inode* est une structure de données qui stocke les attributs d'un objet (comme un fichier ou un répertoire) dans un système de fichiers. Parmi ces attributs figurent les droits d'accès, les propriétaires et les blocs du disque sur lesquels les données de l'objet sont stockées. Il s'agit d'une entrée dans un index, d'où le nom qui vient de "nœud d'index" (*index node*).

Travailler avec les liens physiques

Créer des liens physiques

La commande pour créer un lien physique sous Linux est `ln`. Voici la syntaxe de base :

```
$ ln TARGET LINK_NAME
```

La cible `TARGET` doit déjà exister (c'est le fichier vers lequel le lien pointera), et si la cible n'est pas dans le répertoire courant, ou si vous voulez créer le lien ailleurs, vous devez impérativement spécifier le chemin complet vers ce fichier. Par exemple, la commande :

```
$ ln target.txt /home/carol/Documents/hardlink
```

va créer un fichier nommé `hardlink` dans le répertoire `/home/carol/Documents/`, lié au fichier `target.txt` dans le répertoire courant.

Si vous omettez le dernier paramètre (`LINK_NAME`), un lien portant le même nom que la cible sera créé dans le répertoire courant.

Gérer les liens physiques

Les liens physiques sont des entrées du système de fichiers avec des noms différents mais qui pointent vers les mêmes données sur le disque. Tous ces noms se valent et peuvent être utilisés pour faire référence à un fichier. Si vous modifiez le contenu de l'un des noms, le contenu de tous les autres noms pointant vers ce fichier change puisque tous les noms pointent vers les mêmes données. Si vous supprimez l'un des noms, les autres noms continueront à fonctionner.

En effet, lorsque vous "supprimez" un fichier, les données ne sont pas réellement effacées du

disque. Le système supprime simplement l'entrée de la table du système de fichiers pointant vers l'*inode* correspondant aux données sur le disque. Mais si vous avez une deuxième entrée pointant vers le même *inode*, vous pouvez toujours accéder aux données. Imaginez deux routes qui convergent vers le même point. Même si vous bloquez ou redirigez l'une des routes, vous pouvez toujours atteindre la destination en utilisant l'autre.

Vous pouvez vérifier en utilisant l'option `-i` de `ls`. Prenons le contenu suivant d'un répertoire :

```
$ ls -li
total 224
3806696 -r--r--r-- 2 carol carol 111702 Jun  7 10:13 hardlink
3806696 -r--r--r-- 2 carol carol 111702 Jun  7 10:13 target.txt
```

Le nombre qui précède les droits d'accès est le numéro de l'*inode*. Vous voyez que le fichier `hardlink` et le fichier `target.txt` ont le même numéro (`3806696`) ? C'est parce que l'un est un lien physique de l'autre.

Mais lequel est l'original et lequel est le lien ? Impossible de le savoir, dans la mesure où les deux sont identiques sur le plan pratique.

Notez que chaque lien physique pointant vers un fichier augmente le compte de liens (*link count*) du fichier. C'est le nombre situé juste après les droits d'accès dans l'affichage de `ls -l`. Par défaut, chaque fichier a un compte de liens de `1` (les répertoires ont un compte de `2`), et chaque lien physique pointant vers lui augmente d'un cran ce compte. Ce qui explique le nombre de liens `2` pour les fichiers de la liste ci-dessus.

Contrairement aux liens symboliques, vous pouvez créer des liens physiques vers des fichiers uniquement, et le lien et la cible doivent résider dans le même système de fichiers.

Déplacer et supprimer des liens physiques

Étant donné que les liens physiques sont traités comme des fichiers normaux, ils peuvent être supprimés avec `rm` et renommés ou déplacés dans le système de fichiers avec `mv`. Et comme un lien physique pointe vers le même *inode* que la cible, il peut être déplacé librement, sans risquer de "casser" le lien.

Les liens symboliques

Créer des liens symboliques

La commande utilisée pour créer un lien symbolique est également `ln`, mais avec l'ajout de

l'option `-s`. Voici ce que ça donne :

```
$ ln -s target.txt /home/carol/Documents/softlink
```

Cette opération va créer un fichier nommé `softlink` dans le répertoire `/home/carol/Documents/`, pointant vers le fichier `target.txt` dans le répertoire courant.

Tout comme pour les liens physiques, vous pouvez omettre le nom du lien en argument pour créer un lien qui porte le même nom que la cible dans le répertoire courant.

Gérer les liens symboliques

Les liens symboliques pointent vers un autre chemin dans le système de fichiers. Vous pouvez créer des liens symboliques vers des fichiers aussi bien que des répertoires, même sur des partitions différentes. Il est assez facile de repérer un lien symbolique dans l'affichage de la commande `ls` :

```
$ ls -lh
total 112K
-rw-r--r-- 1 carol carol 110K Jun  7 10:13 target.txt
lrwxrwxrwx 1 carol carol   12 Jun  7 10:14 softlink -> target.txt
```

Dans l'exemple ci-dessus, le premier caractère avant les droits d'accès au fichier `softlink` est `l`, ce qui signifie qu'il s'agit d'un lien symbolique. Par ailleurs, juste après le nom du fichier, vous voyez le nom de la cible vers laquelle pointe le lien, en l'occurrence le fichier `target.txt`.

Notez que dans les listings de fichiers et de répertoires, les liens symboliques eux-mêmes affichent toujours les droits `rwx` pour l'utilisateur, le groupe et les autres, mais en pratique leurs droits d'accès correspondent à ceux de la cible.

Déplacer et supprimer des liens symboliques

Comme les liens physiques, les liens symboliques peuvent être supprimés avec `rm` et déplacés ou renommés avec `mv`. Cependant, il faut faire très attention lors de leur création, pour éviter de "casser" le lien lorsqu'on le déplace depuis son emplacement d'origine.

Lorsque vous créez des liens symboliques, sachez qu'à moins qu'un chemin d'accès ne soit complètement spécifié, l'emplacement de la cible est interprété comme étant *relatif* à l'emplacement du lien. Ce qui peut poser des problèmes quand on déplace le lien ou le fichier vers lequel il pointe.

Un exemple permet de mieux comprendre ce principe. Supposons que vous ayez un fichier nommé `original.txt` dans le répertoire courant, et que vous vouliez créer un lien symbolique appelé `softlink` vers ce fichier. Vous pourriez utiliser :

```
$ ln -s original.txt softlink
```

Apparemment, tout va bien. Vérifions avec `ls` :

```
$ ls -lh
total 112K
-r--r--r-- 1 carol carol 110K Jun  7 10:13 original.txt
lrwxrwxrwx 1 carol carol   12 Jun  7 19:23 softlink -> original.txt
```

Voyez comment le lien est construit : `softlink` pointe vers (→) `original.txt`. Maintenant, voyons ce qui se passe lorsque vous déplacez le lien vers le répertoire parent et que vous essayez d'afficher son contenu en utilisant la commande `less` :

```
$ mv softlink ../
$ less ../softlink
../softlink: No such file or directory
```

Comme le chemin vers `original.txt` n'a pas été spécifié, le système part du principe qu'il se situe dans le même répertoire que le lien. À partir du moment où ce n'est plus le cas, le lien ne fonctionne plus.

Pour éviter cette situation, il suffit de spécifier le chemin d'accès complet à la cible lors de la création du lien :

```
$ ln -s /home/carol/Documents/original.txt softlink
```

Ainsi, quel que soit l'endroit où vous déplacez le lien, il fonctionnera toujours, étant donné qu'il pointe vers l'emplacement absolu de la cible. Vérifiez avec `ls` :

```
$ ls -lh
total 112K
lrwxrwxrwx 1 carol carol   40 Jun  7 19:34 softlink -> /home/carol/Documents/original.txt
```

Exercices guidés

1. Quel est le paramètre de `chmod` en mode *symbolique* pour activer le *sticky bit* sur un répertoire ?

2. Prenons un fichier nommé `document.txt` dans le répertoire `/home/carol/Documents`. Quelle est la commande pour créer un lien symbolique vers ce fichier nommé `text.txt` dans le répertoire courant ?

3. Expliquez la différence entre un lien physique vers un fichier et une copie de ce fichier.

Exercices d'approfondissement

1. Imaginez que dans un répertoire vous créez un fichier appelé `recipes.txt`. À l'intérieur de ce répertoire, vous allez également créer un lien physique vers ce fichier, appelé `receitas.txt`, et un lien symbolique (*soft link*) vers ce fichier, appelé `rezepte.txt`.

   ```
   $ touch recipes.txt
   $ ln recipes.txt receitas.txt
   $ ln -s recipes.txt rezepte.txt
   ```

   Le contenu du répertoire devrait ressembler à ceci :

   ```
   $ ls -lhi
   total 160K
   5388833 -rw-r--r-- 4 carol carol 77K jun 17 17:25 receitas.txt
   5388833 -rw-r--r-- 4 carol carol  0K jun 17 17:25 recipes.txt
   5388837 lrwxrwxrwx 1 carol carol  12 jun 17 17:25 rezepte.txt -> receitas.txt
   ```

   Rappelez-vous qu'en tant que lien physique, `receitas.txt` pointe vers le même *inode* que `recipes.txt`. Que va devenir le lien symbolique `rezepte.txt` si l'on supprime le fichier `receitas.txt` ? Pourquoi ?

2. Imaginez que vous ayez une clé USB branchée sur votre système, montée sur `/media/youruser/FlashA`. Vous voulez créer un lien nommé `schematics.pdf` dans votre répertoire personnel, et qui pointe vers le fichier `esquema.pdf` à la racine de la clé USB. Vous tapez donc la commande :

   ```
   $ ln /media/youruser/FlashA/esquema.pdf ~/schematics.pdf
   ```

   Que se passerait-il ? Pourquoi ?

3. Considérez l'affichage suivant de `ls -lah` :

   ```
   $ ls -lah
   total 3,1M
   drwxr-xr-x 2 carol carol 4,0K jun 17 17:27 .
   drwxr-xr-x 5 carol carol 4,0K jun 17 17:29 ..
   -rw-rw-r-- 1 carol carol 2,8M jun 17 15:45 compressed.zip
   -rw-r--r-- 4 carol carol  77K jun 17 17:25 document.txt
   -rw-rw-r-- 1 carol carol 216K jun 17 17:25 image.png
   ```

```
-rw-r--r-- 4 carol carol  77K jun 17 17:25 text.txt
```

- Combien de liens pointent vers le fichier `document.txt` ?

- S'agit-il de liens symboliques ou de liens physiques ?

- Quelle option devez-vous passer à `ls` pour savoir à quel *inode* correspond chaque fichier ?

4. Imaginez que vous ayez dans votre répertoire `~/Documents` un fichier nommé `clients.txt` avec quelques noms de clients, et un répertoire nommé `somedir`. À l'intérieur de ce répertoire, il y a un fichier *différent* également nommé `clients.txt`, avec des noms distincts. Vous pouvez reproduire cette structure avec les commandes suivantes.

```
$ cd ~/Documents
$ echo "John, Michael, Bob" > clients.txt
$ mkdir somedir
$ echo "Bill, Luke, Karl" > somedir/clients.txt
```

Ensuite, vous créez un lien nommé `partners.txt` dans `somedir` et qui pointe vers ce fichier, avec les commandes suivantes :

```
$ cd somedir/
$ ln -s clients.txt partners.txt
```

La structure de l'arborescence est donc :

```
Documents
|-- clients.txt
`-- somedir
    |-- clients.txt
    `-- partners.txt -> clients.txt
```

Maintenant, vous déplacez `partners.txt` de `somedir` vers `~/Documents`, et vous affichez son contenu.

```
$ cd ~/Documents/
```

```
$ mv somedir/partners.txt .
$ less partners.txt
```

Est-ce que le lien sera toujours fonctionnel ? Si oui, quel fichier verra son contenu affiché ? Pourquoi ?

5. Prenez les fichiers suivants :

```
-rw-r--r-- 1 carol carol 19 Jun 24 11:12 clients.txt
lrwxrwxrwx 1 carol carol 11 Jun 24 11:13 partners.txt -> clients.txt
```

Quels sont les droits d'accès pour `partners.txt` ? Pourquoi ?

Résumé

Dans cette leçon, vous avez appris à :

- Savoir ce que c'est qu'un lien.
- Comprendre la différence entre un lien *symbolique* et un lien *physique*.
- Créer des liens.
- Déplacer, renommer et supprimer ces liens.

Les commandes suivantes ont été abordées dans cette leçon :

- `ln` : La commande qui permet de créer un lien. En elle-même, la commande crée un lien physique. L'option `-s` permet de créer un lien *symbolique* (*soft link*). Rappelez-vous que les liens physiques ne peuvent résider que sur la même partition et le même système de fichiers, et que les liens symboliques peuvent aller au-delà d'une partition et d'un système de fichiers (et même d'un stockage en réseau).
- L'option `-i` de `ls`, qui permet d'afficher le numéro d'*inode* d'un fichier.

Réponses aux exercices guidés

1. Quel est le paramètre de `chmod` en mode *symbolique* pour activer le *sticky bit* sur un répertoire ?

   Le symbole du *sticky bit* en mode symbolique est `t`. Puisque nous voulons activer (ajouter) cette permission au répertoire, le paramètre doit être `+t`.

2. Prenons un fichier nommé `document.txt` dans le répertoire `/home/carol/Documents`. Quelle est la commande pour créer un lien symbolique vers ce fichier nommé `text.txt` dans le répertoire courant ?

   `ln -s` est la commande pour créer un lien symbolique. Comme vous devriez spécifier le chemin complet du fichier vers lequel vous créez un lien, voici la commande :

   ```
   $ ln -s /home/carol/Documents/document.txt text.txt
   ```

3. Expliquez la différence entre un lien physique vers un fichier et une copie de ce fichier.

   Un lien physique est simplement un autre nom pour un fichier. Même s'il ressemble à un doublon du fichier original, les deux sont identiques, puisqu'ils renvoient aux mêmes données sur le disque. Les modifications apportées au contenu du lien seront répercutées sur l'original, et vice-versa. En revanche, une copie est une entité complètement indépendante, qui occupe un emplacement différent sur le disque. Les modifications apportées à la copie ne seront pas répercutées sur l'original, et vice-versa.

Réponses aux exercices d'approfondissement

1. Imaginez que dans un répertoire vous créez un fichier appelé `recipes.txt`. À l'intérieur de ce répertoire, vous allez également créer un lien physique vers ce fichier, appelé `receitas.txt`, et un lien symbolique (*soft link*) vers ce fichier, appelé `rezepte.txt`.

   ```
   $ touch recipes.txt
   $ ln recipes.txt receitas.txt
   $ ln -s receitas.txt rezepte.txt
   ```

   Le contenu du répertoire devrait ressembler à ceci :

   ```
   $ ls -lhi
   total 160K
   5388833 -rw-r--r-- 4 carol carol 77K jun 17 17:25 receitas.txt
   5388833 -rw-r--r-- 4 carol carol  0K jun 17 17:25 recipes.txt
   5388837 lrwxrwxrwx 1 carol carol  12 jun 17 17:25 rezepte.txt -> receitas.txt
   ```

   Rappelez-vous qu'en tant que lien physique, `receitas.txt` pointe vers le même *inode* que `recipes.txt`. Que va devenir le lien symbolique `rezepte.txt` si l'on supprime le fichier `receitas.txt` ? Pourquoi ?

   Le lien symbolique `rezepte.txt` cesserait de fonctionner. En effet, les liens symboliques pointent vers des noms, et non pas vers des *inodes*, et le nom `receitas.txt` n'existe plus, même si les données sont toujours sur le disque sous le nom `recipes.txt`.

2. Imaginez que vous ayez une clé USB branchée sur votre système, montée sur `/media/youruser/FlashA`. Vous voulez créer un lien nommé `schematics.pdf` dans votre répertoire personnel, et qui pointe vers le fichier `esquema.pdf` à la racine de la clé USB. Vous tapez donc la commande :

   ```
   $ ln /media/youruser/FlashA/esquema.pdf ~/schematics.pdf
   ```

   Que se passerait-il ? Pourquoi ?

   La commande échouerait. Le message d'erreur serait `Invalid cross-device link`, et la raison est évidente : les liens physiques ne peuvent pas pointer vers une cible située dans une partition ou sur un périphérique différent. La seule façon de créer un tel lien est d'utiliser un lien *symbolique* (*soft link*) en ajoutant l'option `-s` à `ln`.

3. Considérez l'affichage suivant de `ls -lah` :

```
$ ls -lah
total 3,1M
drwxr-xr-x 2 carol carol 4,0K jun 17 17:27 .
drwxr-xr-x 5 carol carol 4,0K jun 17 17:29 ..
-rw-rw-r-- 1 carol carol 2,8M jun 17 15:45 compressed.zip
-rw-r--r-- 4 carol carol  77K jun 17 17:25 document.txt
-rw-rw-r-- 1 carol carol 216K jun 17 17:25 image.png
-rw-r--r-- 4 carol carol  77K jun 17 17:25 text.txt
```

   - Combien de liens pointent vers le fichier `document.txt` ?

     Chaque fichier commence avec un décompte de liens de `1`. Puisque le nombre de liens pour ce fichier est `4`, il y a trois liens qui pointent vers ce fichier.

   - S'agit-il de liens symboliques ou de liens physiques ?

     Ce sont des liens physiques, étant donné que les liens symboliques n'apparaissent pas dans le décompte des liens d'un fichier.

   - Quelle option devez-vous passer à `ls` pour savoir à quel *inode* correspond chaque fichier ?

     L'option `-i`. L'*inode* s'affichera dans la première colonne de l'affichage de `ls`, comme ceci :

```
$ ls -lahi
total 3,1M
5388773 drwxr-xr-x 2 carol carol 4,0K jun 17 17:27 .
5245554 drwxr-xr-x 5 carol carol 4,0K jun 17 17:29 ..
5388840 -rw-rw-r-- 1 carol carol 2,8M jun 17 15:45 compressed.zip
5388833 -rw-r--r-- 4 carol carol  77K jun 17 17:25 document.txt
5388837 -rw-rw-r-- 1 carol carol 216K jun 17 17:25 image.png
5388833 -rw-r--r-- 4 carol carol  77K jun 17 17:25 text.txt
```

4. Imaginez que vous ayez dans votre répertoire `~/Documents` un fichier nommé `clients.txt` avec quelques noms de clients, et un répertoire nommé `somedir`. À l'intérieur de ce répertoire, il y a un fichier *différent* également nommé `clients.txt`, avec des noms distincts. Vous pouvez reproduire cette structure avec les commandes suivantes.

```
$ cd ~/Documents
$ echo "John, Michael, Bob" > clients.txt
```

```
$ mkdir somedir
$ echo "Bill, Luke, Karl" > somedir/clients.txt
```

Ensuite, vous créez un lien nommé `partners.txt` dans `somedir` et qui pointe vers ce fichier, avec les commandes suivantes :

```
$ cd somedir/
$ ln -s clients.txt partners.txt
```

La structure de l'arborescence est donc :

```
Documents
|-- clients.txt
`-- somedir
    |-- clients.txt
    `-- partners.txt -> clients.txt
```

Maintenant, vous déplacez `partners.txt` de `somedir` vers `~/Documents`, et vous affichez son contenu.

```
$ cd ~/Documents/
$ mv somedir/partners.txt .
$ less partners.txt
```

Est-ce que le lien sera toujours fonctionnel ? Si oui, quel fichier verra son contenu affiché ? Pourquoi ?

Ce cas de figure est un peu épineux, mais le lien sera fonctionnel. Le fichier affiché sera celui qui se trouve dans `~/Documents`, avec les noms `John`, `Michael`, `Bob`.

Rappelez-vous que puisque vous n'avez pas spécifié le chemin complet de la cible `clients.txt` lors de la création du lien symbolique `partners.txt`, l'emplacement de la cible sera interprété comme étant relatif à l'emplacement du lien, qui est ici le répertoire courant.

Lorsque le lien a été déplacé de `~/Documents/somedir` vers `~/Documents`, il aurait dû cesser de fonctionner, puisque la cible n'était plus dans le même répertoire que le lien. Cependant, il se trouve qu'il y a un autre fichier nommé `clients.txt` dans `~/Documents`. Le lien pointera donc vers ce fichier au lieu de la cible originale dans `~/somedir`.

Pour éviter ce genre de situation, indiquez toujours le chemin d'accès complet vers la cible

lorsque vous créez un lien symbolique.

5. Prenez les fichiers suivants :

```
-rw-r--r-- 1 carol carol 19 Jun 24 11:12 clients.txt
lrwxrwxrwx 1 carol carol 11 Jun 24 11:13 partners.txt -> clients.txt
```

Quels sont les droits d'accès pour `partners.txt` ? Pourquoi ?

Les droits d'accès pour `partners.txt` sont `rw-r—r--`, étant donné que les liens héritent toujours des mêmes permissions que la cible.

104.7 Recherche de fichiers et placement des fichiers aux endroits adéquats

Référence aux objectifs de LPI

LPIC-1 version 5.0, Exam 101, Objective 104.7

Valeur

2

Domaines de connaissance les plus importants

- Compréhension de l'emplacement correct des fichiers dans le FHS.
- Recherche de fichiers et de commandes sur un système Linux
- Connaissance de l'emplacement et du but des fichiers et des répertoires importants tels que définis dans la FHS.

Liste partielle de termes, fichiers et utilitaires utilisés pour cet objectif

- `find`
- `locate`
- `updatedb`
- `whereis`
- `which`
- `type`
- `/etc/updatedb.conf`

104.7 Leçon 1

Certification :	LPIC-1
Version :	5.0
Thème :	104 Disques, systèmes de fichiers Linux, arborescence de fichiers standard (FHS)
Objectif :	104.7 Rechercher des fichiers et ranger des fichiers aux emplacements appropriés
Leçon :	1 sur 1

Introduction

Les distributions Linux se déclinent sous plusieurs formats, mais elles partagent presque toutes le *Filesystem Hierarchy Standard* (FHS), une norme qui définit une "disposition standard" pour le système de fichiers, ce qui facilite considérablement l'interopérabilité et l'administration du système. Dans cette leçon, vous en apprendrez plus sur cette norme et sur la manière de retrouver des fichiers sur un système Linux.

L'arborescence de fichiers standard (FHS)

L'arborescence de fichiers standard (FHS) est un projet de la Fondation Linux visant à normaliser la structure et le contenu des répertoires sur les systèmes Linux. Le respect de cette norme n'est pas obligatoire, mais la plupart des distributions s'y conforment.

NOTE Ceux et celles qui s'intéressent aux détails de l'organisation du système de fichiers peuvent consulter la spécification FHS 3.0, disponible dans plusieurs formats à l'adresse suivante : http://refspecs.linuxfoundation.org/fhs.shtml.

Selon la convention, la structure de base des répertoires se présente comme suit :

`/`

Le répertoire racine est le répertoire le plus élevé de la hiérarchie. Tous les autres répertoires sont situés à l'intérieur de celui-ci. Un système de fichiers est souvent comparé à un "arbre", il s'agit donc du "tronc" auquel toutes les branches sont connectées.

`/bin`

Binaires indispensables, disponibles pour tous les utilisateurs.

`/boot`

Fichiers nécessaires au processus de démarrage, y compris le disque mémoire initial (`initrd`) et le noyau Linux lui-même.

`/dev`

Fichiers de périphériques. Il peut s'agir de périphériques physiques connectés au système (par exemple, `/dev/sda` serait le premier disque SCSI ou SATA) ou de périphériques virtuels créés par le noyau.

`/etc`

Fichiers de configuration propres à la machine. Les programmes peuvent créer des sous-répertoires sous `/etc` pour stocker une multitude de fichiers de configuration en cas de besoin.

`/home`

Chaque utilisateur du système dispose d'un répertoire "chez lui" pour stocker ses fichiers personnels et ses préférences, et la plupart d'entre eux sont situés dans `/home`. En règle générale, le répertoire personnel est le même que le nom d'utilisateur, de sorte que l'utilisateur John aura son répertoire sous `/home/john`. Les exceptions notables sont le superutilisateur (`root`) qui a son propre répertoire (`/root`) et certains utilisateurs du système.

`/lib`

Bibliothèques partagées nécessaires au démarrage du système d'exploitation et à l'exécution des binaires sous `/bin` et `/sbin`.

`/media`

Les supports amovibles montés par l'utilisateur, tels que les clés USB, les lecteurs de CD et de DVD-ROM, les disquettes, les cartes mémoire et les disques externes, sont montés à cet endroit.

`/mnt`

Point de montage pour les systèmes de fichiers montés temporairement.

`/opt`

Paquets de logiciels d'application.

`/root`

Répertoire personnel du superutilisateur (`root`).

`/run`

Données variables d'exécution.

`/sbin`

Binaires du système.

`/srv`

Données servies par le système. Par exemple, les pages servies par un serveur web pourront être stockées sous `/srv/www`.

`/tmp`

Fichiers temporaires.

`/usr`

Données utilisateur en lecture seule, y compris les données requises par certains outils et certaines applications secondaires.

`/proc`

Système de fichiers virtuel contenant des données liées aux processus en cours.

`/var`

Données variables enregistrées pendant le fonctionnement du système, y compris la file d'attente d'impression, les données de journalisation, les boîtes aux lettres, les fichiers temporaires, le cache du navigateur, etc.

Gardez à l'esprit que certains de ces répertoires, comme `/etc`, `/usr` et `/var`, renferment toute une hiérarchie de sous-répertoires.

Fichiers temporaires

Les fichiers temporaires sont des fichiers utilisés par les programmes pour stocker des données qui ne sont utilisées que pendant une période limitée. Il peut s'agir des données des processus en cours, des journaux de plantage, des fichiers de récupération d'une sauvegarde automatique, des fichiers intermédiaires utilisés lors de la conversion d'un fichier, des fichiers de cache, etc.

Emplacement des fichiers temporaires

La version 3.0 du *Filesystem Hierarchy Standard* (FHS) définit des emplacements standard pour les fichiers temporaires sur les systèmes Linux. Chaque emplacement a sa propre finalité et son propre comportement, et il est recommandé aux développeurs de suivre les conventions définies par le FHS lorsqu'ils écrivent des données temporaires sur le disque.

`/tmp`

D'après le FHS, les programmes ne doivent pas partir du principe que les fichiers écrits dans ce répertoire seront conservés entre deux invocations d'un programme. Il est recommandé de vider ce répertoire (en effaçant tous les fichiers) lors du démarrage du système, même si cela n'est pas obligatoire.

`/var/tmp`

Un autre emplacement pour les fichiers temporaires, mais celui-ci *ne doit pas être effacé* lors du démarrage du système. Les fichiers stockés ici persistent généralement entre les redémarrages.

`/run`

Ce répertoire contient les données variables d'exécution utilisées par les processus en cours, telles que les fichiers d'identification de processus (`.pid`). Les programmes qui ont besoin de plus d'un fichier d'exécution peuvent créer des sous-répertoires ici. Cet emplacement *doit être nettoyé* lors du démarrage du système. La fonction de ce répertoire était autrefois assumée par `/var/run`, et sur certains systèmes, `/var/run` peut constituer un lien symbolique vers `/run`.

Notez que rien n'empêche un programme de créer des fichiers temporaires ailleurs sur le système, mais il est de bon ton de respecter les conventions établies par le FHS.

Recherche de fichiers

Pour rechercher des fichiers sur un système Linux, vous pouvez utiliser la commande `find`. Il s'agit d'un outil très puissant, doté de nombreux paramètres qui permettent d'adapter son comportement et de modifier les résultats en fonction de vos besoins.

Pour commencer, `find` a besoin de deux arguments : un point de départ et ce qu'il faut chercher. Par exemple, pour rechercher tous les fichiers dans le répertoire courant (et les sous-répertoires) dont le nom se termine par `.jpg`, vous pouvez utiliser :

```
$ find . -name '*.jpg'
./pixel_3a_seethrough_1.jpg
./Mate3.jpg
./Expert.jpg
```

```
./Pentaro.jpg
./Mate1.jpg
./Mate2.jpg
./Sala.jpg
./Hotbit.jpg
```

Ceci va correspondre à n'importe quel fichier dont les quatre derniers caractères du nom sont `.jpg`, indépendamment de ce qui précède, puisque `*` est un joker pour "n'importe quoi". Cependant, voyez ce qui se passe lorsqu'un autre `*` est ajouté à la *fin* du motif :

```
$ find . -name '*.jpg*'
./pixel_3a_seethrough_1.jpg
./Pentaro.jpg.zip
./Mate3.jpg
./Expert.jpg
./Pentaro.jpg
./Mate1.jpg
./Mate2.jpg
./Sala.jpg
./Hotbit.jpg
```

Le fichier `Pentaro.jpg.zip` (mis en évidence ci-dessus) n'a pas été inclus dans la liste précédente, étant donné que même s'il contient `.jpg` dans son nom, il ne correspondait pas au motif car il y avait des caractères supplémentaires après. Le nouveau motif signifie "n'importe quoi `.jpg` n'importe quoi", donc il correspond.

TIP Gardez à l'esprit que le paramètre `-name` est sensible à la casse. Si vous souhaitez effectuer une recherche insensible à la casse, utilisez `-iname`.

L'expression `*.jpg` doit être placée entre guillemets simples, pour éviter que le shell n'interprète le motif lui-même. Essayez sans les guillemets et voyez ce qui se passe.

Par défaut, `find` va commencer au point de départ et descendre dans tous les sous-répertoires (et les sous-répertoires de ces sous-répertoires) trouvés. Vous pouvez restreindre ce comportement avec les paramètres `-maxdepth N`, où `N` représente le nombre maximum de niveaux.

Pour effectuer une recherche uniquement dans le répertoire courant, vous utiliserez `-maxdepth 1`. Supposons que vous ayez la structure de répertoire suivante :

```
directory
├── clients.txt
```

```
├── partners.txt -> clients.txt
└── somedir
    ├── anotherdir
    └── clients.txt
```

Pour effectuer une recherche dans `somedir`, vous devez utiliser `-maxdepth 2` (le répertoire courant +1 niveau plus bas). Pour chercher dans `anotherdir`, il faut utiliser `-maxdepth 3` (le répertoire courant +2 niveaux plus bas). Le paramètre `-mindepth N` fonctionne à l'inverse en ne recherchant que dans les répertoires *au moins* `N` niveaux en dessous.

Le paramètre `-mount` peut être utilisé pour éviter que `find` ne descende à l'intérieur des systèmes de fichiers montés. Vous pouvez également restreindre la recherche à certains types de systèmes de fichiers en utilisant le paramètre `-fstype`. Ainsi, `find /mnt -fstype exfat -iname "*report*"` ne cherchera que dans les systèmes de fichiers exFAT montés en dessous de `/mnt`.

Recherche par caractéristiques

Vous pouvez utiliser les paramètres ci-dessous pour rechercher des fichiers ayant des caractéristiques spécifiques, comme ceux qui sont accessibles en écriture pour votre utilisateur, ceux qui disposent d'un jeu de permissions spécifique ou encore ceux d'une taille donnée :

`-user USERNAME`
: Recherche les fichiers appartenant à l'utilisateur `USERNAME`.

`-group GROUPNAME`
: Recherche les fichiers appartenant au groupe `GROUPNAME`.

`-readable`
: Recherche les fichiers accessibles en lecture pour l'utilisateur actuel.

`-writable`
: Recherche les fichiers accessibles en écriture pour l'utilisateur actuel.

`-executable`
: Recherche les fichiers exécutables par l'utilisateur actuel. Dans le cas des répertoires, cela correspond à tous les répertoires auxquels l'utilisateur peut accéder (permission `x`).

`-perm NNNN`
: Cela va correspondre à tous les fichiers qui ont exactement la permission `NNNN`. Par exemple, `-perm 0664` correspondra à tous les fichiers auxquels l'utilisateur et le groupe peuvent accéder en lecture et en écriture et que les autres peuvent lire (ou `rw-rw-r--`).

Vous pouvez ajouter un `-` avant `NNNN` pour rechercher les fichiers qui ont *au moins* la permission spécifiée. Par exemple, `-perm -644` correspondra aux fichiers qui ont au moins les permissions `644` (`rw-r–r--`). Cela inclut un fichier avec `664` (`rw-rw-r--`) ou même `775` (`rwxrwx-r-x`).

`-empty`

Recherche les fichiers et les répertoires vides.

`-size N`

Recherche tous les fichiers de taille `N`, où `N` est par défaut un nombre de blocs de 512 octets. Vous pouvez ajouter des suffixes à `N` pour d'autres unités : `Nc` comptera la taille en octets, `Nk` en kibibytes (KiB, multiples de 1024 octets), `NM` en mebibytes (MiB, multiples de 1024 * 1024) et `NG` pour gibibytes (GiB, multiples de 1024 * 1024 * 1024).

Là encore, vous pouvez ajouter les préfixes `+` ou `-` (qui signifient ici *plus grand que* et *plus petit que*) pour rechercher des tailles relatives. Par exemple, `-size -10M` correspondra à tous les fichiers dont la taille est inférieure à 10 MiB.

Par exemple, pour rechercher des fichiers dans votre répertoire personnel qui contiennent le motif `report` insensible à la casse dans n'importe quelle partie du nom, qui disposent des permissions `0644`, qui ont été consultés il y a dix jours et dont la taille est d'au moins 1 Mib, vous pouvez utiliser la commande suivante :

```
$ find ~ -iname "*report*" -perm 0644 -atime 10 -size +1M
```

Recherche par période

En dehors de la recherche de caractéristiques, vous pouvez également effectuer des recherches en fonction du temps, afin de trouver les fichiers auxquels vous avez accédé, dont les caractéristiques ont été modifiées ou qui ont été modifiés au cours d'une période donnée. Voici les paramètres :

`-amin N`, **`-cmin N`**, **`-mmin N`**

Ceci va correspondre aux fichiers qui ont été accédés, dont les attributs ont été changés ou qui ont été modifiés (respectivement) il y a `N` minutes.

`-atime N`, **`-ctime N`**, **`-mtime N`**

Cela va correspondre aux fichiers qui ont été accédés, dont les attributs ont été changés ou qui ont été modifiés il y a `N*24` heures.

Pour `-cmin N` et `-ctime N`, tout changement d'attribut va déclencher une correspondance, y compris un changement de permissions, l'écriture ou la lecture du fichier. Ce qui rend ces

paramètres particulièrement efficaces, puisque pratiquement toute opération impliquant le fichier déclenchera une correspondance.

L'exemple suivant correspondrait à tout fichier du répertoire courant qui a été modifié il y a moins de 24 heures et dont la taille est supérieure à 100 MiB :

```
$ find . -mtime -1 -size +100M
```

Utiliser `locate` et `updatedb`

`locate` et `updatedb` sont des commandes qui peuvent être utilisées pour trouver rapidement un fichier qui correspond à un motif donné sur un système Linux. Contrairement à `find`, `locate` ne va pas rechercher le motif dans le système de fichiers : au lieu de cela, il va le chercher dans une base de données construite grâce à la commande `updatedb`. Cette approche fournit des résultats très rapides, mais qui peuvent être imprécis en fonction de la date de la dernière mise à jour de la base de données.

La manière la plus simple d'utiliser `locate` est de lui fournir un motif de recherche. Par exemple, pour trouver toutes les images JPEG sur votre système, vous pourrez utiliser `locate jpg`. La liste des résultats peut être assez longue, mais elle devrait ressembler à ceci :

```
$ locate jpg
/home/carol/Downloads/Expert.jpg
/home/carol/Downloads/Hotbit.jpg
/home/carol/Downloads/Mate1.jpg
/home/carol/Downloads/Mate2.jpg
/home/carol/Downloads/Mate3.jpg
/home/carol/Downloads/Pentaro.jpg
/home/carol/Downloads/Sala.jpg
/home/carol/Downloads/pixel_3a_seethrough_1.jpg
/home/carol/Downloads/jpg_specs.doc
```

Lorsque le motif `jpg` est demandé, `locate` va afficher tout ce qui le contient, indépendamment de ce qui le précède ou ce qui le suit. Vous pouvez voir un exemple de ce comportement dans le fichier `jpg_specs.doc` de la liste ci-dessus : il contient le motif, mais l'extension n'est pas `jpg`.

TIP N'oubliez pas qu'avec `locate` vous recherchez des motifs, pas des extensions de fichiers.

Par défaut, le motif est sensible à la casse. Cela veut dire que les fichiers contenant `.JPG` ne seront pas affichés puisque le motif est en minuscules. Pour éviter ceci, passez le paramètre `-i` à `locate`.

Reprenons notre exemple ci-dessus :

```
$ locate -i .jpg
/home/carol/Downloads/Expert.jpg
/home/carol/Downloads/Hotbit.jpg
/home/carol/Downloads/Mate1.jpg
/home/carol/Downloads/Mate1_old.JPG
/home/carol/Downloads/Mate2.jpg
/home/carol/Downloads/Mate3.jpg
/home/carol/Downloads/Pentaro.jpg
/home/carol/Downloads/Sala.jpg
/home/carol/Downloads/pixel_3a_seethrough_1.jpg
```

Notez que le fichier `Mate1_old.JPG`, en gras ci-dessus, n'était pas présent dans les résultats de recherche précédents.

Vous pouvez fournir plusieurs motifs à `locate`, il suffit de les séparer par des espaces. L'exemple ci-dessous effectue une recherche insensible à la casse pour tous les fichiers qui contiennent les motifs `zip` et `jpg` :

```
$ locate -i zip jpg
/home/carol/Downloads/Expert.jpg
/home/carol/Downloads/Hotbit.jpg
/home/carol/Downloads/Mate1.jpg
/home/carol/Downloads/Mate1_old.JPG
/home/carol/Downloads/Mate2.jpg
/home/carol/Downloads/Mate3.jpg
/home/carol/Downloads/OPENMSXPIHAT.zip
/home/carol/Downloads/Pentaro.jpg
/home/carol/Downloads/Sala.jpg
/home/carol/Downloads/gbs-control-master.zip
/home/carol/Downloads/lineage-16.0-20190711-MOD-quark.zip
/home/carol/Downloads/pixel_3a_seethrough_1.jpg
/home/carol/Downloads/jpg_specs.doc
```

Lorsque vous utilisez plusieurs motifs, vous pouvez demander à `locate` de n'afficher que les fichiers qui correspondent à *tous* les motifs. Cela se fait avec l'option `-A`. L'exemple suivant affichera tous les fichiers correspondant aux motifs `.jpg` *et* `.zip` :

```
$ locate -A .jpg .zip
/home/carol/Downloads/Pentaro.jpg.zip
```

Si vous souhaitez compter le nombre de fichiers qui correspondent à un motif donné au lieu d'afficher leur chemin complet, vous pouvez utiliser l'option `-c`. Par exemple, pour compter le nombre de fichiers `.jpg` sur un système :

```
$ locate -c .jpg
1174
```

Un problème avec `locate` c'est qu'il n'affiche que les entrées présentes dans la base de données générée par `updatedb` (située dans `/var/lib/mlocate.db`). Si cette base de données est obsolète, la recherche peut montrer des fichiers qui ont été supprimés depuis la dernière fois qu'elle a été mise à jour. Une manière d'éviter cela est d'ajouter le paramètre `-e`, qui vérifiera si le fichier existe toujours avant de l'afficher dans les résultats.

Bien évidemment, cela ne va pas résoudre le problème des fichiers créés *après* la dernière mise à jour de la base de données et qui n'apparaissent pas. Pour cela, vous devrez mettre à jour la base de données avec la commande `updatedb`. Le temps que cela va prendre dépendra de la quantité de fichiers sur votre disque.

Contrôler le comportement de `updatedb`

Le comportement de `updatedb` peut être contrôlé par le fichier `/etc/updatedb.conf`. C'est un fichier texte où chaque ligne définit une variable. Les lignes vides sont ignorées et les lignes qui commencent par le caractère `#` sont traitées comme des commentaires.

`PRUNEFS=`

Tous les systèmes de fichiers indiqués à la suite de ce paramètre ne seront pas analysés par `updatedb`. La liste doit être séparée par des espaces, et les systèmes de fichiers sont insensibles à la casse, ce qui veut dire que `NFS` et `nfs` sont équivalents.

`PRUNENAMES=`

Une liste de noms de répertoires séparés par des espaces, et qui ne doivent pas être analysés par `updatedb`.

`PRUNEPATHS=`

Une liste de chemins qui doivent être ignorés par `updatedb`. Les chemins doivent être séparés par des espaces et spécifiés de la même manière qu'ils seraient affichés par `updatedb` (par exemple, `/var/spool/media`).

`PRUNE_BIND_MOUNTS=`

Une simple variable `yes` ou `no`. Si elle vaut `yes`, les montages *bind* (répertoires montés ailleurs

avec la commande `mount --bind`) vont être ignorés.

Recherche de fichiers binaires, de pages de manuel et de code source

`which` est une commande très utile qui affiche le chemin complet d'un exécutable. Par exemple, si vous voulez localiser l'exécutable de `bash`, vous pouvez utiliser :

```
$ which bash
/usr/bin/bash
```

Avec l'option `-a`, la commande affichera tous les chemins correspondant à l'exécutable. Notez la différence :

```
$ which mkfs.ext3
/usr/sbin/mkfs.ext3

$ which -a mkfs.ext3
/usr/sbin/mkfs.ext3
/sbin/mkfs.ext3
```

TIP Pour savoir quels répertoires se trouvent dans le `PATH`, utilisez la commande `echo $PATH`. Cela va afficher (`echo`) le contenu de la variable `PATH` (`$PATH`) dans votre terminal.

`type` est une commande comparable qui va afficher des informations sur un binaire, y compris sa localisation et son type. Il suffit d'utiliser `type` suivi du nom de la commande :

```
$ type locate
locate is /usr/bin/locate
```

Le paramètre `-a` fonctionne de la même manière qu'avec `which`, en affichant tous les chemins qui correspondent à l'exécutable. Par exemple :

```
$ type -a locate
locate is /usr/bin/locate
locate is /bin/locate
```

Le paramètre `-t` indique le type de fichier de la commande qui peut être `alias` (alias), `keyword` (mot-clé), `function` (fonction), `builtin` (primitive du *shell*) ou `file` (fichier). Par exemple :

```
$ type -t locate
file

$ type -t ll
alias

$ type -t type
type is a built-in shell command
```

La commande `whereis` est plus polyvalente. En dehors des binaires, elle peut également être utilisée pour indiquer l'emplacement des pages de manuel ou même du code source d'un programme (à condition qu'il soit disponible sur le système). Tapez simplement `whereis` suivi du nom du binaire :

```
$ whereis locate
locate: /usr/bin/locate /usr/share/man/man1/locate.1.gz
```

Le résultat ci-dessus comprend les binaires (`/usr/bin/locate`) et les pages de manuel compressées (`/usr/share/man/man1/locate.1.gz`).

Vous pouvez filtrer sommairement les résultats en utilisant des options en ligne de commande comme `-b` qui n'affichera que les binaires, `-m` qui affichera les pages de manuel ou `-s` qui affichera le code source. En répétant l'exemple ci-dessus, vous obtiendrez :

```
$ whereis -b locate
locate: /usr/bin/locate

$ whereis -m locate
locate: /usr/share/man/man1/locate.1.gz
```

Exercices guidés

1. Admettons qu'un programme doive créer un fichier temporaire à usage unique et qui ne sera plus jamais utilisé après la fermeture du programme. Quel serait le répertoire correct pour créer ce fichier ?

2. Quel est le répertoire temporaire qui *doit* être vidé pendant la procédure de démarrage ?

3. En utilisant `find`, recherchez uniquement dans le répertoire courant les fichiers accessibles en écriture pour l'utilisateur, modifiés au cours des 10 derniers jours et dont la taille est supérieure à 4 GiB.

4. En utilisant `locate`, cherchez tous les fichiers contenant les deux motifs `report` et soit `updated`, `update` ou `updating` dans leurs noms.

5. Comment savoir où se trouve la page de manuel de `ifconfig` ?

6. Quelle variable doit être ajoutée à `/etc/updatedb.conf` pour que `updatedb` ignore les systèmes de fichiers `ntfs` ?

7. Un administrateur système souhaite monter un disque interne (`/dev/sdc1`). D'après le FHS, sous quel répertoire faut-il monter ce disque ?

Exercices d'approfondissement

1. Lorsqu'on utilise `locate`, les résultats sont extraits d'une base de données générée par `updatedb`. Cependant, cette base de données peut être obsolète, ce qui fait que `locate` affiche des fichiers qui n'existent plus. Comment faire en sorte que `locate` n'affiche que les fichiers existants dans ses résultats ?

2. Recherchez tous les fichiers du répertoire actuel ou des sous-répertoires jusqu'à deux niveaux en-dessous, en excluant les systèmes de fichiers montés, et qui contiennent le motif `Status` ou `statute` dans leur nom.

3. En limitant la recherche aux systèmes de fichiers `ext4`, recherchez tous les fichiers dans le répertoire `/mnt` qui ont au moins les permissions d'exécution pour le groupe, qui sont accessibles en lecture pour l'utilisateur actuel et dont l'un des attributs a été modifié au cours des deux dernières heures.

4. Trouvez les fichiers vides qui ont été modifiés il y a plus de 30 jours et qui se trouvent au moins deux niveaux plus bas que le répertoire actuel.

5. Supposons que les utilisateurs `carol` et `john` fassent partie du groupe `mkt`. Trouvez dans le répertoire personnel de `john` tous les fichiers qui sont également accessibles en lecture pour `carol`.

Résumé

Dans cette leçon, vous avez appris l'organisation de base du système de fichiers sur une machine Linux selon le FHS, et comment trouver des binaires et des fichiers, soit par leur nom, soit par leurs caractéristiques. Les commandes suivantes ont été abordées dans cette leçon :

`find`

Une commande polyvalente utilisée pour trouver des fichiers et des dossiers en fonction d'une série de critères de recherche.

`locate`

Un outil qui utilise une base de données locale contenant les emplacements des fichiers stockés localement.

`updatedb`

Met à jour la base de données locale utilisée par la commande `locate`.

`which`

Affiche le chemin complet d'un exécutable.

`whereis`

Affiche l'emplacement des pages de manuel, des fichiers binaires et du code source sur le système.

`type`

Affiche l'emplacement d'un binaire et le type d'application dont il s'agit (programme installé, primitive du *shell* Bash, etc.).

Réponses aux exercices guidés

1. Admettons qu'un programme doive créer un fichier temporaire à usage unique et qui ne sera plus jamais utilisé après la fermeture du programme. Quel serait le répertoire correct pour créer ce fichier ?

   Puisque nous n'avons plus besoin du fichier une fois que le programme a fini de s'exécuter, le répertoire correct est `/tmp`.

2. Quel est le répertoire temporaire qui *doit* être vidé pendant la procédure de démarrage ?

   Le répertoire est `/run` ou `/var/run`, selon les systèmes.

3. En utilisant `find`, recherchez uniquement dans le répertoire courant les fichiers accessibles en écriture pour l'utilisateur, modifiés au cours des 10 derniers jours et dont la taille est supérieure à 4 GiB.

   Pour cela, vous aurez besoin des paramètres `-writable`, `-mtime` et `-size` :

   ```
   find . -writable -mtime -10 -size +4G
   ```

4. En utilisant `locate`, cherchez tous les fichiers contenant les deux motifs `report` et soit `updated`, `update` ou `updating` dans leurs noms.

   Étant donné que `locate` doit correspondre à plusieurs motifs, utilisez l'option `-A` :

   ```
   locate -A "report" "updat"
   ```

5. Comment savoir où se trouve la page de manuel de `ifconfig` ?

   Utilisez le paramètre `-m` pour `whereis` :

   ```
   whereis -m ifconfig
   ```

6. Quelle variable doit être ajoutée à `/etc/updatedb.conf` pour que `updatedb` ignore les systèmes de fichiers `ntfs` ?

   La variable est `PRUNEFS=` suivie du type de système de fichiers : `PRUNEFS=ntfs`.

7. Un administrateur système souhaite monter un disque interne (`/dev/sdc1`). D'après le FHS,

sous quel répertoire faut-il monter ce disque ?

Théoriquement, le disque peut être monté n'importe où. Ceci étant dit, le FHS recommande que les montages temporaires soient effectués en-dessous de `/mnt`.

Réponses aux exercices d'approfondissement

1. Lorsqu'on utilise `locate`, les résultats sont extraits d'une base de données générée par `updatedb`. Cependant, cette base de données peut être obsolète, ce qui fait que `locate` affiche des fichiers qui n'existent plus. Comment faire en sorte que `locate` n'affiche que les fichiers existants dans ses résultats ?

   Ajoutez le paramètre `-e` à locate, comme dans `locate -e PATTERN`.

2. Recherchez tous les fichiers du répertoire actuel ou des sous-répertoires jusqu'à deux niveaux en-dessous, en excluant les systèmes de fichiers montés, et qui contiennent le motif `Status` ou `statute` dans leur nom.

   Rappelez-vous que pour `-maxdepth` vous devez également prendre en compte le répertoire courant, nous voulons donc trois niveaux (le répertoire courant et deux niveaux inférieurs) :

   ```
   find . -maxdepth 3  -mount -iname "*statu*"
   ```

3. En limitant la recherche aux systèmes de fichiers `ext4`, recherchez tous les fichiers dans le répertoire `/mnt` qui ont au moins les permissions d'exécution pour le groupe, qui sont accessibles en lecture pour l'utilisateur actuel et dont l'un des attributs a été modifié au cours des deux dernières heures.

   Utilisez le paramètre `-fstype` de `mount` pour limiter la recherche à des types de systèmes de fichiers spécifiques. Un fichier lisible par l'utilisateur actuel doit avoir au moins `4` dans le premier chiffre des permissions, et un fichier exécutable par le groupe doit avoir au moins `1` dans le deuxième chiffre. Puisque nous ne nous soucions pas des permissions des autres, nous pouvons utiliser `0` pour le troisième chiffre. Utilisez `-cmin N` pour filtrer les changements d'attributs récents, en vous rappelant que N est spécifié en minutes. Donc :

   ```
   find /mnt -fstype ext4 -perm -410 -cmin -120
   ```

4. Trouvez les fichiers vides qui ont été modifiés il y a plus de 30 jours et qui se trouvent au moins deux niveaux plus bas que le répertoire actuel.

   Le paramètre `-mindepth N` peut être utilisé pour limiter la recherche à `N` niveaux au moins, mais rappelez-vous que vous devez inclure le répertoire courant dans le décompte. Utilisez `-empty` pour détecter les fichiers vides, et `-mtime N` pour vérifier la date de modification. Donc :

```
find . -empty -mtime +30 -mindepth 3
```

5. Supposons que les utilisateurs `carol` et `john` fassent partie du groupe `mkt`. Trouvez dans le répertoire personnel de `john` tous les fichiers qui sont également accessibles en lecture pour `carol`.

   Sachant qu'ils sont membres du même groupe, nous avons besoin d'au moins un `r` (4) sur les permissions du groupe, et nous ne nous préoccupons pas des autres. Donc :

   ```
   find /home/john -perm -040
   ```

Impression

PDF généré: 2025-09-16

Les supports de cours du LPI sont une initiative du Linux Professional Institute (https://lpi.org). Les supports de cours et leurs traductions sont disponibles à l’adresse https://learning.lpi.org.

Pour toute question ou commentaire sur cette édition ou sur l’ensemble du projet, contactez-nous à l’adresse suivante : learning@lpi.org.